Contents lists available at ScienceDirect

ISA Transactions

journal homepage: www.elsevier.com/locate/isatrans

Practice article

Analysis of a Fast Control Allocation approach for nonlinear over-actuated systems

M.F. Santos [a], L.M. Honório [b,*], A.P.G.M. Moreira [c], P.A.N. Garcia [b], M.F. Silva [b], V.F. Vidal [b]

[a] Department of Electroelectronics, CEFET-MG, José Peres Street, 558, Leopoldina, Brazil
[b] Department of Energy Systems, UFJF, José Lourenço Kelmer Street, Juiz de Fora, Brazil
[c] Department of Electrical and Computer Engineering, FEUP, UPorto, Dr. Roberto Frias Street, Porto, Portugal

ARTICLE INFO

Article history:
Received 1 August 2019
Received in revised form 26 July 2021
Accepted 5 August 2021
Available online 19 August 2021

Keywords:
Nonlinear control allocation
Over-actuated systems
Linear-partially-dependent system
Tilt-rotors UAV

ABSTRACT

Autonomous Robots with multiple directional thrusters are normally over-actuated systems that require nonlinear control allocation methods to map the forces that drive the robot's dynamics and act as virtual control variables to the actuators. This process demands computational efforts that, sometimes, are not available in small robotic platforms. The present paper introduces a new control allocation approach with fast convergence, high accuracy, and dealing with complex nonlinear problems, especially in embedded systems. The adopted approach divides the desired nonlinear system into coupled linear problems. For that purpose, the Real Actions (RAs) and Virtual Control Variables (VCVs) are broke in two or more sets each. While the RA subsets are designed to linearize the system according to different input subspaces, the VCV is designed to be partially coupled to overlap the output subspaces. This approach generates smaller linear systems with fast and robust convergence used sequentially to solve nonlinear allocation problems. This methodology is assessed in mathematical tutorial cases and over-actuated UAV simulations.

1. Introduction

In dynamical systems, Real Actions (RAs) are actuators that direct the behavior of mechanical devices like autonomous vehicles. When an actuator impacts over just one Degree of Freedom (DoF), it is only necessary to design its respective control strategy. However, there are several systems with more actuators than DoFs. In such cases, control states are used as DoFs for several reasons. For instance, it reduces the number of control loops, makes the design more intuitive and, finally, it makes it is easier to deal with fault-tolerant strategies [1]. In that way, the actions computed by a control system are based on forces that act directly over the DoFs. These forces are also known as Virtual Control Variables (VCVs) and are fewer that the real actuators. Therefore, the control of autonomous systems is not limited to find the best-closed loop methodology and adjustment; several cases require a Control Allocation Techniques (CATs) to map the VCVs into the final RAs.

Two metrics analyze the performance of a CAT; the error between the evaluated and desired forces and the necessary computational time. The first one is related to the methodology's accuracy and, the other, its real-time applicability. The final system's stability depends on both metrics.

Many problems are categorized as fully or under-actuated systems and use simple CAT approaches. There are also cases where it is possible to simplify over-actuated systems to fit one of those two categories [2]. However, depending on the system's physical characteristics, it is necessary a complex CAT. Ref. [1] shows that the control allocation is a optimization problem, and it is divided into three main categories; unconstrained linear, constrained linear, and constrained nonlinear, where each one has several different variations. The most simple one is the unconstrained linear where the error function is minimized by adopting single-step algorithms such as least-square [3]. When it is necessary to consider the physical limits, the actuators are grouped, ranked, and if any action saturates, the respective group is frozen. When nonlinear systems are considered, the best approach is to adopt the iterative Karush–Kuhn–Tucker/Lagrange Multipliers (KKT/LM) [4] based approaches. However, these approaches invert the high-dimensional Hessian matrix at each iteration, which is prohibitive in systems with low computational capacity.

A deeper analysis of these approaches shows that the CATs are widespread and established for linear models. Still, when considering nonlinear ones, the numerical optimization approach is a possible alternative [5], but it is a special attention on the computational effort. In general, CAT methodologies are performed through: Direct Allocation; Pseudo-Inverse; Linear Programming; and Quadratic Programming [1].

* Corresponding author.
E-mail address: leonardo.honorio@ufjf.edu.br (L.M. Honório).

https://doi.org/10.1016/j.isatra.2021.08.010

The Direct CAT considers the unconstrained allocation space with a particular pseudo-inverse, satisfying only the control constraints. Consequently, it searches for an allocated parameter set that preserves the VCV direction [6]. Moreover, this approach is not trivial for cases where the VCV set size is too large [1].

For Pseudo-Inverse methods, the first step consists of solving the problem without saturation constraints [7]. In the end, if the result satisfies the previously unrestricted RAs, no additional steps are required. Otherwise, the RAs free optimal vector is projected into the limits [8]. Next, the unconstrained elements are recomputed through a similar procedure using a reduced pseudo-inverse to decrease the difference between the desired and allocated VCVs. After this, the new RAs can be saturated, and the redistribution procedure is repeated until a feasible solution or stopping criteria is reached. This method is efficient and straightforward, but it demonstrates that the sub-optimal control allocation does not guarantee a possible solution and the error minimization [1].

By choosing the 2-norm, the control allocation becomes a Quadratic Programming problem, which is also solved by numerical methods [9]. Moreover, a feasible solution always exists and permits a unique optimal solution if all weights in the cost function are necessarily positive [10]. Concerning numerical methods, three algorithms deserve remarks: active set, interior-point, and fixed-point methods.

A high-level analysis of nonlinear approaches shows that its main problem is computational complexity, which requires high computational efforts. Thus, although it is possible to have complex nonlinear control allocations algorithms running over powerful companion board processors [11], in small autonomous robotic systems this scenario increases weight and power consumption, impacting mission time. As a solution, it is presented here a new control allocation approach named Fast Control Allocation (FCA), designed to be used in small autonomous robots with multiple directional propulsion systems. It is based in two main ideas: separation and mapping. While the first divides the nonlinear RAs in independent groups, i.e., Real Actions Sets (RASs), as a mean of breaking the nonlinear problem into linear subproblems, the mapping chooses which VCV will be related with the previously linear subproblems. It means that each RAS generates one subproblem, each one associated with a different set of VCVs. This problem-breaking process overlaps the linear systems over the nonlinear one generation intersections and multiples coupled linear systems which are solved interactively. Although a similar approach can be seen in [12,13], the same state-variables (θ and V, as input)s have different functions (active and reactive power, as outputs), both systems are nonlinear. The approach transforms a high-dimensional system into two smaller ones to reduce the computational complexity of inverting the Hessian matrix. Analyzing the survey present in [1], and other recent studies [14–21], it is possible to verify that this is a applicable new methodology. Therefore this work presents the theoretical details of the FCA approach with a deeper convergence analysis and the assessment of its full potential.

The rest of the paper is organized as follows: Section 2 presents the control allocation problem, Section 3 proposed FCA methodology, resulting in partially-dependent linear subsystems; Section 4 illustrates some mathematical tutorial cases, considering different possibilities of RAS superposition in the breaking step; Section 5 applies the method in an over-actuated tilt-rotor Unmanned Aerial Vehicle (UAV); and Section 6 concludes this work.

2. The control allocation problem

As stated before, the control of autonomous systems is not limited to find the best controller topology; some situations demand to map the VCVs into the final control signals. Moreover, a reasonable allocation strategy must be transparent to the controller. Fig. 1 shows this process where X_{sp} is the set point vector containing all state variables related to the system's desired position and velocity. τ is a vector of the desirable signals along each DoF that runs the system's dynamics. The vector $u(t-T)$ contains the physical signals that feed the actuators where T is the delay response time, $\hat{\tau}$ is the estimated vector containing the torques provided by u.

Also, the approach must ensure to minimize $\left|\tau - \hat{\tau}\right|$ and T with u respecting all physical limits.

Although nonlinear approaches are the most precise approach to evaluate the control allocation problem, they need a high degree of computational resources. The present approach improves the velocity by dividing the original nonlinear system into two coupled linear ones, which are solved interactively. To demonstrate the FCA, Section 2.1 shows the traditional linear approach, Section 2.2 the nonlinear formulation, and 3 shows the linearization process.

2.1. Traditional linear approach

A linear CAT transforms the virtual variables $\boldsymbol{\tau} \in \mathbb{R}^m$ into the real control signals $\boldsymbol{u} \in \mathbb{R}^n$, where m is the number of DoFs, and n is the number of actuators. Now consider a Control Effectiveness Matrix (CEM) that embeds the constructive parameters Control Effectiveness Matrix (CEM) [1]:

$$\hat{\boldsymbol{\tau}} = \boldsymbol{M}(\boldsymbol{\Gamma})\boldsymbol{u} \tag{1}$$

where $\hat{\boldsymbol{\tau}} \in \mathbb{R}^m$ is the estimated virtual variables given the real signals $\boldsymbol{u}$, $\boldsymbol{M}(\boldsymbol{\Gamma}) \in \mathbb{R}^{m\times n}$ represents the actuators behavior, and $\boldsymbol{\Gamma} \in \mathbb{R}^p$ are the necessary set of parameters.

Equation $\boldsymbol{\varepsilon} = |\hat{\boldsymbol{\tau}} - \boldsymbol{\tau}|$ defines the error between the desired and actual virtual control variables, which leads to:

$$\boldsymbol{\varepsilon} = |\boldsymbol{M}(\boldsymbol{\Gamma})\boldsymbol{u} - \boldsymbol{\tau}| \tag{2}$$

Therefore, the control allocation is the solution to the following optimization problem:

$$\begin{aligned} &\text{Minimize} \quad f(\boldsymbol{u}) = \tfrac{1}{2}(\boldsymbol{M}(\boldsymbol{\Gamma})\boldsymbol{u} - \boldsymbol{\tau})^T(\boldsymbol{M}(\boldsymbol{\Gamma})\boldsymbol{u} - \boldsymbol{\tau}) \\ &\text{Subject to} \quad \underline{\boldsymbol{u}} \leq \boldsymbol{u} \leq \overline{\boldsymbol{u}} \end{aligned} \tag{3}$$

where $\overline{\boldsymbol{u}}$ and $\underline{\boldsymbol{u}}$ are the upper and lower limits of the real control variables.

It is necessary to use the Karush–Kuhn–Tucker (KKT) optimality conditions to solve Eq. (3) [10]. Considered $\boldsymbol{M}(\boldsymbol{\Gamma}) = \boldsymbol{M}$, it provides Eq. (4).The constraints are considered by an outside routine approximating the evaluated result to the closest feasible one.

Considering the Lagrangian

$$L(\boldsymbol{u}) = f(\boldsymbol{u}) \tag{4}$$

the KKT condition is

$$\frac{\partial L}{\partial \boldsymbol{u}}(\boldsymbol{u}) = \boldsymbol{M}^T\boldsymbol{M}\boldsymbol{u} - \boldsymbol{M}^T\boldsymbol{\tau} = 0 \tag{5}$$

the optimum solution for (3) is:

$$\boldsymbol{u} = \left(\boldsymbol{M}^T\boldsymbol{M}\right)^{-}\boldsymbol{M}^T\boldsymbol{\tau} \tag{6}$$

If a feasible control set ($\underline{\boldsymbol{u}} \leq \boldsymbol{u} \leq \overline{\boldsymbol{u}}$) is not found, the system is requiring more forces from the actuators than their capabilities. In such case, it is necessary to reduce the desired performance goals by finding the closest feasible solution $\tilde{\boldsymbol{u}}$.

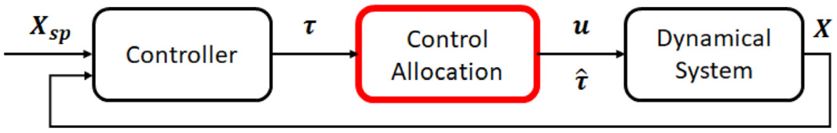


Fig. 1. The traditional control allocation problem.

Finally, if $\boldsymbol{M}$ has $m < n$, then (M^TM) will probably have a rank lower than n, making it numerically unstable. In such case, it is possible to use the follow representation

$$\boldsymbol{u} = \boldsymbol{M}^{\dagger}\boldsymbol{\tau} \tag{7}$$

where $\boldsymbol{M}^{\dagger}$ is the Moore–Penrose pseudo-inverse matrix that can be evaluated by:

$$\boldsymbol{M}^{\dagger} = \boldsymbol{M}^T(\boldsymbol{M}\boldsymbol{M}^T)^{-} \text{ , if } m < n \text{ or} \tag{8}$$

$$\boldsymbol{M}^{\dagger} = (\boldsymbol{M}^T\boldsymbol{M})^{-}\boldsymbol{M}^T \text{ , if } m > n \tag{9}$$

2.2. Nonlinear approach

The nonlinear control allocation algorithms [22], are more complex and time-consuming than the linear ones.

Consider the following system:

$$\boldsymbol{\kappa} = \boldsymbol{K}(\boldsymbol{u}) - \hat{\boldsymbol{\tau}} \tag{10}$$

where $\boldsymbol{K}(\boldsymbol{u}) \in \mathbb{R}^m$ is now a nonlinear control allocation problem and $\boldsymbol{\kappa} \in \mathbb{R}^m$ is the residual error considering the desired and achieved control allocation.

Thus, the problem is given by:

$$\begin{array}{ll}\text{Minimize} & f(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{\kappa}^T\boldsymbol{\kappa} \\ \text{Subject to} & \underline{\boldsymbol{u}} \le \boldsymbol{u} \le \overline{\boldsymbol{u}}\end{array} \tag{11}$$

The inequality in Eq. (11) must be manipulated by adding the slack variables $\boldsymbol{s_1} \in \mathbb{R}^n$ and $\boldsymbol{s_2} \in \mathbb{R}^n$

$$\begin{array}{ll}\text{Minimize} & f(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{\kappa}^T\boldsymbol{\kappa} \\ \text{Subject to} & \boldsymbol{u} - \overline{\boldsymbol{u}} + \boldsymbol{s}_1 = 0 \\ & \boldsymbol{u} - \underline{\boldsymbol{u}} - \boldsymbol{s}_2 = 0\end{array} \tag{12}$$

giving the following Lagrangian

$$L(\boldsymbol{u}, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2) = f(\boldsymbol{u}) - \boldsymbol{\lambda_1}\left(\boldsymbol{u} - \overline{\boldsymbol{u}} + \boldsymbol{s}_1\right) - \boldsymbol{\lambda_2}\left(\boldsymbol{u} - \underline{\boldsymbol{u}} - \boldsymbol{s}_2\right) \tag{13}$$

where $\boldsymbol{\lambda_1} \in \mathbb{R}^n$ and $\boldsymbol{\lambda_2} \in \mathbb{R}^n$ are the Lagrange multipliers.

