SCHOOL OF INDUSTRIAL AND INFORMATION ENGINEERING

Master's degree in Space Engineering

POLITECNICO
MILANO 1863

Robust Guidance of a Free-Floating Manipulator with Gaussian Mixtures Models

Tutor: prof. Mauro MASSARI

Luca DIDONE'

900135

Academic year 2019/20

Abstract

The need of robotic systems capable of interacting with Resident Space Objects (RSO), providing on-orbit servicing, is always increasing. However, before any kind of maintenance work can take place, the servicing spacecraft must grab its target. In particular, the most challenging tasks comes when approaching a tumbling target. Various approaching methodologies have already been proposed for the capture maneuver of a non-stabilized target by a Spacecraft-Manipulator System (SMS). Despite of that, a really suitable and efficient solution that overcomes the guidance complexity of this problem, is still not present nowadays. Standard on-line optimization algorithms seem to be a good solution, but their high computational cost make them quite inaccurate for space applications. For this reason, alternative methods like convex programming or mapping have been applied with the hope of better results.

In this thesis it is proposed a new approach for the construction of a stable and robust guidance algorithm. The adopted method relies on the choice of writing the Dynamical System (DS), describing the chaser motion, as a Linear Parameter Varying (LPV) system, whose parameters are approximated using Gaussian Mixture Models (GMM). A learning algorithm, based on Gaussian Mixture Regression (GMR) techniques, is constructed starting from a set of kinematically feasible trajectories composing the dataset. The chaser's trajectories are in turn generated off-line by an optimal-control framework. It will be shown, thanks to Lyapunov stability theory, that the resulting DS is stable to the target and leads the mounted end-effector to the desired grasping location with the desired orientation and velocity.

Index

List of Figures

List of Tables

1. Introduction

On-orbit servicing holds the promise to refuel, maintain, upgrade, and repair existing spacecraft as well as to actively remove orbital debris. Before the servicing operations can take place, the servicing spacecraft must capture its target. Among the various methods that have been proposed to capture a space object, a chaser spacecraft equipped with a robotic manipulator is widely seen as a promising and versatile approach.

The robotic capture of cooperative and attitude-stabilized spacecraft has already been demonstrated; e.g. the Space Shuttle's Remote Manipulator System [9] has been used, under human control, to successfully catch an attitude-stabilized target.

Considering instead a tumbling target, the capturing maneuver is significantly more challenging. The greatest difficulty facing such kind of approach maneuver can probably be attributed to its guidance complexity. The trajectory generated by the guidance algorithm must be computed in a timely manner, the control constraints satisfied, the nonlinear multibody dynamics solved, and the propellant usage minimized. With respect to the capture maneuver, previous works considered mainly two distinct approaches.

The first one relies on the assumption that the chaser can initiate the capture maneuver at a close-enough distance, where the target's grapple fixture is within the chaser's manipulator grasping range. Simply actuating the robotic arm, while leaving the base-spacecraft uncontrolled, is enough to capture the target in this scenario. However, in presence of a tumbling target with large appendages, large time-varying keep-out zone constraints exclude the existence of a safe position in its proximity. Examples of this approach can be found in [10] or [11].

For this reason, the second method considers a chaser that starts its approaching maneuver at a sufficiently faraway hold position. Beginning from a "folded" configuration, where the studied system can be considered as a whole rigid body, the robotic arm deploys, and the chaser moves to the grasping position. This type of maneuver has been extensively studied with a wide variety of guidance and control approaches, such as optimal control [7] or optimization-based [12]. However, the usage of common on-line optimization problems may be inadequate for real-time space applications. The reasons for this can be found again in the complexity of the system nonlinear dynamics and, consequently, in the high computational costs that the on-board computer must sustain. Furthermore, since perfect control can never be satisfied in practice, control strategies that offer high robustness in the face of uncertainties, e.g. slight variations on the boundary conditions due to imprecise sensing or actuation, must be taken into account. Probably, the most efficient solution developed nowadays has been introduced in [1] and [2], where a sequential convex optimization procedure is employed. The nonlinear dynamics and the non-convex constraints are linearized with the consequent increment of the computational efficiency. Moreover, algorithms used to solve convex programming problems guarantee convergence in polynomial time.