Then, the KKT condition for Eq. (13) is:

$$\frac{\partial L}{\partial \boldsymbol{w}} = 0 \tag{14}$$

where $w = (\boldsymbol{u}, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2)$.

3. Fast control allocation - FCA

3.1. Motivation

Sections 2.1 and 2.2 have shown two ways of solving control allocation problems. While the approach demonstrated in 2.1 is fast and easily implemented, the mathematical complexity of solving 2.2 is high. Considering that the control allocation system of most of the traditional drones and small robotic systems can be represented by linear relationships, advanced typologies with directional thrusters are intrinsically nonlinear. In that case, the Flight Control Units (FCU) embedded in such drones and other directional types of small robotic systems do not have the computational capability to solve a full nonlinear control allocation problem. In such a case, it is necessary to have companion boards for ancillary processing.

The motivation behind this work is to provide a robust and fast approach that can run in embedded small FCUs.

The main idea is that, to break a well known nonlinear control allocation problem $\hat{\boldsymbol{\tau}} = \boldsymbol{M}(\boldsymbol{u})$ into two or more linear problems. While $\boldsymbol{M}(\boldsymbol{u})$ is previously well known and evaluated at the design time, it is not fit for running in small FCU units.

While this work does define a formal mathematical procedure of the breaking process, it provides a well-structured heuristic based on linearization. It means that the system must be analyzed and the variables related to the nonlinearities must be divided into individual groups. These groups will generate multiple linear versions of the original system $\boldsymbol{M}(\boldsymbol{u})$. Each one of these new versions is easily solved by a given FCU. Moreover, this approach changes a high-dimensional nonlinear system into linear ones which are solved individually and interactively.

Analyzing the survey presented in [1], and other recent works [14–21], it is possible to verify the originality and applicability of such technique.

3.2. Mathematical framework

The FCA breaks Eq. (10) into real actions and virtual variables and generates a faster multiple linear version as the presented in Eq. (6). Thus, consider that the nonlinear system

$$\hat{\boldsymbol{\tau}} = \boldsymbol{M}(\boldsymbol{u}) \tag{15}$$

where $\hat{\boldsymbol{\tau}}$ and $\boldsymbol{u}$ are the virtual and real controls respectively. The main idea is to generate two systems such as:

$$\hat{\boldsymbol{\tau}}_a = \boldsymbol{M_a}(\boldsymbol{u_b})\boldsymbol{u_a} \tag{16}$$

$$\hat{\boldsymbol{\tau}}_b = \boldsymbol{M_b}(\boldsymbol{u_a})\boldsymbol{u_b} \tag{17}$$

where $\boldsymbol{u}_a \in \mathbb{R}^q$ and $\boldsymbol{u}_b \in \mathbb{R}^r$ are real actions, $\hat{\boldsymbol{\tau}}_a \in \mathbb{R}^{ma}$, $\hat{\boldsymbol{\tau}}_b \in \mathbb{R}^{mb}$ are virtual variables and $\boldsymbol{M_a}(\boldsymbol{u_b}) \in \mathbb{R}^{ma\times q}$ and $\boldsymbol{M_b}(\boldsymbol{u_a}) \in \mathbb{R}^{mb\times r}$ are subsystems linearized around $\boldsymbol{u_b}$ and $\boldsymbol{u_a}$ respectively.

The mathematical justification of the proposed algorithm will be constructed in steps. First, Section 3.3 will prove the convergence process for recursive linearization when the problem is broken using two different control sets and has just one solution space. Section 3.4 will further show the convergence process for an unconstrained system where the virtual forces are arranged among different subspaces. Finally, Section 3.5 will extend these ideas and show the convergence process for a constrained scenario, which is the case for most real applications.

3.3. Recursive linearization

First, consider the systems represented by Eqs. (16) and (17) have the same solution space, i.e., $\tau = \tau_a = \tau_b$. In this particular scenario $M_a(u_b)$ and $M_b(u_a)$ map two vectors into the same solution space. It is also considered that $\boldsymbol{u_a}$ and $\boldsymbol{u_b}$ are subsets from $\boldsymbol{u}$ that are capable of breaking the system‘s nonlinearities. If one subset is fixed considering, for instance, $\boldsymbol{u_b} = x_b$ or $\boldsymbol{u_a} = x_a$, the problem is now:

$$\begin{array}{ll}\text{Minimize} & f_i(\boldsymbol{u_a}, \boldsymbol{u_b}) \\ \text{Subject to} & g_i(\boldsymbol{u_a}, \boldsymbol{u_b}) = 0\end{array} \tag{18}$$

where where $x_a \in \mathbb{R}^q$ and $x_b \in \mathbb{R}^r$ are fixed values at a given interaction. , $f_i(\boldsymbol{u_a}, \boldsymbol{u_b})$ the same as presented in Eq. (12), i.e,

$$f_i(\boldsymbol{u_a}, \boldsymbol{u_b}) = \frac{1}{2}\kappa(\boldsymbol{u_a}, \boldsymbol{u_b})^T\kappa(\boldsymbol{u_a}, \boldsymbol{u_b}) \tag{19}$$

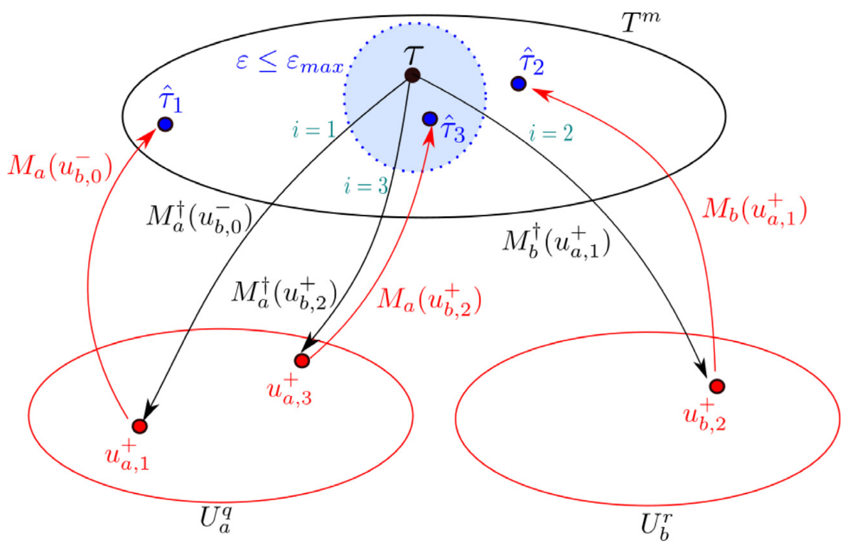


Fig. 2. Convergence analysis for the FCA when the reprojections are into the same vector space $\boldsymbol{T}^m$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

with

$$\kappa(\boldsymbol{u_a}, \boldsymbol{u_b}) = K(\boldsymbol{u_a}, \boldsymbol{u_b}) - \boldsymbol{\tau} \tag{20}$$

and g_i may assume one of two forms, depending of the desired linearization vector u_a or u_b. These two forms are shown next:

$$g_i(\boldsymbol{u_a}, \boldsymbol{u_b}) = \boldsymbol{u_b} - \boldsymbol{x_b} = 0\Big|_{u_b=x_b} \tag{21}$$

$$g_i(\boldsymbol{u_a}, \boldsymbol{u_b}) = \boldsymbol{u_a} - \boldsymbol{x_a} = 0\Big|_{u_a=x_a} \tag{22}$$

Supposing that $\boldsymbol{u^0} = (\boldsymbol{u^0_a}, \boldsymbol{u^0_b})$ is a vector such that $\dot{\boldsymbol{u}} = \kappa(\boldsymbol{u^0_a}, \boldsymbol{u^0_b}) = 0$. In such case, $\boldsymbol{u^0}$ is an equilibrium point and the Taylor Series [23,24] expansion around it is given by:

$$\kappa(u^0_a + \Delta u_a, u^0_b + \Delta u_b) = \kappa(u^0_a, u^0_b) + \Delta u_a \left[\frac{\partial \kappa}{\partial u_a}\right]_{(x_a,x_b)} + \Delta u_b \left[\frac{\partial \kappa}{\partial u_b}\right]_{(x_a,x_b)} \tag{23}$$

providing

$$\kappa(u_a, u_b) \approx \frac{\partial \kappa}{\partial u_a}\Big|_{u_a=x_a} u_a + \frac{\partial \kappa}{\partial u_b}\Big|_{u_b=x_b} u_b \tag{24}$$

considering

$$M_a(u_b) = \left[\frac{\partial \kappa}{\partial u_{a1}}, \dots, \frac{\partial \kappa}{\partial u_{an}}\right]^T \tag{25}$$

$$M_b(u_a) = \left[\frac{\partial \kappa}{\partial u_{b1}}, \dots, \frac{\partial \kappa}{\partial u_{bn}}\right]^T \tag{26}$$

and that any linearization would proceed only around a given set at time (i.e $\boldsymbol{u_a}$ or $\boldsymbol{u_b}$), Eq. (19) can be rewrite as

$$f_i(\boldsymbol{u_a}, \boldsymbol{u_b}) = (\boldsymbol{M_a}(\boldsymbol{u_b})\boldsymbol{u_a} - \boldsymbol{\tau})^T(\boldsymbol{M_a}(\boldsymbol{u_b})\boldsymbol{u_a} - \boldsymbol{\tau}) + (\boldsymbol{M_b}(\boldsymbol{u_a})\boldsymbol{u_b} - \boldsymbol{\tau})^T(\boldsymbol{M_b}(\boldsymbol{u_a})\boldsymbol{u_b} - \boldsymbol{\tau}) \tag{27}$$

Finally, considering that the Lagrangian from Eq. (18) is given by:

$$L(u_a, u_b, \lambda) = f_i(\boldsymbol{u_a}, \boldsymbol{u_b}) - \lambda g_i(\boldsymbol{u_a}, \boldsymbol{u_b})\Big|_{u_b=x_b} \tag{28}$$

and also considering as a first step ($i = 1$) the partial linearization around $\boldsymbol{u_b} = \boldsymbol{x_b}$, it would provide the following KKT conditions:

$$\frac{\partial L}{\partial \boldsymbol{u_a}} = \frac{\partial f_1}{\partial \boldsymbol{u_a}} = 0 \tag{29}$$

$$\frac{\partial L}{\partial \boldsymbol{u_b}} = \frac{\partial f_1}{\partial \boldsymbol{u_b}} - \boldsymbol{\lambda}\frac{\partial g}{\partial \boldsymbol{u_b}} = \frac{\partial f_1}{\partial \boldsymbol{u_b}} - \boldsymbol{\lambda} = 0 \tag{30}$$

$$\frac{\partial L}{\partial \boldsymbol{\lambda}} = g_1(\boldsymbol{u_a}, \boldsymbol{u_b}) = \boldsymbol{u_b} - \boldsymbol{x_b} = 0 \tag{31}$$

Since $\boldsymbol{u_b} = \boldsymbol{x_b}$ and $\boldsymbol{\lambda} = \frac{\partial f_1}{\partial \boldsymbol{u_b}}$ for any possible $\boldsymbol{u_a}$ and $\boldsymbol{u_b}$ the only active equation to solve the KKT condition is (29) and its solution for $\boldsymbol{u_a}$ is given by Eq. (7).

Now, using the result of $\boldsymbol{u_a}$ as a fixed value and evaluating the minimum of $f_2(\boldsymbol{u_a}, \boldsymbol{u_b})$ by searching in the $\boldsymbol{u_b}$ space and so on, it will produce a monotonic minimization sequence where

$$f_1(\underline{\boldsymbol{u_a}}, \boldsymbol{u_b}) \geq f_2(\boldsymbol{u_a}, \underline{\boldsymbol{u_b}}) \geq f_3(\underline{\boldsymbol{u_a}}, \boldsymbol{u_b}) \geq \cdots \tag{32}$$

will converge to a local minimum. The underlined sets, i.e. $(\underline{u_a}, \underline{u_b})$ are considered variables and, without the underline, parameters. Finally, if $\boldsymbol{u_a}$, $\boldsymbol{u_b}$ are base of the solution space, the evaluated local minimum from this recursive problem, i.e.,

$$\frac{\partial f}{\partial \boldsymbol{u_a}} = 0 \tag{33}$$

$$\frac{\partial f}{\partial \boldsymbol{u_b}} = 0 \tag{34}$$

is also a local minimum for $f(\boldsymbol{u})$.

This approach is better visualized in Fig. 2 where

$$\boldsymbol{T}^m = \{(x_1, \dots, x_m) : x_j \in \mathbb{R} \text{ for } j = 1, \dots, m\}. \tag{35}$$

is the vector space of all possible virtual forces,

$$\boldsymbol{U}^q_a = \{(x_1, \dots, x_q) : x_j \in \mathbb{R} \text{ for } j = 1, \dots, q\}. \tag{36}$$

$$\boldsymbol{U}^r_b = \{(x_1, \dots, x_r) : x_j \in \mathbb{R} \text{ for } j = 1, \dots, r\}. \tag{37}$$

are vector spaces for all possible control actions u_a and u_b. Moreover,

$$\boldsymbol{M}^\dagger_a(u_b) : \boldsymbol{T}^m \to \boldsymbol{U}^q_a \tag{38}$$

$$\boldsymbol{M}^\dagger_b(u_a) : \boldsymbol{T}^m \to \boldsymbol{U}^r_b \tag{39}$$

are linear maps responsible for projecting the virtual space into the action spaces. Similarly,

$$\boldsymbol{M}_a(u_b) : \boldsymbol{U}^q_a \to \boldsymbol{T}^m \tag{40}$$

$$\boldsymbol{M}_b(u_a) : \boldsymbol{U}^r_b \to \boldsymbol{T}^m \tag{41}$$

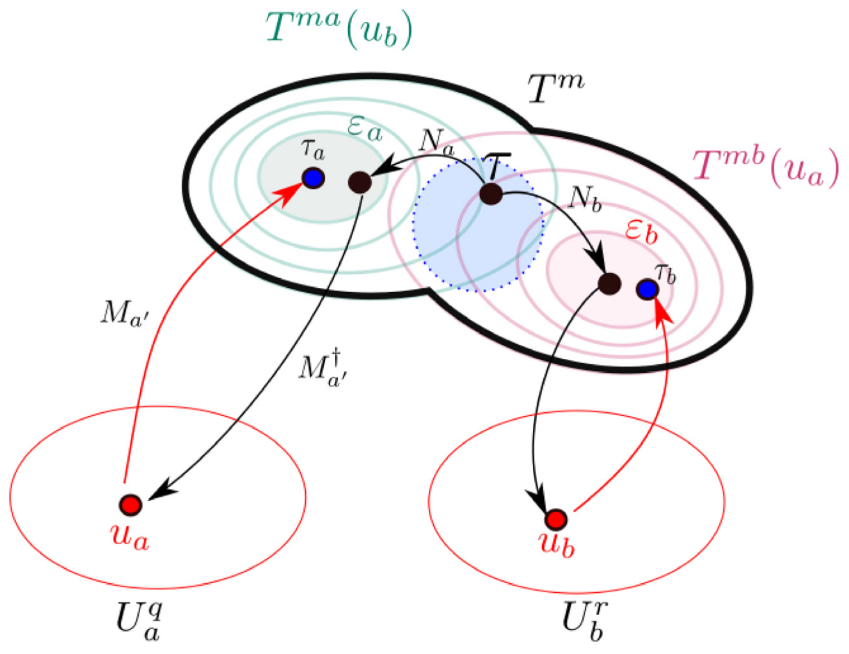


Fig. 3. The FCA reprojections with two different vector spaces $\boldsymbol{T}^{ma}$ and $\boldsymbol{T}^{mb}$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

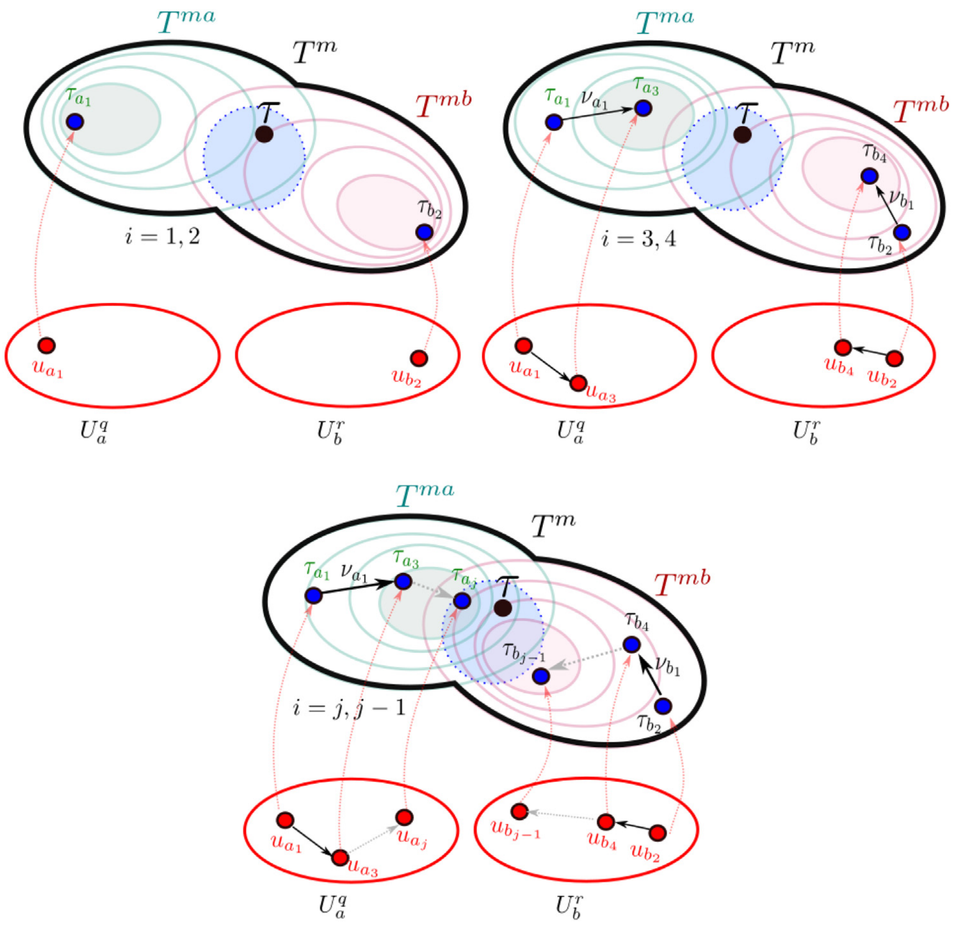


Fig. 4. The convergence process and iterative changes in the solution space.

are also linear maps responsible for projecting the real actions into the virtual space. This reprojection ($\hat{\tau}_i$) is used to evaluate the convergence error.

The blue region is defined by $\|\tau - \hat{\tau}_i\| \leq \varepsilon_{max}$. For the scenario illustrated by Fig. 2, the resulting control actions, for a given τ is represented by the set $(u^+_{a,3}, u^+_{b,2})$.

3.4. Unconstrained partial mapping functions

One objective of the FCA approach is to reduce computational complexity to increase performance in low processing hardware. The approach of doing so is to break a complex nonlinear system into two linear ones, and also to reduce the time to solve each new subsystem. The goal for second part can be achieve by braking the solution space in two lower dimensional vector spaces.

For that, consider the vector space $\boldsymbol{T}^m$ and two subspaces $\boldsymbol{T}^{ma}$ and $\boldsymbol{T}^{mb}$ with

$$\boldsymbol{T}^m = \boldsymbol{T}^{ma} \cup \boldsymbol{T}^{mb} \tag{42}$$

From basic definition, if $ma < m$ it is not possible to find any set of vectors in $\boldsymbol{T}^{ma}$, such as $\boldsymbol{\nu}_a = (\nu_{a_1}, \dots, \nu_{a_{ma}})$ that spans the vector space $\boldsymbol{T}^m$ where the spam is defined by

$$span(\nu_{a_1}, \dots, \nu_{a_{ma}}) = \left\{a_1\nu_{a_1}, \dots, a_{ma}\nu_{a_{ma}} : a_i \in \mathbb{R};\ \nu_{a_i} \in \boldsymbol{T}^m\right\} \tag{43}$$

In such case it is possible to consider $\boldsymbol{T}^{ma}$ as a partial projection of the solution space $\boldsymbol{T}^m$ and any vector $\nu_{a_i} \in \boldsymbol{T}^m$ will provide just a restricted direction considering the entire solution space.

Fig. 3 shows that, considering $\tau \in \boldsymbol{T}^m$, the linear maps that projects τ into $\boldsymbol{T}^{ma}$ and $\boldsymbol{T}^{mb}$ are defined by

$$\boldsymbol{N}_a : \boldsymbol{T}^m \to \boldsymbol{T}^{ma} \tag{44}$$

$$\boldsymbol{N}_b : \boldsymbol{T}^m \to \boldsymbol{T}^{mb} \tag{45}$$

The mapping functions from $\boldsymbol{T}^{ma}$ and $\boldsymbol{T}^{mb}$ to $\boldsymbol{U}_a^q$ and $\boldsymbol{U}_b^r$ are, respectively

$$\boldsymbol{M}_{a'}^{\dagger}(u_b) : \boldsymbol{T}^{ma} \to \boldsymbol{U}_a^q \tag{46}$$

$$\boldsymbol{M}_{b'}^{\dagger}(u_a) : \boldsymbol{T}^{mb} \to \boldsymbol{U}_b^r \tag{47}$$

Finally, the inverse mapping functions (reprojections) are represented by

$$\boldsymbol{M}_{a'}(u_b) : \boldsymbol{U}_a^q \to \boldsymbol{T}^{ma} \tag{48}$$

$$\boldsymbol{M}_{b'}(u_a) : \boldsymbol{U}_b^r \to \boldsymbol{T}^{mb} \tag{49}$$

Fig. 3 shows these operations where the light green and light red lines represent the contour fields of T^{ma} and T^{mb}, the green (ε_a) and red (ε_b) circles represent the minimum region of each solution space for a given u_b and u_a, respectively. The dark blue circles τ_a and τ_b represent the reprojection point in each solution space. The region space that minimizes $\tau_a = M_{a'}u_a$ and $\tau_b = M_{b'}u_b$ are dependent of the u_b and u_a, respectively.

Fig. 4 shows the process in which the convergence regions change in direction of $\boldsymbol{\tau}$. The mapping processes were not displayed for better visualization, and the remapping was defined as dotted light red arrows. The dotted light gray arrows indicate multiple iterative processes.

3.5. Constrained partial mapping functions

The convergence process of the FCA approach were demonstrated in Sections 3.3 and 3.5 for unconstrained problems. However, it is most likely that real control allocation scenarios are nonconvex and non-continuous, multi-modal optimization problems. In such a case, the projection procedure used to evaluate u_a, u_b, or any other control subspace is restricted to the system‘s capability to deliver a given action. These real limits may generate disconnected feasible regions inside each subspace where reaching an inner limit is crucial for the convergence process. As previously demonstrated, the iterative FCA approach linearizes the solution space T^{ma} and T^{mb} around control actions. If a given

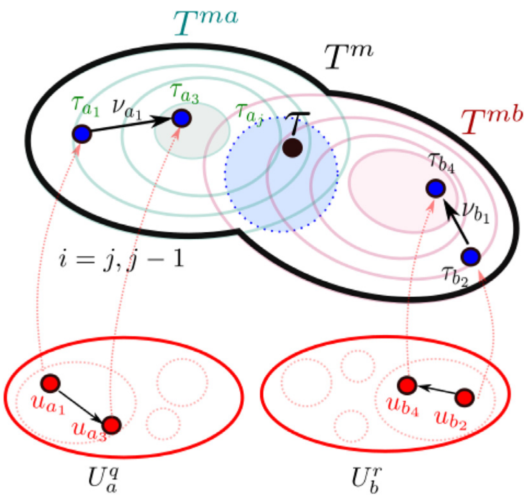


Fig. 5. Constrained convergence process scenario.

limit is reached at one iteration, the linearization point stabilizes and may not reach a minimum point, even that such a point exists in a different region. Fig. 5 shows this process where point u_{a_3} reaches an operations limit. In such a case, if the next iteration does not change the solution space T^{ma} leading the process into a different region, the convergence stops, and the system maybe not be solved. In the context of this work, this situation will be named as a hard constraint.

Although this could be seen as a drawback in the approach, it is possible to define strategies to solve scenarios like this:

- Correct design of each solution space set
  - The FCA is a procedure designed to enhance control allocation speed. It represents a systematic approach of decoupling nonlinearities and reducing the solution space. It does not mean that a given problem can be randomly broken. The forces and actions must be carefully selected in order to achieve convergence.
- Possibility of Dynamical changes over the solution sets
  - When the system reach a hard constraint, it is possible to improve the search space by adding the dimension related to the highest Lagrange multiplier.

All these scenarios will be demonstrated in detail at the performance evaluation section.

3.6. Proposed FCA algorithm

It is possible to develop different breaking strategies. For a general approach, the following heuristics rules were used:

a- Choose the real actions set in a way that enables the linearization of one set regarding the other transforming the nonlinear system into two overlapped linear ones;
b- The overlapping is when at least one virtual variable is present in both systems.
c- The constraints are enforced by an outside function after each single iteration.

Although this approach is not robust as Interior Point algorithms, it is fast and well established in literature [25].

To solve Eq. (15) by using Eq. (16) and (17), this work proposes the Algorithm 1, where each epoch represents a converged system and each *while* inner looping represents one iteration.

The upper-scripts $(^+)$ and $(^-)$ are the actual and previous FCA iteration values, respectively. The input arguments are the VCV set $\boldsymbol{\tau} = [\boldsymbol{\tau}_a, \boldsymbol{\tau}_b]$ from the system controllers, and the previous RAs $\boldsymbol{u}^- = [\boldsymbol{u}_a^-, \boldsymbol{u}_b^-]$. The output is the updated RA set $\boldsymbol{u}^+ = [\boldsymbol{u}_a^+, \boldsymbol{u}_b^+]$. The stopping criteria are the convergence error ε_{max}, the maximum number of iterations i_{max} and the amplitude of each variable $\boldsymbol{u}_j$ represented by $\Delta\boldsymbol{u}_j$ with maximum of Δu_{max}.

Algorithm 1 : FCA algorithm per epoch.

```
1:  function $\boldsymbol{u}^+$ = FDCA($\boldsymbol{\tau}, \boldsymbol{u}^-$)
2:      Set: $\boldsymbol{\tau}_a, \boldsymbol{\tau}_b, \boldsymbol{u}_a^-, \boldsymbol{u}_b^-, i_{max}, rep_{max}$
3:      Initialize: $i = 1$, $rep = 1$, $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4 = 10^3$
4:      while do$\varepsilon_1, \varepsilon_2, \varepsilon_3 \geq \varepsilon_{max}$, $\varepsilon_4 \geq \Delta u_{max}$, $i \leq i_{max}$ and $rep < rep_{max}$
5:          $\boldsymbol{u}_a^+ = \boldsymbol{M}_a^\dagger(\boldsymbol{u}_b^-)\boldsymbol{\tau}_a$
6:          Enforce: $\underline{\boldsymbol{u}}_a \leq \boldsymbol{u}_a^+ \leq \bar{\boldsymbol{u}}_a$
7:          $\varepsilon_1 = ||\boldsymbol{M}_a(\boldsymbol{u}_b^-)\boldsymbol{u}_a^+ - \boldsymbol{\tau}_a||$
8:          $\boldsymbol{u}_b^+ = \boldsymbol{M}_b^\dagger(\boldsymbol{u}_a^+)\boldsymbol{\tau}_b$
9:          Enforce: $\underline{\boldsymbol{u}}_b \leq \boldsymbol{u}_b^+ \leq \bar{\boldsymbol{u}}_b$
10:         $\varepsilon_2 = ||\boldsymbol{M}_b(\boldsymbol{u}_a^+)\boldsymbol{u}_b^+ - \boldsymbol{\tau}_b||$
11:         $\varepsilon_3 = ||\boldsymbol{M}(\boldsymbol{u}) - \boldsymbol{\tau}||$
12:         $\varepsilon_4 = \sum_{j=1}^n ||\boldsymbol{u}_j^- - \boldsymbol{u}_j^+||/\Delta u_j$
13:         i = i + 1
14:         $\boldsymbol{u}_a^- = \boldsymbol{u}_a^+$
15:         $\boldsymbol{u}_b^- = \boldsymbol{u}_b^+$
16:         if $i = imax$ then
17:             set $i = 1$, $rep = rep + 1$
18:             Set $d$ = dimension with highest error in $(\boldsymbol{M}(\boldsymbol{u}) - \boldsymbol{\tau})$
19:             if $not(d \in \tau_a)$ then
20:                 add $d \longrightarrow \tau_a$, update $\boldsymbol{M}_a$
21:             else
22:                 add $d \longrightarrow \tau_b$, update $\boldsymbol{M}_b$
23:             end if
24:         end if
25:     end while
26:     $\boldsymbol{u}^+ = [\boldsymbol{u}_a^+, \boldsymbol{u}_b^+]$
27:     return : $\boldsymbol{u}^+$
28: end function
```

Finally, the Algorithm 1 was formulated considering only 2 RASs and 2 VCV grouping sets, but it may be broken into more subsystems.