The new approach presented in this work is aimed to construct a robust guidance algorithm rewriting the nonlinear DS, describing the reach and follow motion, as an LPV system. The motivation behind this originates from at least two sources. Thinking of LPV systems as a weighted combination of Linear Time Invariant (LTI) systems, the first reason relies on the possibility of using many concepts (e.g. invariant subspaces or bases) and tools (e.g. linear algebra tools) valid only for the linear case. Another motivation can be found in the fact that the solution of an optimal control problem, described trough a nonlinear DS, requires a solution to an associated Hamilton Jacobi Bellman partial differential equation. A procedure often complicated and applicable only under certain precise conditions. For LPV system instead, this goal can be addressed via the solution of a set of Linear Matrix Inequality (LMI) problems, i.e. essentially a set

of convex constraints. Many optimization problems with convex objective functions and LMI constraints can easily be solved efficiently using many existing software. The parameters of the LPV system may thus be estimated using arbitrary regressive techniques. Here, a probabilistic approach through Gaussian Mixture Regression (GMR) is employed. To train the system, a set of kinematically feasible demonstrations, generated off-line by an optimal control problem built on the minimization of the control forces, is needed. It will be shown, thanks to Lyapunov stability theory, that the resulting DS is stable and converges to the desired intercept location with the appropriate velocity direction. In particular, focusing on the end-effector, the goal for it was to reach the grasping fixture with zero relative velocity with respect to the target, while avoiding dangerous impacts. The use of LPV systems, whose parameters are computed off-line through regression techniques, should overcome the problems deriving from the dynamic's nonlinearity and guarantee robustness properties in front of small changes on the boundary conditions, thus permitting a fast computation of the chaser motion. This kind of procedure has been already adopted successfully on fixed-based robotic manipulators for on-ground applications [3], [4]. The challenging task of the thesis is then to replicate the same method with the increased complexity of the DS.

1.2 System Architecture

The chaser system is composed by a free-floating base-spacecraft equipped with a n degree of freedom robotic arm manipulator. Throughout the thesis the motion is represented in Cartesian coordinates system.

Fig. 1 shows a schematic of the control flow. The key point is the training of a DS, described as an LPV system, with the aim to reproduce the ideal motion of the Spacecraft-Manipulator System (SMS) and to generalize it outside the design conditions. With this purpose, an on-line block containing the LPV system, parametrized through GMMs, and an off-line learning block, that generates the training dataset, are needed. Referring to the learning blocks, first N kinematically feasible demonstrations are founded solving an optimal-control problem based on the minimization of the control forces. Here $\xi = (\boldsymbol{r}_b, \boldsymbol{r}_{e.e})$ is the vector representing the positions and orientations of the spacecraft base and of the end-effector, $\dot{\xi}$ and $\ddot{\xi}$ are its first and second derivatives respectively. Secondly, a learning algorithm is employed in order to obtain the vector of parameters $\boldsymbol{\theta}$ whereby the DS $\ddot{\xi} = \hat{f}(\xi, \dot{\xi})$ is approximated through GMM.

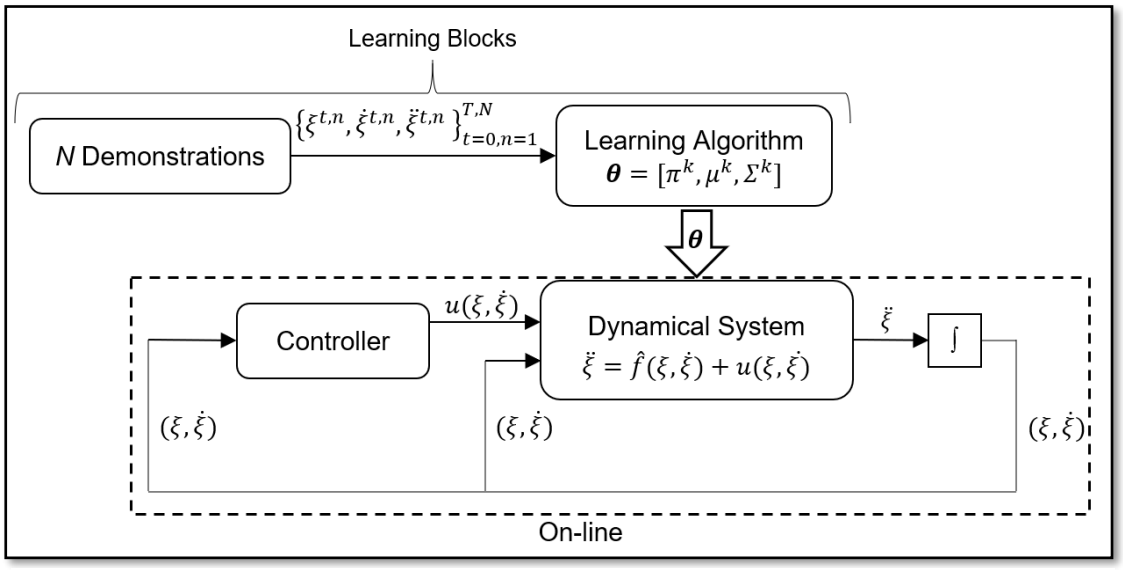


Figure 1: A typical system's architecture illustrating the control flow of the system considered in this thesis. An off-line learning block is used to create a dataset of feasible trajectories. An on-line block describes the parametrized dynamics of the chaser system.

Regarding the on-line blocks, it consists of a controller generating the required commands to follow the desired motion and a system block that uses the solution of the learning process to model the dynamics of the spacecraft-manipulator system:

$$\ddot{\xi} = \hat{f}(\xi, \dot{\xi}) + u(\xi, \dot{\xi})$$

The convergence of the dynamic system to the desired target location is guaranteed by Lyapunov stability theory.

1.3 Outline of the thesis

The following work is organized as follows: in Chapter 2 the properties of the SMS and the reformulation of the DS through LPV system are analyzed; in Chapter 3 the steps required to achieve the dataset are reported, while the learning algorithm is explained in Chapter 5; Chapter 4 describes both how to approximate the LPV system using GMM and its asymptotic stability; finally, in Chapters 6 and 7 the results obtained are presented and discussed, respectively.

2. System Dynamics

In this chapter the properties of the SMS are first introduced; then the LPV formulation of the DS and the prove of its stability are provided.

2.1 SMS set-up

A 2D squared-base spacecraft equipped with a three-link robotic arm has been considered. In order to simplify the problem, the end-effector was made to coincide with the end point of the last link. Dimensions and inertia parameters of the SMS are supposed to be known and are reported in tab.1.
Regarding the target satellite, a reliable estimate of the states and of its dynamic parameters are obtained from vision data thanks to cameras mounted on the chaser. A Kalman-Filter can be employed to promptly predict the target motion as soon as the filter converges [7]. As a consequence, the configuration, position and velocity that the SMS has to attain at grasping time t_f are known by applying an inverse-kinematic algorithm:

$$\begin{cases} \xi(t_f) = \xi^* = \left[\boldsymbol{r}_b(t_f), \boldsymbol{r}_{e.e}(t_f)\right]^T = \left[x_b(t_f), y_b(t_f), \alpha_b(t_f), x_{e.e}(t_f), y_{e.e}(t_f), \alpha_{e.e}(t_f)\right]^T \\ \dot{\xi}(t_f) = \dot{\xi}^* = \left[\dot{\boldsymbol{r}}_b(t_f), \dot{\boldsymbol{r}}_{e.e}(t_f)\right]^T = \left[\dot{x}_b(t_f), \dot{y}_b(t_f), \dot{\alpha}_b(t_f), \dot{x}_{e.e}(t_f), \dot{y}_{e.e}(t_f), \dot{\alpha}_{e.e}(t_f)\right]^T \\ \ddot{\xi}(t_f) = \ddot{\xi}^* = \left[\ddot{\boldsymbol{r}}_b(t_f), \ddot{\boldsymbol{r}}_{e.e}(t_f)\right]^T = \left[\ddot{x}_b(t_f), \ddot{y}_b(t_f), \ddot{\alpha}_b(t_f), \ddot{x}_{e.e}(t_f), \ddot{y}_{e.e}(t_f), \ddot{\alpha}_{e.e}(t_f)\right]^T \end{cases}$$