4. FCA performance evaluation

To present the FCA approach, consider the mathematical problem presented in Eq. (50). It is important to note that the focus of this simulation is to demonstrate the FCA mathematical performance using a framework easy to understand. Thus, the dynamics, control law and physical meaning will be disregarded, and only the convergence analysis considering several situations will be shown. However, as the control allocation problem depends on these characteristics, it will be considered that the virtual forces, that should be provided by the low level control system, will be given by Eq. (51). Moreover, to enrich the simulations, the control allocation matrix was designed to emulate an over-actuated unbalanced system. This system is shown in Eq. (50)

$$\hat{\boldsymbol{\tau}} = \begin{bmatrix} \hat{\boldsymbol{\tau}}_1 \\ \hat{\boldsymbol{\tau}}_2 \\ \hat{\boldsymbol{\tau}}_3 \end{bmatrix} = \boldsymbol{M}(\boldsymbol{u}) = \begin{bmatrix} 8x_1\cos(\gamma_1) - 3x_2\cos(\gamma_2) \\ 3x_1\cos(\gamma_1) + 6x_2\cos(\gamma_2) \\ -x_2\cos(\gamma_1) - 3x_1\cos(\gamma_2) \end{bmatrix} \tag{50}$$

where $\hat{\boldsymbol{\tau}}$ is a set of virtual forces provided by an imaginary control law, $\boldsymbol{M}(\boldsymbol{u})$ is the control allocation matrix that, for a real system, would be designed by considering the system's physical characteristics.

Although this process has no physical meaning, it is similar to a 3 degree of freedom (DoF) Autonomous Surface Vessel (ASV) [17,26] with two-directional thrusters. This configuration would provide forces along with surge (forward), sway (sideways), and yaw. In the system represented by Eq. (50), these forces are taken as the VCVs and are represented by $[\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \boldsymbol{\tau}_3]$ and the real actions are forces generated by x_1 and x_2 along their respective direction γ_1 and γ_2. Again, this is just an illustration with the purpose of better visualizing the relationship between VCVs and RA, and it does not represent the control allocation problem of a real ASV.

Therefore the virtual signals $\boldsymbol{\tau}$, which in a real process would be evaluated by a closed-loop controller, accordingly to Fig. 1 are given by the following open-loop set of functions:

$$\boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \\ \boldsymbol{\tau}_3 \end{bmatrix} = \begin{bmatrix} 0.1 + 0.1\sin(\pi t/40) \\ 0.05 + 0.1\sin(\pi t/30) \\ -0.3 + 0.2\sin(\pi t/20) \end{bmatrix} \tag{51}$$

The goal is, given Eq. (50), to define and evaluate the set of real actions $\boldsymbol{u}$ where $\left|\hat{\boldsymbol{\tau}} - \boldsymbol{\tau}\right|$ is minimal for any realizable set $\boldsymbol{\tau}$. To apply the FCA it is first necessary to break the nonlinear system represented by the right-hand matrix in Eq. (50) into two linear systems. Analyzing Eq. (50), it is possible to see that the nonlinearities lies on the trigonometrical function $\cos(\gamma_i)$ and its multiplication by x_i. To break these nonlinearities it is necessary to consider $y_1 = \cos(\gamma_1)$, $y_1 = \cos(\gamma_1)$ and separate this new set (y_1, y_2) from (x_1, x_2). While the first strategy will generate the system shown in Eq. (52),

$$\boldsymbol{M}(\boldsymbol{u}) = \begin{bmatrix} 8x_1 \cdot y_1 - 3x_2 \cdot y_2 \\ 3x_1 \cdot y_1 + 6x_2 \cdot y_2 \\ -x_2 \cdot y_1 - 3x_1 \cdot y_2 \end{bmatrix} \tag{52}$$

to divide the variables into the sets $u_b = (y_1, y_2)$ and $u_a = (x_1, x_2)$ generates two independent linear systems as shown in Eqs. (53) and (54)

$$\boldsymbol{M_a}(\boldsymbol{u_b}) = \begin{bmatrix} 8x_1 - 3x_2 \\ 3x_1 + 6x_2 \\ -x_2 - 3x_1 \end{bmatrix} \tag{53}$$

$$\boldsymbol{M_b}(\boldsymbol{u_a}) = \begin{bmatrix} 8y_1 - 3y_2 \\ 3y_1 + 6y_2 \\ -y_1 - 3y_2 \end{bmatrix} \tag{54}$$

Thus, it is considered:

$$\boldsymbol{u} = [\cos(\gamma_1), \cos(\gamma_2), x_1, x_2]^T \tag{55}$$

subject to:

$$[-1, -1, -\infty, -\infty]^T \leq \boldsymbol{u} \leq [1, 1, \infty, \infty]^T \tag{56}$$

As the system will threat $\cos(\gamma_1)$ and $\cos(\gamma_2)$ as any ordinary variable, it is necessary to define its valid limits. Moreover, for this first example, x_1 and x_2 will be considered unbounded.

The vector $\boldsymbol{u}$ can now be broken to isolate the nonlinearities:

$$\boldsymbol{u}_a = [\cos(\gamma_1), \cos(\gamma_2)]^T \tag{57}$$
$$\boldsymbol{u}_b = [x_1, x_2]^T \tag{58}$$

The next step is to define the subsystems M_a and M_b by choosing which virtual variable will be evaluated at each subsystem according to

$$\hat{\boldsymbol{\tau}}_a = \boldsymbol{M_a}(\boldsymbol{u_b})\boldsymbol{u_a}$$
$$\hat{\boldsymbol{\tau}}_b = \boldsymbol{M_b}(\boldsymbol{u_a})\boldsymbol{u_b}$$

There are several combinations for that and the next subsections will evaluate the performance of other possible configurations. In all cases, let the maximum number of iterations $i_{max} = 200$, maximum squared error $\boldsymbol{\varepsilon}_{max} = 1 \times 10^{-20}$, and

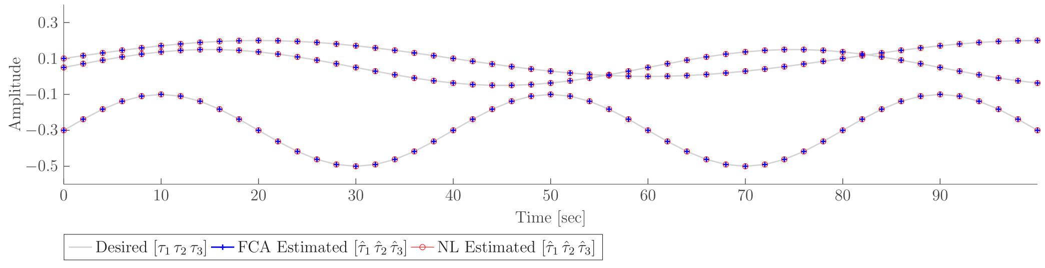


Fig. 6. Desired $\boldsymbol{\tau}$ and estimated $\hat{\boldsymbol{\tau}}$.

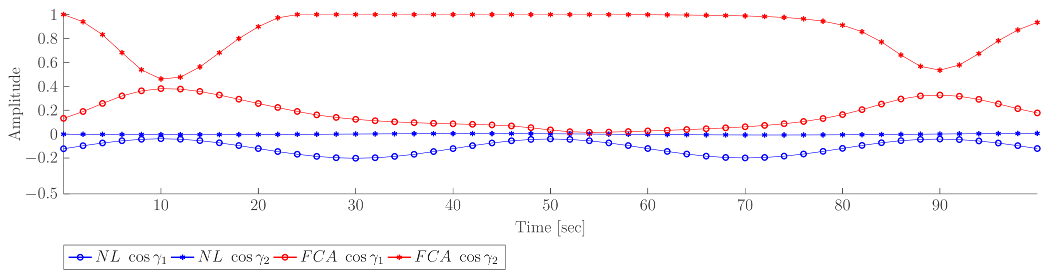


Fig. 7. Values of $\cos(\gamma_1)$ and $\cos(\gamma_2)$ estimated by FCA and *fmincon*.

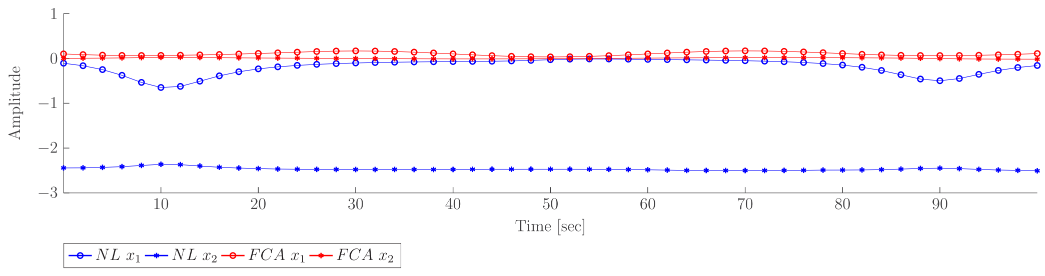


Fig. 8. Values of x_1 and x_2 estimated by FCA and *fmincon* f.

$\Delta u_{max} = 1 \times 10^{-3}$ to be the three convergence criteria. The RA initial conditions are $\boldsymbol{u} = [0.5, 0.5, 0.5, 0.5]$, and for each change in a desired set of τs, both optimization algorithms will start with the last evaluated $\boldsymbol{u}$.

In other to compare the results, the Matlab *fmincon* (nonlinear constrained optimization function toolbox) interior-point algorithm was used.

4.1. Fully coupled virtual variables evaluation performance

This configuration consider that all virtual variables will be present on both systems, i.e.

$$\hat{\boldsymbol{\tau}}_a(x_1, x_2) = \hat{\boldsymbol{\tau}}_b(\cos(\gamma_1), \cos(\gamma_2)) = [\hat{\boldsymbol{\tau}}_1, \hat{\boldsymbol{\tau}}_2, \hat{\boldsymbol{\tau}}_3]^T \tag{59}$$

which gives the following subsystems:

$$\begin{bmatrix} \hat{\boldsymbol{\tau}}_1 \\ \hat{\boldsymbol{\tau}}_2 \\ \hat{\boldsymbol{\tau}}_3 \end{bmatrix} = \begin{bmatrix} 8\cos(\gamma_1) & -3\cos(\gamma_2) \\ 3\cos(\gamma_1) & 6\cos(\gamma_2) \\ -3\cos(\gamma_2) & -\cos(\gamma_1) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{60}$$

$$\begin{bmatrix} \hat{\boldsymbol{\tau}}_1 \\ \hat{\boldsymbol{\tau}}_2 \\ \hat{\boldsymbol{\tau}}_3 \end{bmatrix} = \begin{bmatrix} 8x_1 & -3x_2 \\ 3x_1 & 6x_2 \\ -x_2 & -3x_1 \end{bmatrix} \begin{bmatrix} \cos(\gamma_1) \\ \cos(\gamma_2) \end{bmatrix} \tag{61}$$

These systems are solved interactively by using Eqs. (62) and (63), according to Algorithm 1.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \boldsymbol{M}_a^{\dagger}(\cos(\gamma_1), \cos(\gamma_2)) \begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \\ \boldsymbol{\tau}_3 \end{bmatrix} \tag{62}$$

$$\begin{bmatrix} \cos(\gamma_1) \\ \cos(\gamma_2) \end{bmatrix} = \boldsymbol{M}_b^{\dagger}(x_1, x_2) \begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \\ \boldsymbol{\tau}_3 \end{bmatrix} \tag{63}$$

Fig. 6 shows the desired $\boldsymbol{\tau}$ (gray lines) provided by Eq. (51) and the estimated by the FCA (blue crosses) and *fmincon* (red circles).

It is possible to see that both methods have been able to find all solutions. However, as the system has several possible solutions, each methodology has presented a different output as it can be seen in Figs. 7 and 8. An important observation is that the FCA had to enforce the upper limit over $\cos(\gamma_2)$ and, even so, it did not have any impact over its performance.

Considering the squared error and convergence time as performance metrics, it is possible to see by analyzing Figs. 9 and 10 that the FCA was able to achieve the desired precision, i.e., 10^{-20} while the nonlinear Interior-Point (IP) algorithm was not. Moreover, the FCA was, in average, 130 times faster than the IP approach. Comparing with other nonlinear algorithms, the FCA was 10 times faster than the Active-Set (AS) [27], and 15 times faster than the Sequential Quadratic Programming (SQP) approach [28]. Also, the minimum precision error was 10^{-11} and 10^{-12} for the AS and SQP, respectively. Note that Fig. 9 shows the final result of the FCA approach for each setpoint in $t = [0, 100]s$. The individual convergence of the proposed approach is shown in Fig. 11, where in the case of the same solution space, it is possible to ensure the monotonic convergence demonstrated in Section 3.2.