Spacecraft Base		**Robotic Manipulator**	
Parameter	Value	Parameter	Value
Mass	13 Kg	Mass x link	2.9 Kg
Inertia	0.28 Kgm^2	Inertia x link	0.0364 Kgm^2
Dimensions ($L \times W$)	0.27 x 0.27 m	Link length	0.38 m

Table 1: in the table the chosen values of the mass, inertia and dimension for the spacecraft base and robotic manipulator are reported.

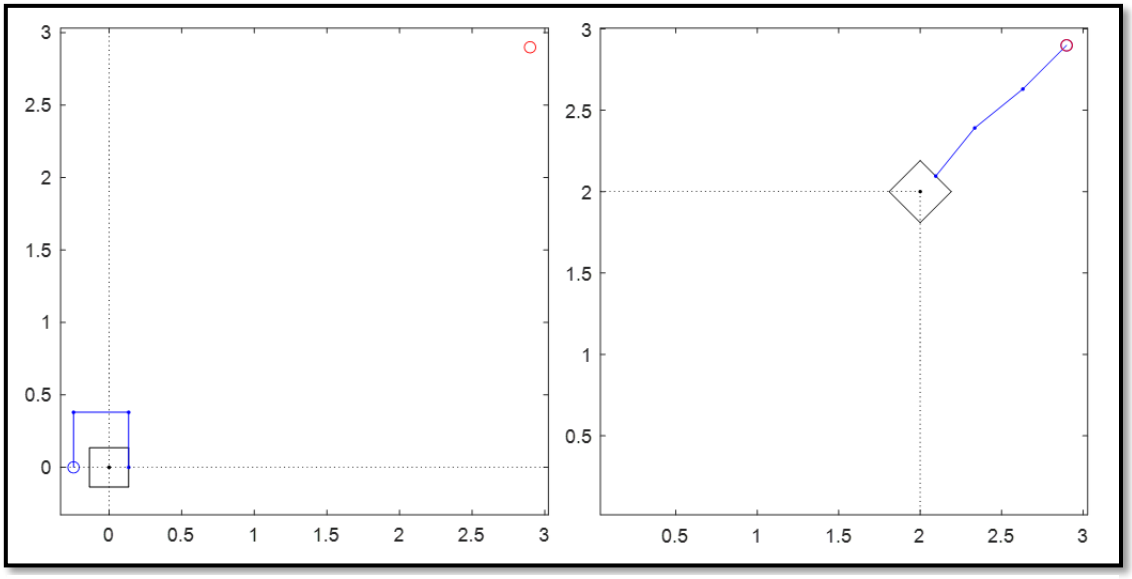


Figure 2: on the left side a possible initial configuration $\xi(t_0)$ of the chaser system is depicted, while on the right side it is reported its final configuration $\xi(t_f)$.

In figure 2 a possible initial "folded" configuration $\xi(t_0)$ (left-side) and the final one $\xi(t_f)$ when docking with the target (right-side) are depicted.

Additionally, the following assumptions are made:

1. The chaser multibody system and the target RSO are composed of rigid bodies.
2. The relative velocity between the two systems is set to zero at the beginning of phase two.
3. Environmental forces (gravity gradient, solar radiation pressure, *etc.*) as well as the relative orbital dynamics are neglected. This can be justified by the short duration of the studied maneuver and the close proximity of the two vehicles.
4. The target RSO has a designated grapple fixture.
5. The chaser mass remains constant during the maneuver.

2.2 DS described as LPV system

Dynamical systems are popular and powerful methods for autonomously generating stable motions according to training data-points. Formulating DSs as LPV systems allows modeling a wide class of nonlinear systems and the use of many tools from the linear systems theory for analysis and control. An LPV system can be thought as a

weighted combination of linear models, each valid at a specific operating point, whose state-space descriptions are known functions of time-varying parameters. The time variation of each of the parameters is not known in advance but is assumed to be measurable in real-time. The chosen continuous-time LPV system is given by the following model [3]:

$$\ddot{\xi}(t) = A^1\big(\theta_{A^1}(t)\big)\xi(t) + A^2\big(\theta_{A^2}(t)\big)\dot{\xi}(t) + u(t) \tag{1}$$

Here $\xi(t) = [\boldsymbol{r}_b, \boldsymbol{r}_{e.e}] \in \mathbb{R}^4$; $\theta_{A^i}(t) \in \mathbb{R}^{K_1 \times 1}\ \forall i \in \{1,2\}$ are the time-dependent vectors of scheduling parameters, not known a priori but measurable in real time:

$$\theta_{A^i} = [\theta_{A_1^i} \dots \theta_{A_{K_1}^i}]^T \qquad \forall i \in \{1,2\} \tag{2}$$

These parameters can be a function of time (t), state of the system $\xi(t)$ or external signal $d(t)$, i.e. $\theta_{A^i}(t, \xi(t), d(t))$. $A^i(.) : \mathbb{R}^{K_i} \to \mathbb{R}^{D \times D}\ \forall i \in \{1,2\}$ are the affine dependences of the state-space matrices on the scheduling parameters and the state vectors:

$$\begin{aligned} A^1\big(\theta_{A^1}(t)\big) &= \sum_{k=1}^{K_1} \theta_{A_k^1} A_k^1 \quad A_k^1 \in \mathbb{R}^{D \times D} \quad \theta_{A_k^1} \in \mathbb{R}^{1 \times 1} \\ A^2\big(\theta_{A^2}(t)\big) &= \sum_{k=1}^{K_2} \theta_{A_k^2} A_k^2 \quad A_k^2 \in \mathbb{R}^{D \times D} \quad \theta_{A_k^2} \in \mathbb{R}^{1 \times 1} \end{aligned} \tag{3}$$

To achieve convergence to the desired final state $[\xi^*, \dot{\xi}^*]$, the following control input vector $u(t)$ is proposed:

$$u(t) = \ddot{\xi}^* - A^1\big(\theta_{A^1}(t)\big)\xi^* - A^2\big(\theta_{A^2}(t)\big)\dot{\xi}^* \tag{4}$$

By introducing (3) and (4) in (1) it is obtained:

$$\ddot{\xi}(t) - \ddot{\xi}^* = \sum_{k=1}^{K_1} \theta_{A_k^1} A_k^1 \left(\xi(t) - \xi^*\right) + \sum_{k=1}^{K_2} \theta_{A_k^2} A_k^2 \left(\dot{\xi}(t) - \dot{\xi}^*\right) \tag{5}$$

It has then been obtained the LPV formulation of the DS in consideration.

2.3 Stability analysis

In general, it can be said that a DS is stable if the solution that start out near an equilibrium point ξ^* stay near ξ^* forever, i.e. the generated trajectories do not change too much under small perturbations. The kinds of stability that a DS can exploit are defined as follows:

1. An equilibrium state ξ^* is stable if there exists an arbitrary small number $\varepsilon_0 > 0$ with the following property: for all $\varepsilon_1, 0 < \varepsilon_1 < \varepsilon_0$ there is an $\varepsilon > 0$ such that if $\|\xi^* - \xi_0\| < \varepsilon$, then $\|\xi^* - \xi(t)\| < \varepsilon_1$ for all $t > t_0$.
2. An equilibrium state ξ^* is asymptotically stable if it is stable and there is an $\varepsilon > 0$ such that whenever $\|\xi^* - \xi_0\| < \varepsilon$, then $\xi(t) \to \xi^*$ as $t \to \infty$.
3. An equilibrium state ξ^* is globally asymptotically stable if it is stable and with arbitrary initial state $\xi_0 \in X, \xi(t) \to \xi^*$ as $t \to \infty$.