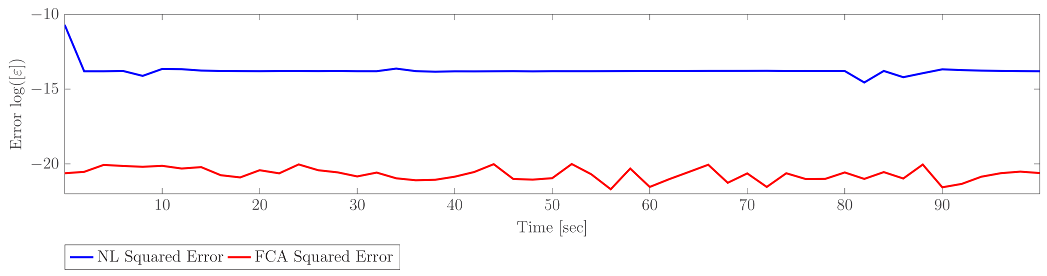


Fig. 9. Squared error of the simulation.

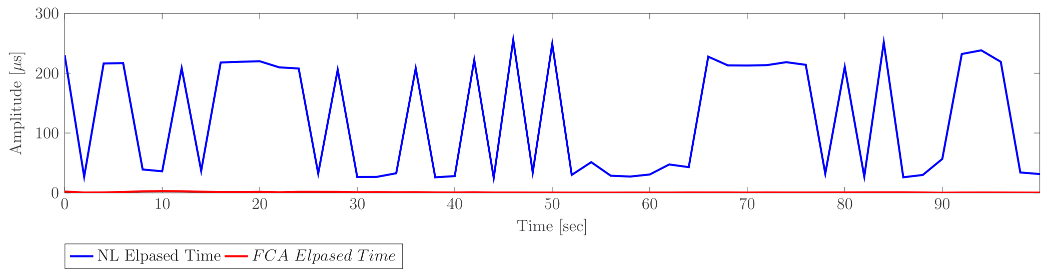


Fig. 10. Convergence time (μs).

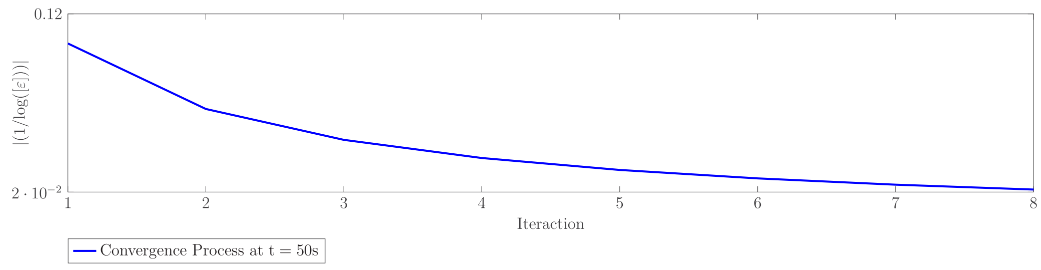


Fig. 11. Single iteration evaluation performance at $t = 50$ s.

4.2. Changing the subsystems solution order

The control allocation problem presented at Eq. (50) was solved by breaking the system into Eq. (62) and (63). Changing the subsystems order at algorithm, i.e., solving Eq. (63) first than Eq. (62), also changes the quantitative results. It means that the execution time and convergence precision were the same, but the resulting values for $(x_1, x_2, \cos(\gamma_1), \cos(\gamma_2))$ were different as shown in Figs. 12 and 13. Also, the new solution order is demonstrated in blue and named as FCA 2. The original approach is shown in red and named as FCA 1. As the system solution order has changed, the search direction into the solution space also changes. Considering that a nonlinear system have several local minima, changes in the search and initialization may reach different solutions.

4.3. Sensitivity to the starting point

To evaluate the sensitivity to the starting point, the value of

$$\tau = [0.0293, -0.0366, -0.1000]^T$$

corresponding to $t = 50$ s was arbitrarily selected. A total of 100 simulations were performed with starting points randomly choose by using

$$\underline{\boldsymbol{u}} = [-2, -2, -1, -1]^T \text{ and, } \overline{\boldsymbol{u}} = [2, 2, 1, 1]$$

as the lower and upper limits, respectively. Figs. 14 and 15 show the initialization point in blue and the results provide by the FCA in red lines. It is possible to see that all simulations have converged, demonstrating robustness to the initialization point.

4.4. Preliminary contingency analysis

Considering the tutorial case system, but now supposing that the variable γ_1 represents a malfunctioning servomotor with fixed position in 0 degrees. In such case, $\cos(\gamma_1) = 1$ is fixed during all simulation. Figs. 16 and 17 show the variables $(\cos(\gamma_1), \cos(\gamma_2))$ and (x_1, x_2) respectively. An interesting result is, considering this fault, both FCA and the nonlinear approach have achieved exactly the same solutions.

4.5. Sub-coupled systems

Although all simulation scenarios have considered just two fully coupled sub-systems so far, meaning that all virtual variables are completely overlapped, it is possible to have different topologies. Table 1 shows the results of several possible scenarios, where u_a, u_b, M_a and M_b are the same ones defined in Eqs. (60) and (61). The notation $M_k^{i,j,..}$ represents the matrix formed by lines $\{i, j, ..\}$ of M_k. Finally these results do not consider the dynamical changes over the solution space. This will be demonstrated further.

The first important observation is that cases 2, 7 and 9 are the only ones that did not converge. Furthermore, they are also the only ones where the virtual control variable τ_3 are not presented in both subsystems M_a and M_b. All other cases, including case 6 where the only superposition is over τ_3, have converged well. Note that case 8 breaks M_b into two different systems demonstrating that is possible to have several configurations. Finally,

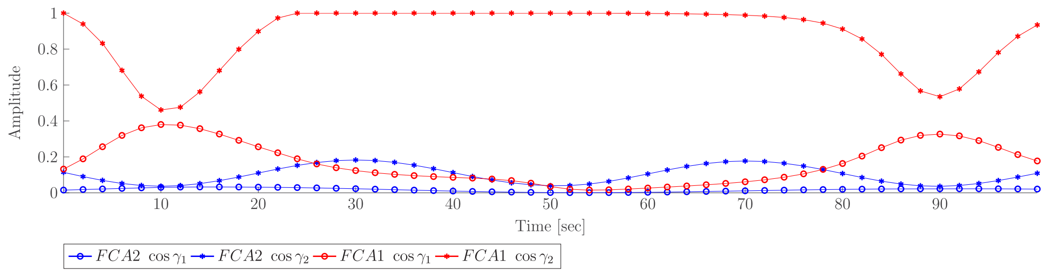


Fig. 12. Values of $\cos(\gamma_1)$ and $\cos(\gamma_2)$ estimated by FCA 1 and FCA 2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

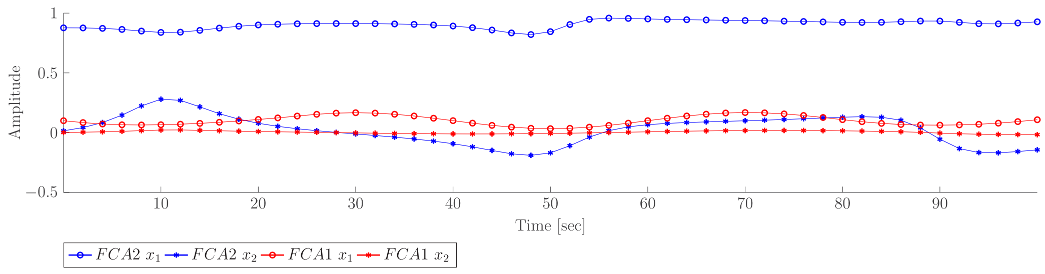


Fig. 13. Values of x_1 and x_2 estimated by FCA 1 and FCA 2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

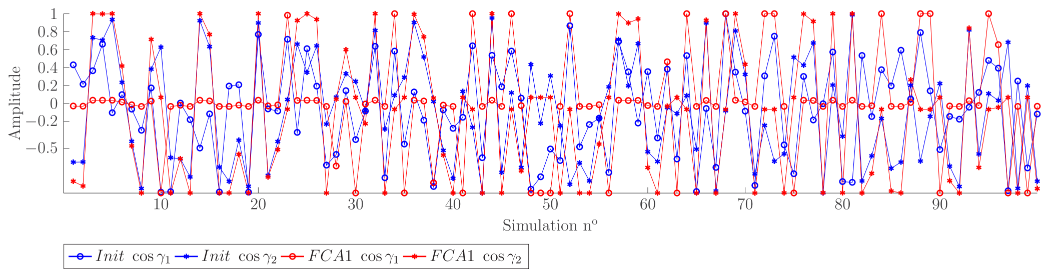


Fig. 14. Values of $\cos(\gamma_1)$ and $\cos(\gamma_2)$ initialized (blue) and estimated (red) by FCA. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

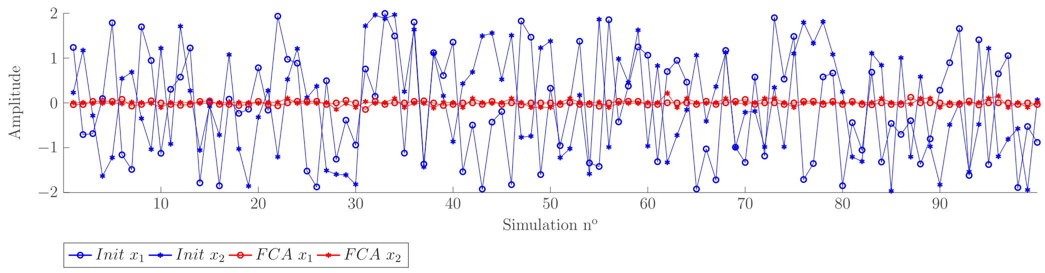


Fig. 15. Values of x_1 and x_2 initialized (blue) and estimated (red) by FCA. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

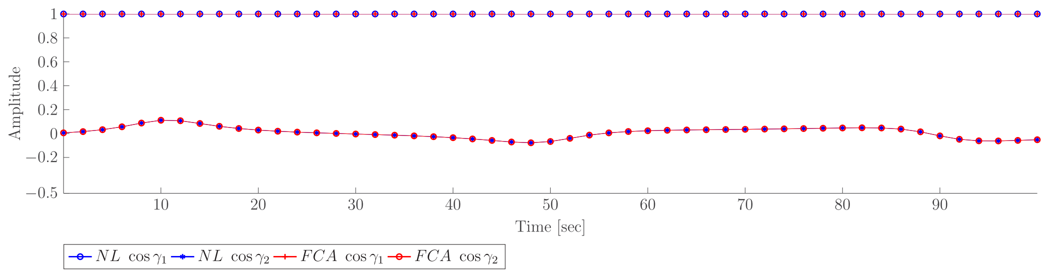


Fig. 16. Values of $\cos(\gamma_1)$ and $\cos(\gamma_2)$ estimated by FCA and *fmincon*.

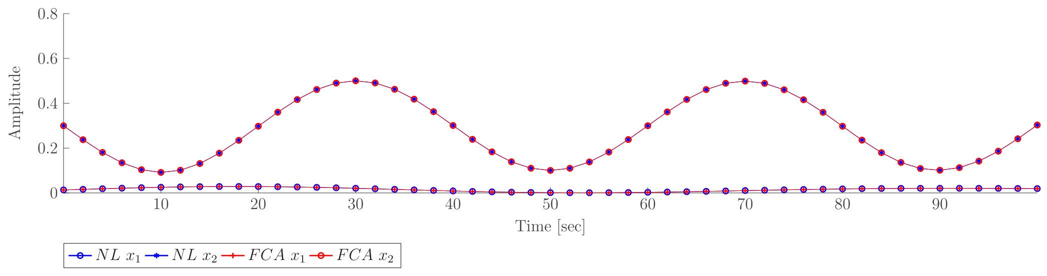


Fig. 17. Values of x_1 and x_2 estimated by FCA and *fmincon*.

Table 1
Different possible topologies.

Case	System	Max. Time (μs)	Max. $\varepsilon_3(\times 10^{-20})$
1	$[\hat{\tau}_1, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{1,3}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_1, \hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M_b}(\boldsymbol{u_a})\boldsymbol{u_b}$	140	0.945
2	$[\hat{\tau}_1, \hat{\tau}_2]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{1,2}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_1, \hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M_b}(\boldsymbol{u_a})\boldsymbol{u_b}$	640	0.092×10^{20}
3	$[\hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{2,3}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_1, \hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M_b}(\boldsymbol{u_a})\boldsymbol{u_b}$	180	0.971
4	$[\hat{\tau}_1, \hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M_a}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_1, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{b}}^{\mathbf{1,3}}(\boldsymbol{u_a})\boldsymbol{u_b}$	100	0.954
5	$[\hat{\tau}_1, \hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M_a}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{b}}^{\mathbf{2,3}}(\boldsymbol{u_a})\boldsymbol{u_b}$	160	0.985
6	$[\hat{\tau}_1, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{1,3}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{b}}^{\mathbf{2,3}}(\boldsymbol{u_a})\boldsymbol{u_b}$	120	0.915
7	$[\hat{\tau}_1, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{1,3}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_1, \hat{\tau}_2]^T = \boldsymbol{M}_{\boldsymbol{b}}^{\mathbf{1,2}}(\boldsymbol{u_a})\boldsymbol{u_b}$	510	0.122×10^{20}
8	$[\hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{2,3}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_1]^T = \boldsymbol{M}_{\boldsymbol{b}}^{\mathbf{1}}(\boldsymbol{u_a})\boldsymbol{u_b}$ $[\hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{c}}^{\mathbf{2,3}}(\boldsymbol{u_a})\boldsymbol{u_b}$	170	0.932
9	$[\hat{\tau}_1, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{1,3}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_2]^T = \boldsymbol{M}_{\boldsymbol{b}}^{\mathbf{2}}(\boldsymbol{u_a})\boldsymbol{u_b}$	550	0.522×10^{20}

Table 2
Cases 2, 7 and 9 with the dynamical allocation approach.

Case	System	Max. Time (μs)	Max. $\varepsilon_3(\times 10^{-20})$
2	$[\hat{\tau}_1, \hat{\tau}_2]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{1,2}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_1, \hat{\tau}_2, \hat{\tau}_3]^T = \boldsymbol{M_b}(\boldsymbol{u_a})\boldsymbol{u_b}$	210	0.943
7	$[\hat{\tau}_1, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{1,3}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_1, \hat{\tau}_2]^T = \boldsymbol{M}_{\boldsymbol{b}}^{\mathbf{1,2}}(\boldsymbol{u_a})\boldsymbol{u_b}$	190	0.988
9	$[\hat{\tau}_1, \hat{\tau}_3]^T = \boldsymbol{M}_{\boldsymbol{a}}^{\mathbf{1,3}}(\boldsymbol{u_b})\boldsymbol{u_a}$ $[\hat{\tau}_2]^T = \boldsymbol{M}_{\boldsymbol{b}}^{\mathbf{2}}(\boldsymbol{u_a})\boldsymbol{u_b}$	188	0.937

case 9 shows a fully decoupled system, where there is no superposition between τ_a and τ_b. Other possible decoupled systems were also tested and all did not converge. This demonstrates that superposition is an important characteristic of this approach and must be carefully analyzed.

The tutorial case provided by Eqs. (60) and (61) had a maximum time of 240 μs and maximum error of 0.066. Thus, reducing the system's order may improve computational performance. However, the convergence depends on how the virtual variables are separated among the sub-systems.

To better understand the problem, a deeper analysis of cases 2, 7, and 9 will be provided. Take, for instance, the convergence process of case 2 as shown in Fig. 18 and the output values corresponding to the angles and modules from the toy system as shown in Figs. 19 and 20. It is possible to note that most non-converged iterations have occurred when the angles have hit their upper or lower limits, more specifically $cos(\gamma_1)$. As the set of an equation, M_a does not have the τ_c variable. They do not help drive the system towards the desired operating condition.

To demonstrate this condition, the first action is to remove the trigonometrical limits $[-1, 1]$. The system converges with a max error of 0.910×10^{-20} and the result related to the angles is shown in Fig. 21, which corroborates the theory.

Furthermore, the next test will apply the trigonometrical limits and the dynamical changes over the solution sets. For case 2, the variable τ_3 is always responsible for the bad performance. Adding this variable to the first set of the equation just when the process does not converge, the system now presents a fast convergence process.

The results for this strategy used in cases 2, 7, and 9 are shown in Table 2.

5. Over-actuated system application

For a better evaluation, this section will apply the FCA in a Quadrotor Tilt-Rotor (QTR) over-actuated system. Fig. 22 shows some details of the servomotor mounting design, as much as its VCVs:

The aircraft has 4 propulsion sets. Each one has a brushless and servomotor unit to provide thrust and controlling the tilting angles, respectively. Servomotor tilting angles have their positive/negative directions according to Fig. 22(b).

Therefore, the real actions $\boldsymbol{u} = [\delta_1, \delta_2, \delta_3, \delta_4, \gamma_1, \gamma_2, \gamma_3, \gamma_4]$ that control the QTR are the speeds of the motors and tilt angles provided by the 4 motors and 4 servos, respectively.

5.1. QTR dynamics and kinematics modeling

Whereas roll, pitch and yaw angles, as well angular velocities are measured in the Inertial Frame $\mathcal{F}^I$, linear accelerations are represented in Body-Fixed Frame $\mathcal{F}^b$. Vector $\boldsymbol{\eta_1} = [p_n, p_e, -h]^T \in \mathbb{R}^3$ represents the inertial North, East and Altitude (facing down) coordinates of the aircraft center of gravity along the $(\hat{i}^I, \hat{j}^I, \hat{k}^I)$ axes represented in the inertial frame; vector $\boldsymbol{\eta_2} = [\phi, \theta, \psi]^T \in \mathbb{R}^3$ represents the roll, pitch and yaw angles considering the vehicle frame $(\hat{i}^v, \hat{j}^v, \hat{k}^v)$; vectors $\boldsymbol{\nu_1} = [u, \upsilon, \omega]^T \in \mathbb{R}^3$ and $\boldsymbol{\nu_2} = [p, q, r]^T \in \mathbb{R}^3$ represents the three dimensional linear speeds and angular velocities over the axes $(\hat{i}^b, \hat{j}^b, \hat{k}^b)$ over frame $\mathcal{F}^b$.

The default nomenclature is presented from Eqs. (64) to (69), according to [29]:

$$\boldsymbol{\eta_1} = [p_n \quad p_e \quad -h]^T \tag{64}$$

$$\boldsymbol{\eta_2} = [\phi \quad \theta \quad \psi]^T \tag{65}$$

$$\boldsymbol{\eta} = [\boldsymbol{\eta_1} \quad \boldsymbol{\eta_2}]^T \tag{66}$$

$$\boldsymbol{\nu_1} = [\boldsymbol{v^b}] = [u \quad v \quad w]^T \tag{67}$$

$$\boldsymbol{\nu_2} = [\boldsymbol{\omega^b}] = [p \quad q \quad r]^T \tag{68}$$

$$\boldsymbol{\nu} = [\boldsymbol{\nu_1} \quad \boldsymbol{\nu_2}]^T \tag{69}$$

The 6 DoFs rigid body kinematics model is expressed in Eq. (70).

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{70}$$

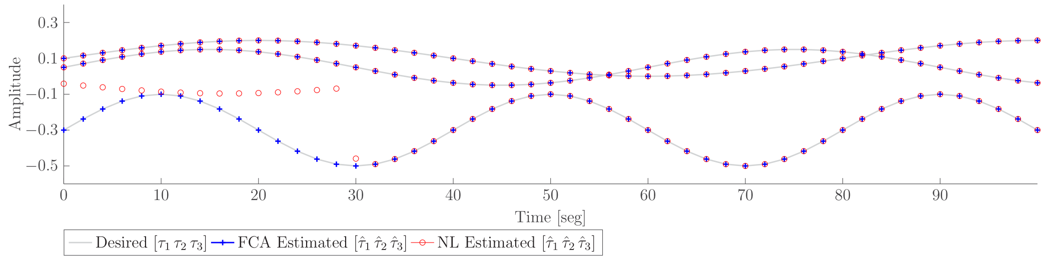


Fig. 18. Desired $\boldsymbol{\tau}$ and estimated $\hat{\boldsymbol{\tau}}$ for case 2.

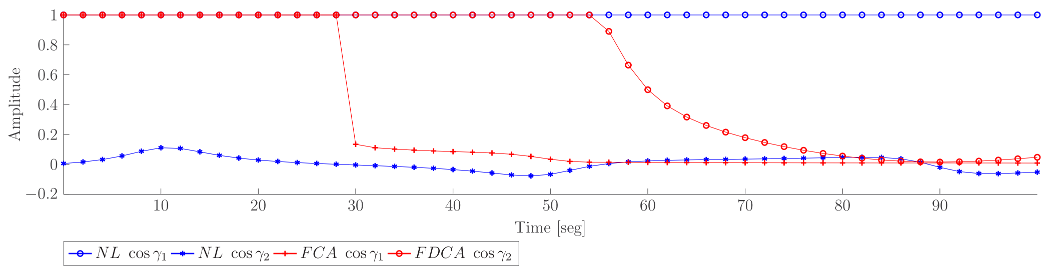


Fig. 19. Constrained angles for case 2.

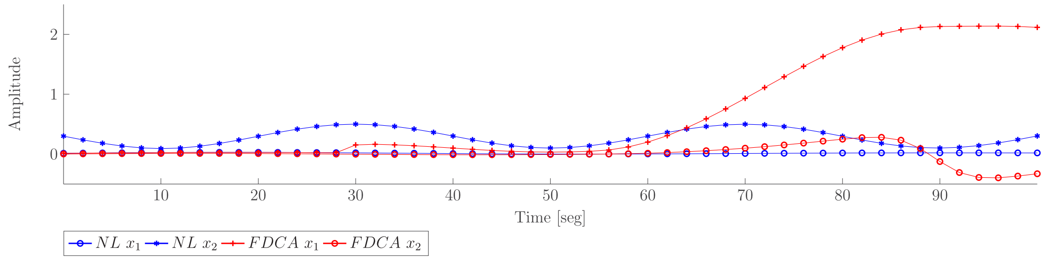


Fig. 20. Resulting forces for case 2.

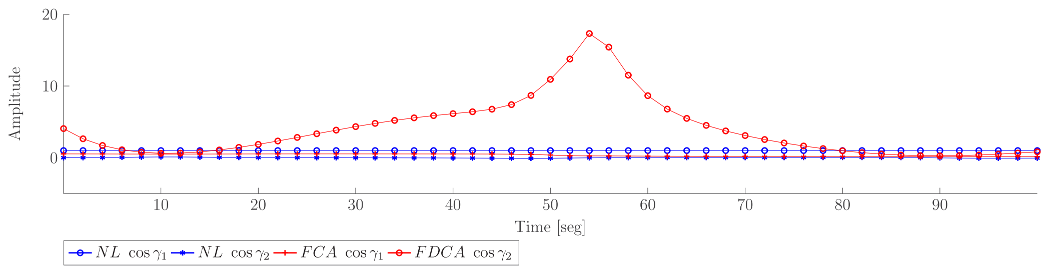


Fig. 21. Resulting angles for case 2 without limits.

where $\boldsymbol{\nu} \in \mathbb{R}^6$ is the generalized velocity vector in $\mathcal{F}^b$ and $\dot{\boldsymbol{\eta}} \in \mathbb{R}^6$ is the velocity vector in $\mathcal{F}^I$. $\boldsymbol{J}(\boldsymbol{\eta}) \in \mathbb{R}^{6\times 6}$ is the transformation matrix.

UAV dynamics model is described by differential equations from Newton–Euler method, such as shown in Eq. (71).

$$\boldsymbol{I^b}\dot{\boldsymbol{\nu}} + \boldsymbol{C^b}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{\tau^b_a} - \boldsymbol{\tau^b_g} = \underbrace{\boldsymbol{\tau^b_p}}_{\boldsymbol{\tau}} \tag{71}$$

where $\boldsymbol{I^b} \in \mathbb{R}^{6\times 6}$ is the system inertia matrix, $\boldsymbol{C^b}(\boldsymbol{\nu}) \in \mathbb{R}^{6\times 6}$ is the Coriolis–centripetal matrix in frame $\mathcal{F}^b$, $\boldsymbol{\tau^b_a}$, $\boldsymbol{\tau^b_g}$, $\boldsymbol{\tau^b_p} \in \mathbb{R}^6$ are the Aerodynamics, Gravitational and Propulsion resultant vectors, respectively, compounded by forces and torques at frame $\mathcal{F}^b$.

Considering that the real actions are $\boldsymbol{u} = [\delta_1, \delta_2, \delta_3, \delta_4, \gamma_1, \gamma_2, \gamma_3, \gamma_4]$, each set of motor rotation speed δ_i and servomotor tilt angle γ_i has a individual impact over the virtual control variables $\hat{\boldsymbol{\tau}} = [X, Z, L, M, N]^T$ such as

$$X_i = k_1\delta_i \sin\gamma_i \tag{72}$$

$$Z_i = k_1\delta_i \cos\gamma_i \tag{73}$$

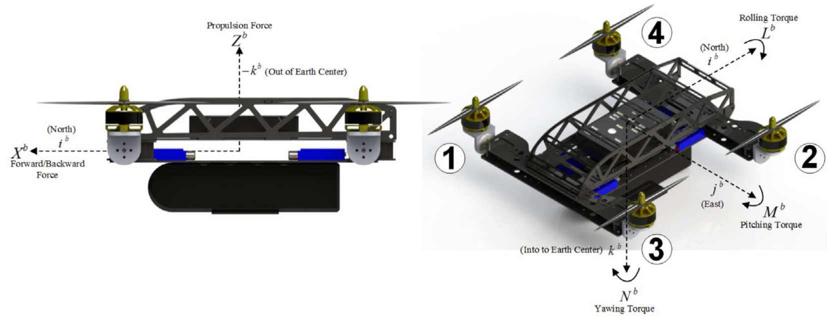
$$L_i = (\pm k_1 d_{\llcorner} \cos\gamma_i \pm k_2 \sin\gamma_i)\delta_i \tag{74}$$

$$M_i = k_1 d_{\llcorner} \cos\gamma_i\delta_i \tag{75}$$

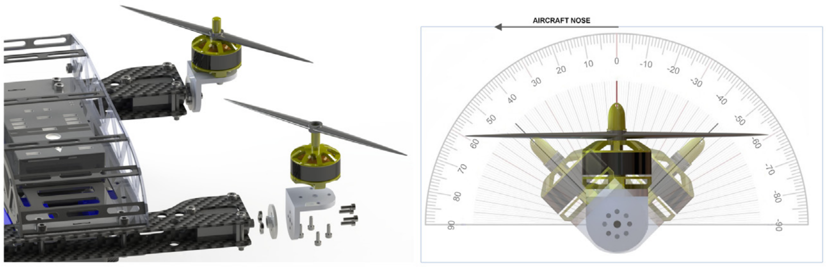
$$N_i = (\pm k_1 d_{\llcorner} \sin\gamma_i \pm k_2 \cos\gamma_i)\delta_i \tag{76}$$

where (X_i, Z_i) and (L_i, M_i, N_i) are the impact of propulsion set i over the forces and torques on the body frame axes $(\hat{\boldsymbol{i}}^b, \hat{\boldsymbol{k}}^b)$ and $(\hat{\boldsymbol{i}}^b, \hat{\boldsymbol{j}}^b, \hat{\boldsymbol{k}}^b)$, respectively. The signal $\pm$ depends on the placement of each individual set at the vehicle, as shown in Fig. 22(a). Finally, $d_{\llcorner} = l\frac{\sqrt{2}}{2}$, l is the QTR arm length, and (k_1, k_2) are constants related to propulsion force and torque, respectively.

Therefore the CEM matrix given in Box I where $c\gamma_i = \cos(\gamma_i)$ and $s\gamma_i = \sin(\gamma_i)$, maps the real actions u to the virtual control



(a) Explanation of VCVs in the Body-Fixed frame.



(b) Exploded view of a propulsion set formed by a servo and a brushless motors (γ).

Fig. 22. Illustration of the tilt-rotor UAV.

$$\boldsymbol{M}(\boldsymbol{u}) = \begin{bmatrix} k_1\delta_1 s\gamma_1 + k_1\delta_2 s\gamma_2 + k_1\delta_3 s\gamma_3 + k_1\delta_4 s\gamma_4 \\ k_1\delta_1 c\gamma_1 + k_1\delta_2 c\gamma_2 + k_1\delta_3 c\gamma_3 + k_1\delta_4 c\gamma_4 \\ (-k_1 d_{\llcorner} c\gamma_1 - k_2 s\gamma_1)\delta_1 + (k_1 d_{\llcorner} c\gamma_2 - k_2 s\gamma_2)\delta_2 + (k_1 d_{\llcorner} c\gamma_3 + k_2 s\gamma_3)\delta_3 + (-k_1 d_{\llcorner} c\gamma_4 + k_2 s\gamma_4)\delta_4 \\ +k_1 d_{\llcorner} c\gamma_1\delta_1 - k_1 d_{\llcorner} c\gamma_2\delta_2 + k_1 d_{\llcorner} c\gamma_3\delta_3 - k_1 d_{\llcorner} c\gamma_4\delta_4 \\ (-k_1 d_{\llcorner} s\gamma_1 + k_2 c\gamma_1)\,\delta_1 + (k_1 d_{\llcorner} s\gamma_2 + k_2 c\gamma_2)\delta_2 + (k_1 d_{\llcorner} s\gamma_3 - k_2 c\gamma_3)\delta_3 + (-k_1 d_{\llcorner} s\gamma_4 - k_2 c\gamma_4)\delta_4 \end{bmatrix} \quad (77)$$

Box I.

variables set τ. Comparing this formulation with traditional UAV models, the detailed QTR has one more DoF (X_p^b) and four more real control actions ($\gamma_1, \gamma_2, \gamma_3, \gamma_4$). Therefore the system has five virtual actions defined by $\hat{\boldsymbol{\tau}} = [X, Z, L, M, N]^T$ which are the forces that acts over the vehicle and eight real actuators (4 PMW motors and 4 directional servos). The actions taken by the real actuators will provide the forces that drive the UAV. A different model of overactuated system with directional propulsion system can be seen in [17,26].

5.2. Remarks about the control techniques

This work does not focus or demonstrate the control loop specifications. Furthermore, it is used the Successive Loop Closure as presented in [30], such as other techniques provided by similar results [31,32].

5.3. Application of the FCA in the QTR

The QTR UAV used in this work is an over-actuated system with five controllable DoFs (roll, pitch and yaw angles, altitude and the forward/backward velocity) and eight actuators (four propulsion motors and four servomotors).

Analyzing the FCA approach presented at Section 3, Eqs. (77) represents the nonlinear system $\hat{\boldsymbol{\tau}} = \boldsymbol{M}(\boldsymbol{u})$ in Eq. (15). Therefore, the goal is to break (77) into two linear systems such as in Eq. (16) and (17). The first step is to define two sets of real control variables $\boldsymbol{u_a}$ and $\boldsymbol{u_b}$, capable of breaking the nonlinearities. For the QTR particular problem, it is considered the trigonometrical *cos* axes perpendicular to the aircraft and facing up, and the *sin* axes pointing to the aircraft's nose, as shown in Fig. 22(b). It is also considered that the maximum tilt angle for each servo is ±45 degrees, limiting the range to the 1st and 3rd trigonometrical quadrants. In such situation the *sin* operation is enough to determine the desired position. Furthermore, by separating the actuators by types, it generates two sets of control variables; one containing motors and another the servomotors, as shown next:

$$\boldsymbol{u}'_a \in \mathbb{R}^5 = [\sin(\gamma_1), \sin(\gamma_2), \sin(\gamma_3), \sin(\gamma_4), 1]^T \quad (78)$$

$$\boldsymbol{u_b} \in \mathbb{R}^4 = [\delta_1, \delta_2, \delta_3, \delta_4]^T \quad (79)$$

Note that the ordinal 1 in Eq. (78) is responsible for summing all terms related to cos. In that way, all related nonlinearities are treated as inner constants. One advantage of this strategy is it works as a normalization factor: as $\boldsymbol{u_a}$ is evaluated by the system, it is possible a returned value other than 1. In such case the full vector $\boldsymbol{u}'_a$ is normalized to keep it 1. After this normalization, the real signals $\boldsymbol{u}_a$ used to feed the servomotors are evaluated by:

$$\boldsymbol{u}_a \in \mathbb{R}^4 = [\sin^{-}(\gamma_1), \sin^{-}(\gamma_2), \sin^{-}(\gamma_3), \sin^{-}(\gamma_4)]^T \quad (80)$$

The next step in designing the system is to elect two sets of virtual variables $\boldsymbol{\tau_a}$ and $\boldsymbol{\tau_b}$. Considering the analysis provided by the tutorial case section, it is possible to have several configurations with the best choice being highly dependent on the

Table 3
Setpoints specifications for the simulation results.

Time (sec)	0	2.5	4	7	10
$\boldsymbol{h}$ (m)	10	10	10	10	10
$\boldsymbol{V_x}$ (m/s)	0	1	1	1	1
$\boldsymbol{\psi}$ (deg)	0	0	90	90	90
$\boldsymbol{\theta}$ (deg)	0	0	0	−5	0
$\boldsymbol{\phi}$ (deg)	0	0	0	10	0

Table 4
ISE of the methods.

ISE	FCA 1	FCA 2	NL 1	NL 2
h	106.94	108.89	107.01	109.74
V_x	39.51	39.52	36.15	36.74
ψ	18.96	19.29	19.43	19.44
θ	37.73	37.79	58.49	65.38
ϕ	39.08	39.34	37.17	42.12
Total	**242.22**	244.86	258.25	273.42

system. In this work, three configurations are analyzed. The first one separates the virtual actions by heuristically associating the real control variables to its most impacting virtual partners: the servomotors have more effective impact over the horizontal velocity and yaw, while the motors impact all variables together. This approach has the benefit of having all variables, including the most critical ones (roll, pitch and altitude), controlled by the actuators with the fastest responses, but also has the servomotors helping to control variables responsible for better performance.

Finally, it uses the concept $\boldsymbol{\tau_a} \subset \boldsymbol{\tau_b}$, which although may not be the fastest, but it is the more reliable according to Section 4. Although this seems to be the best strategy, a second scenario is tested, where the horizontal velocity in $\boldsymbol{\tau_b}$ is not superposed.

For both simulations, the RA initial conditions are zero, with the drone at rest and facing north. The stopping criteria are the maximum number iterations $i_{max} = 50$, maximum squared error $\boldsymbol{\varepsilon}_{max} = 1 \times 10^{-6}$, and $\Delta u_{max} = 1 \times 10^{-3}$. For the system feasible constrained solutions, normalized values were considered for propulsion motor rotations $[0, 1] = [0, 100\%]$, and for servomotor tilting angles $[-1, +1] = [-45, +45]$ degrees.

For all experiments, the setpoints shown in Table 3 are used in ramp, except altitude dynamics.

For a more precise comparison, the Integral of the Squared Error (ISE) is adopted:

$$ISE_v = \sum_{i=0}^{i=N} \sqrt{(\boldsymbol{v}_i^s - \boldsymbol{v}_i^m)^2} \tag{81}$$

where N is the total number of iterations, $\boldsymbol{v}^s$ is a variable representing the desired setpoint specifications according Table 3 $\left(\boldsymbol{h}_t^s, \boldsymbol{Vx}_t^s, \boldsymbol{\psi}_t^s, \boldsymbol{\theta}_t^s, \boldsymbol{\phi}_t^s\right)$. Finally, $\boldsymbol{v}^m$ represents $\left(\boldsymbol{h}_t^m, \boldsymbol{Vx}_t^m, \boldsymbol{\psi}_t^m, \boldsymbol{\theta}_t^m, \boldsymbol{\phi}_t^m\right)$ which is the simulation results.

Finally, all results are compared to the nonlinear Matlab® toolbox *fmincon* using Eq. (77), without the procedure of breaking the initial system CEM. Thus, the objective is to analyze their behaviors and computational efforts.

5.3.1. FCA 1

The first configuration aims to show the total superposition of VCVs in both RASs, where $\boldsymbol{M_a}(\boldsymbol{u_b}) \in \mathbb{R}^{2\times5}$ and $\boldsymbol{M_b}(\boldsymbol{u_a}) \in \mathbb{R}^{5\times4}$ are

$$\hat{\boldsymbol{\tau}}_a = [X_p^b, N_p^b]^T \tag{82}$$

$$\hat{\boldsymbol{\tau}}_b = [X_p^b, Z_p^b, L_p^b, M_p^b, N_p^b]^T \tag{83}$$

The reason for choosing only X_p^b and N_p^b in the first group of VCVs is to direct control them by the RA group $\boldsymbol{u_a}$. Thus, the remaining VCVs will be controlled by group $\boldsymbol{u_b}$. Also, from physical concepts, the VCVs in Eq. (82) are more energetically efficient if controlled by the servomotors.

Moving on, the respective RASs are shown in Eq. (84) and (85) (see Box II).

5.3.2. FCA 2

The second configuration shows the partial superposition of VCVs in the RASs, where $\boldsymbol{M_a}(\boldsymbol{u_a}) \in \mathbb{R}^{2\times4}$ and $\boldsymbol{M_b}(\boldsymbol{u_b}) \in \mathbb{R}^{4\times4}$ are

$$\hat{\boldsymbol{\tau}}_a = [X_p^b, N_p^b]^T \tag{86}$$

$$\hat{\boldsymbol{\tau}}_b = [Z_p^b, L_p^b, M_p^b, N_p^b]^T \tag{87}$$

with Eq. (88) and (89) given in Box III.

5.3.3. Comparisons among FCA 1, FCA 2 and NonLinear approaches

The results of FCA1 and FCA2 presented at the previous sections are shown numerically in Table 4 and graphically in Fig. 23. *NL*1 and *NL*2 represent the solution provided by the IP and AS algorithms. It is possible to see that FCA 1 is more efficient than the nonlinear approaches.

Figs. 23-1 to 23-5 show the setpoints ($\boldsymbol{h}$, $\boldsymbol{Vx}$, $\boldsymbol{\psi}$, $\boldsymbol{\theta}$, $\boldsymbol{\phi}$) according Table 3. The red lines (dark and light) represent the FCA methodology, while the blue lines (also dark and light) represent the Interior Point and Active Set approaches. Figs. 23-6 to 23-9 represent the angles of servomotors 1 to 4, respectively. Although it is not critical, note that the solutions provided by the FCA are smoother than the ones obtained by the NL approaches, which is better for the aircraft flight dynamics. Figs. 23-10 and 23-11 show the resulting tilt angles for the propulsion motors δ_1 and δ_3, respectively. Motors δ_2 and δ_3 had similar results and, for the sake of simplicity, were not listed. Finally, Figs. 23-12 and 23-13 show the evolution of the squared error and the convergence time, respectively. Note that the convergence time for the traditional nonlinear approaches are much higher than the FCA which was, in some cases, 189 times faster.

To better understand the convergence process, Fig. 24 shows the system evaluation at $t = 5$ s. The convergence time for FCA 1 and FCA 2 were 1.8×10^{-3} and 2.0×10^{-3} s that took 8 and 12 iterations, respectively. It is possible to see that the convergence process of FCA 1 is smoother and took fewer iterations than FCA 2. Considering the time and evaluations, it is possible to conclude that, although FCA 2 presents a faster individual iteration time, the fact that it does not have all variables at subsystem 2 makes the process more difficult and, as consequence, the total convergence time higher.

6. Conclusions

This work has designed and analyzed a new methodology of control allocation mainly focused on autonomous systems with directional thrusters. The approach was presented thought tutorials and simulations of a QTR UAV.

The new methodology (named FCA) was formulated as a sequence of two or more partially-dependent linearized systems solved interactively. A dynamical procedure analyzes the convergence process and defines if it is necessary to change the original set's configuration. The mathematical differences among this approach and traditional linear and nonlinear ones were enforced through the paper.

By using a tutorial case, several possible configurations, including contingency scenarios, were analyzed, resulting in a heuristic theory that partial-depended systems can be fast and robust if the final linearized one embeds all virtual variables. The tutorial case was specially designed to be both numerically representative and easily reproduced. The results have demonstrated

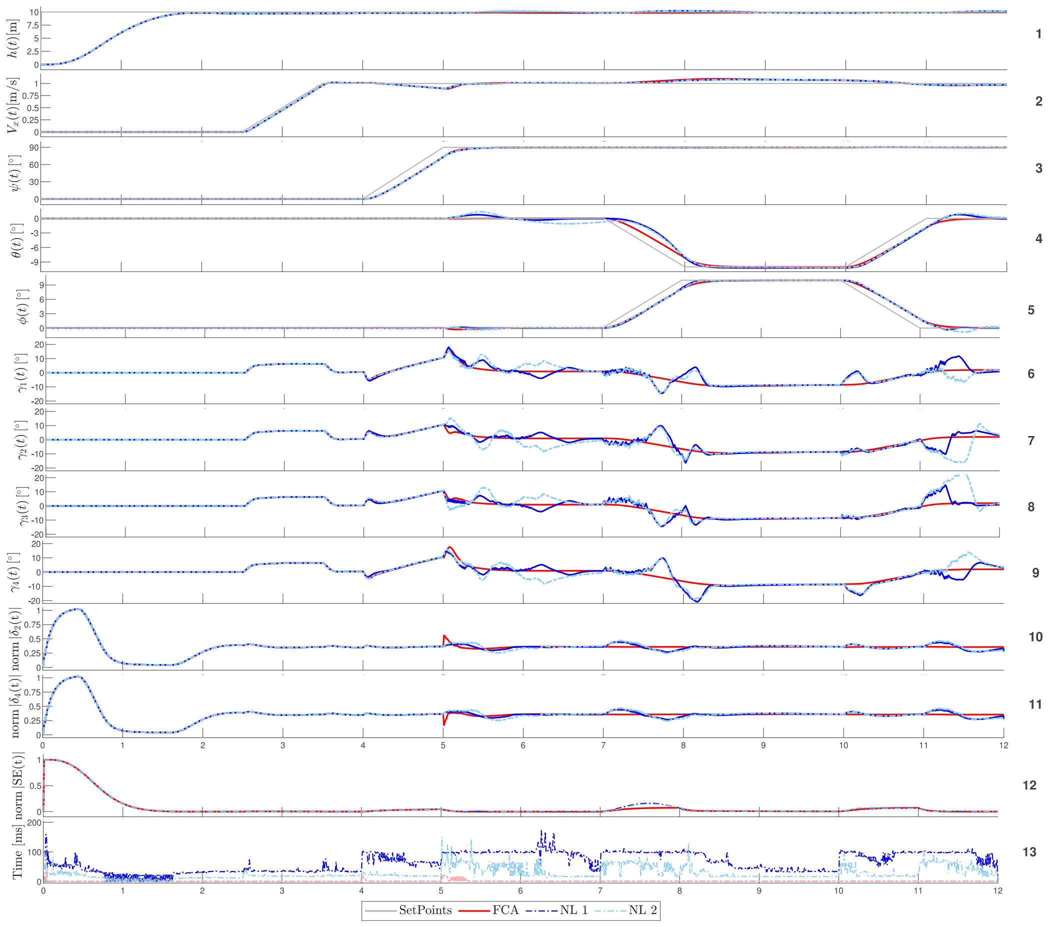


Fig. 23. QTR controlled responses with NonLinear programming (blue) and FCA (red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

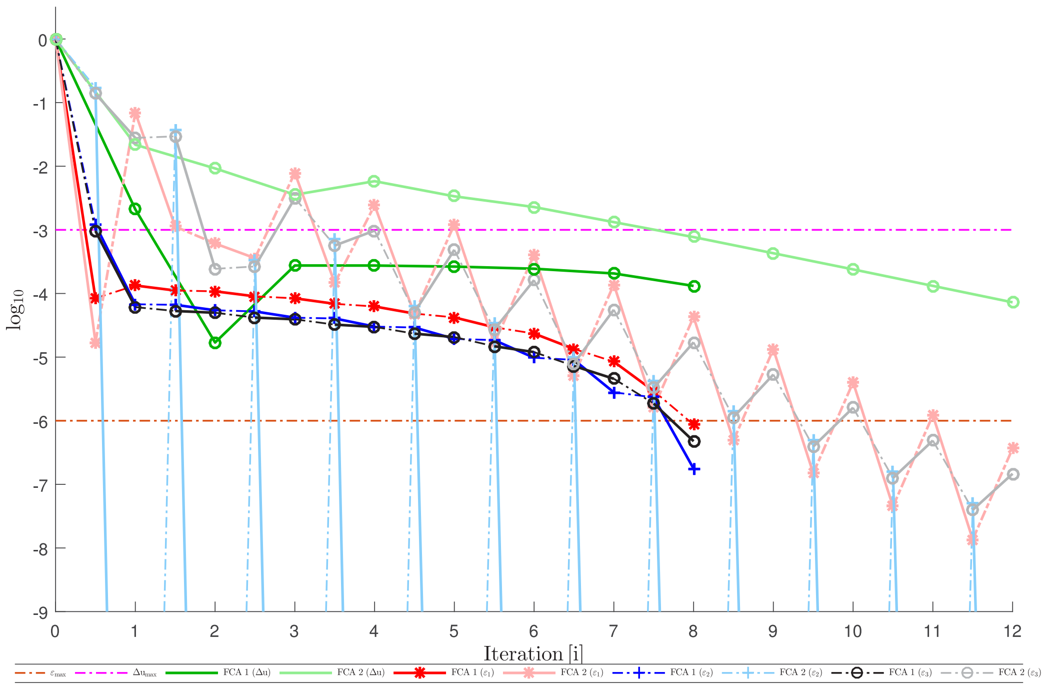


Fig. 24. FCA convergence evolution during the instant $t = 5$ s.

$$\overbrace{\begin{bmatrix} X_p^b \\ N_p^b \end{bmatrix}}^{\hat{\tau}_a} = \overbrace{\begin{bmatrix} k_1\delta_1 & k_1\delta_2 & k_1\delta_3 & k_1\delta_4 & 0 \\ -k_1 d_{\llcorner}\delta_1 & -k_1 d_{\llcorner}\delta_2 & k_1 d_{\llcorner}\delta_3 & k_1 d_{\llcorner}\delta_4 & k_2(\delta_1 c\gamma_1 + \delta_2 c\gamma_2 - \delta_3 c\gamma_3 - \delta_4 c\gamma_4) \end{bmatrix}}^{\boldsymbol{M}_a(\boldsymbol{u}_b)} \overbrace{\begin{bmatrix} \sin(\gamma_1) \\ \sin(\gamma_2) \\ \sin(\gamma_3) \\ \sin(\gamma_4) \\ 1 \end{bmatrix}}^{\boldsymbol{u}'_a} \tag{84}$$

$$\overbrace{\begin{bmatrix} X_p^b \\ Z_p^b \\ L_p^b \\ M_p^b \\ N_p^b \end{bmatrix}}^{\hat{\tau}_b} = \overbrace{\begin{bmatrix} k_1 s\gamma_1 & k_1 s\gamma_2 & k_1 s\gamma_3 & k_1 s\gamma_4 \\ k_1 c\gamma_1 & k_1 c\gamma_2 & k_1 c\gamma_3 & k_1 c\gamma_4 \\ (-k_1 d_{\llcorner} c\gamma_1 - k_2 s\gamma_1) & (k_1 d_{\llcorner} c\gamma_2 - k_2 s\gamma_2) & (k_1 d_{\llcorner} c\gamma_3 + k_2 s\gamma_3) & (-k_1 d_{\llcorner} c\gamma_4 + k_2 s\gamma_4) \\ k_1 d_{\llcorner} c\gamma_1 & -k_1 d_{\llcorner} c\gamma_2 & k_1 d_{\llcorner} c\gamma_3 & -k_1 d_{\llcorner} c\gamma_4 \\ (-k_1 d_{\llcorner} s\gamma_1 + k_2 c\gamma_1) & (k_1 d_{\llcorner} s\gamma_2 + k_2 c\gamma_2) & (k_1 d_{\llcorner} s\gamma_3 - k_2 c\gamma_3) & (-k_1 d_{\llcorner} s\gamma_4 - k_2 c\gamma_4) \end{bmatrix}}^{\boldsymbol{M}_b(\boldsymbol{u}_a)} \overbrace{\begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \end{bmatrix}}^{\boldsymbol{u}_b} \tag{85}$$

Box II.

$$\overbrace{\begin{bmatrix} X_p^b \\ N_p^b \end{bmatrix}}^{\hat{\tau}_a} = \overbrace{\begin{bmatrix} k_1\delta_1 & k_1\delta_2 & k_1\delta_3 & k_1\delta_4 & 0 \\ -k_1 d_{\llcorner}\delta_1 & -k_1 d_{\llcorner}\delta_2 & k_1 d_{\llcorner}\delta_3 & k_1 d_{\llcorner}\delta_4 & k_2(\delta_1 c\gamma_1 + \delta_2 c\gamma_2 - \delta_3 c\gamma_3 - \delta_4 c\gamma_4) \end{bmatrix}}^{\boldsymbol{M}_a(\boldsymbol{u}_b)} \overbrace{\begin{bmatrix} \sin(\gamma_1) \\ \sin(\gamma_2) \\ \sin(\gamma_3) \\ \sin(\gamma_4) \\ 1 \end{bmatrix}}^{\boldsymbol{u}'_a} \tag{88}$$

$$\overbrace{\begin{bmatrix} Z_p^b \\ L_p^b \\ M_p^b \\ N_p^b \end{bmatrix}}^{\hat{\tau}_b} = \overbrace{\begin{bmatrix} k_1 c\gamma_1 & k_1 c\gamma_2 & k_1 c\gamma_3 & k_1 c\gamma_4 \\ (-k_1 d_{\llcorner} c\gamma_1 - k_2 s\gamma_1) & (k_1 d_{\llcorner} c\gamma_2 - k_2 s\gamma_2) & (k_1 d_{\llcorner} c\gamma_3 + k_2 s\gamma_3) & (-k_1 d_{\llcorner} c\gamma_4 + k_2 s\gamma_4) \\ k_1 d_{\llcorner} c\gamma_1 & -k_1 d_{\llcorner} c\gamma_2 & k_1 d_{\llcorner} c\gamma_3 & -k_1 d_{\llcorner} c\gamma_4 \\ (-k_1 d_{\llcorner} s\gamma_1 + k_2 c\gamma_1) & (k_1 d_{\llcorner} s\gamma_2 + k_2 c\gamma_2) & (k_1 d_{\llcorner} s\gamma_3 - k_2 c\gamma_3) & (-k_1 d_{\llcorner} s\gamma_4 - k_2 c\gamma_4) \end{bmatrix}}^{\boldsymbol{M}_b(\boldsymbol{u}_a)} \overbrace{\begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \delta_4 \end{bmatrix}}^{\boldsymbol{u}_b} \tag{89}$$

Box III.

that, for multiple directional systems, the proposed approach has fast convergence.

This theory was tested and corroborated by using a QTR UAV system. Moreover, comparing the FCA against traditional nonlinear programming techniques, it was demonstrated that the new approach achieved a much superior computational performance (up to 189 times faster) with also better qualitative convergence results.

Finally, as future work the proposed FCA will be implemented in a real overactuated QTR with individual control actions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to thanks the support of CNPq, CAPES and INERGE. Moreover, the work reported in this paper was performed as part of an interdisciplinary research and development project undertaken by Federal University of Juiz de Fora. The author acknowledge the financial funding and support of the TBE under supervision of ANEEL - The Brazilian Regulatory Agency of Electricity , through project number PD-02651-0016/2019.

References

[1] Johansen TA, Fossen TI. Control allocation - A survey. Automatica 2013;49(5):1087–103.
[2] Saied M, Shraim H, Lussier B, Fantoni I, Francis C. Local controllability and attitude stabilization of multirotor UAVs: Validation on a coaxial octorotor. Robot Auton Syst 2017;91:128–38.
[3] Luenberger DG, Ye Y, et al. Linear and nonlinear programming, vol. 2. Springer; 1984.
[4] Levine WS. The control handbook: Control system applications (Chapter 8 - Control allocation). CRC press; 2018.
[5] Gai W, Liu J, Zhang J, Li Y. A new closed-loop control allocation method with application to direct force control. Int J Control Autom Syst 2018;16(3):1355–66.
[6] Durham WC. Constrained control allocation. J Guid Control Dyn 1993;16(4):717–25.
[7] Ahani A, Ketabdari MJ. Alternative approach for dynamic-positioning thrust allocation using linear pseudo-inverse model. Appl Ocean Res 2019;90:101854.
[8] Shi J, Zhang W, Li G, Liu X. Research on allocation efficiency of the redistributed pseudo inverse algorithm. Sci China Inf Sci 2010;53(2):271–7.
[9] Simon D, Härkegård O, Löfberg J. Command governor approach to maneuver limiting in fighter aircraft. J Guid Control Dyn 2016;40(6):1514–27.
[10] Oliveira EJ, Oliveira LW, Pereira JLR, Honório LM, Junior ICS, Marcato ALM. An optimal power flow based on safety barrier interior point method. Int J Electr Power Energy Syst 2015;64:977–85.
[11] Ryll M, Bülthoff HH, Giordano PR. A novel overactuated quadrotor unmanned aerial vehicle: Modeling, control, and experimental validation. IEEE Trans Control Syst Technol 2015;23(2):540–56.
[12] Mok K-T, Tan S-C, Hui SYR. Decoupled power angle and voltage control of electric springs. IEEE Trans Power Electron 2016;31(2):1216–29.
[13] Liu L, Li H, Xue Y, Liu W. Decoupled active and reactive power control for large-scale grid-connected photovoltaic systems using cascaded modular multilevel converters. IEEE Trans Power Electron 2015;30(1):176–87.

[14] Allenspach M, Bodie K, Brunner M, Rinsoz L, Taylor Z, Kamel M, et al. Design and optimal control of a tiltrotor micro aerial vehicle for efficient omnidirectional flight. 2020, arXiv preprint arXiv:2003.09512.
[15] da Silva MF, de Mello Honório L, dos Santos MF, dos Santos Neto AF, Cruz NA, Matos AC, et al. Project and control allocation of a 3 DoF autonomous surface vessel with aerial azimuth propulsion system. IEEE Access 2020.
[16] Bhargavapuri M, Sahoo SR, Kothari M, et al. Robust nonlinear control of a variable-pitch quadrotor with the flip maneuver. Control Eng Pract 2019;87:26–42.
[17] Regina BA, Honório LM, Pancoti AA, Silva MF, Santos MF, Lopes VM, et al. Hull and aerial holonomic propulsion system design for optimal underwater sensor positioning in autonomous surface vessels. Sensors 2021;21(2):571.
[18] Kamel M, Alexis K, Achtelik M, Siegwart R. Fast nonlinear model predictive control for multicopter attitude tracking on SO(3). In: Conference on control applications (CCA). IEEE; 2015, p. 1160–6.
[19] Kamel M, Verling S, Elkhatib O, Sprecher C, Wulkop P, Taylor Z, et al. The Voliro omniorientational hexacopter: An agile and maneuverable tiltable-rotor aerial vehicle. IEEE Robot Autom Mag 2018;25(4):34–44.
[20] Wang J, Solis JM, Longoria RG. On the control allocation for coordinated ground vehicle dynamics control systems. In: American control conference. IEEE; 2007, p. 5724–9.
[21] Zhang A, Ma S, Li B, Wang M. Curved path following control for planar eel robots. Robot Auton Syst 2018;108:129–39.
[22] Lei J, Chen H-F, Fang H-T. Primal–dual algorithm for distributed constrained optimization. Systems Control Lett 2016;96:110–7.
[23] Nocedal J, Wright S. Numerical optimization. Springer Science & Business Media; 2006.
[24] Hamming R. Numerical methods for scientists and engineers. Courier Corporation; 2012.
[25] Dennis Jr JE, Schnabel RB. Numerical methods for unconstrained optimization and nonlinear equations, vol. 16. Siam; 1996.
[26] Neto AFDS, Honório LDM, Da Silva MF, Junior ICDS, Westin LGF. Development of optimal parameter estimation methodologies applied to a 3DOF autonomous surface vessel. IEEE Access 2021;9:50035–49.
[27] Hager WW, Zhang H. A new active set algorithm for box constrained optimization. SIAM J Optim 2006;17(2):526–57.
[28] Martinsen F, Biegler LT, Foss BA. A new optimization algorithm with application to nonlinear MPC. J Process Control 2004;14(8):853–65.
[29] Fossen TI. Mathematical models for control of aircraft and satellites. Department of Engineering Cybernetics Norwegian University of Science and Technology; 2011.
[30] Beard RW, McLain TW. Small unmanned aircraft: Theory and practice. Princeton, EUA: Princeton University Press; 2012.
[31] Zhao W, Go TH. Quadcopter formation flight control combining MPC and robust feedback linearization. J Franklin Inst B 2014;351(3):1335–55.
[32] Argentim LM, Rezende WC, Santos PE, Aguiar RA. PID, LQR and LQR-pid on a quadcopter platform. In: International conference on informatics, electronics and vision (ICIEV). IEEE; 2013, p. 1–6.