Given this, it is wanted that the LPV system (5) is asymptotically stable to the target $\left[\xi^* \ \dot{\xi}^*\right]^T$, or equally, that (5) asymptotically converges to the equilibrium point $\left[\xi^* \ \dot{\xi}^*\right]^T$; i.e.

$$\lim_{t\to\infty} \|\xi - \xi^*\| = 0 \tag{6}$$

$$\lim_{t\to\infty} \left\|\dot{\xi} - \dot{\xi}^*\right\| = 0 \tag{7}$$

To obtain this kind of behavior, (5) must meets the following constraints:

$$\begin{cases} A^1_{k^1} + \left(A^1_{k^1}\right)^T \prec 0 \quad A^2_{k^2} + \left(A^2_{k^2}\right)^T \prec 0 \\ A^1_{k^1} = \left(A^1_{k^1}\right)^T \quad \forall k^1 \in \{1 \dots K^1\} \quad \forall k^2 \in \{1 \dots K^2\} \\ 0 \le \theta_{A^1_{k^1}} \le 1 \quad 0 \le \theta_{A^2_{k^2}} \le 1 \end{cases} \tag{8}$$

The resulting LPV system is composed by equations (5) among with the constraints in (8). To prove the stability of the system, Lyapunov's second stability theorem has been utilized:

Lyapunov's second stability theorem: *A DS determined by* $\ddot{\xi} = \hat{f}\left(\xi, \dot{\xi}\right)$ *is asymptotically stable at the target* $[\xi^*, \dot{\xi}^*]$ *if there exists a continuous and continuously differentiable Lyapunov function* $V\left(\xi, \dot{\xi}\right) : \mathbb{R}^D \to \mathbb{R}$ *such that:*

$$\begin{cases} V(\xi,\dot{\xi}) > 0 \quad and \quad V(\xi^*,\dot{\xi}^*) = 0 \\ \dot{V}(\xi,\dot{\xi}) < 0 \quad and \quad \dot{V}(\xi^*,\dot{\xi}^*) = 0 \end{cases}$$

Considering a Lyapunov function of the form

$$V(\xi,\dot{\xi}) = \frac{1}{2}(\dot{\xi}-\dot{\xi}^*)^T(\dot{\xi}-\dot{\xi}^*) - \frac{1}{2}(\xi-\xi^*)^T \sum_{k=1}^{K_1} \theta_{A_k^1} A_k^1 (\xi-\xi^*) \tag{9}$$

it can be noted that it is radially unbounded, continuous and continuously differentiable. The first term is a quadratic form and so positive, while the second part is again a quadratic form multiplied by a negative definite matrix (A_k^1); consequently, $V > 0$. Additionally, $V(\xi^*,\dot{\xi}^*) = 0$. The first derivative results:

$$\dot{V} = \frac{dV}{dt} = (\dot{\xi}-\dot{\xi}^*)^T(\ddot{\xi}-\ddot{\xi}^*) - (\xi-\xi^*)^T \sum_{k=1}^{K_1} \theta_{A_k^1} A_k^1 (\dot{\xi}-\dot{\xi}^*) \tag{10}$$

inserting (5) it results:

$$\dot{V} = (\dot{\xi}-\dot{\xi}^*)^T \sum_{k=1}^{K_2} \theta_{A_k^2} A_k^2 (\dot{\xi}-\dot{\xi}^*) \tag{11}$$

Applying the same reasoning as for eq. (9), $\dot{V} < 0$ and $\dot{V}(\xi^*,\dot{\xi}^*) = 0$. This proves the global stability of the DS; i.e. ξ and $\dot{\xi}$ are bounded as ξ^* and $\dot{\xi}^*$ are bounded.

3. Trajectory generation: dataset

The underlying idea of the thesis is to approximate the parameters of the LPV system composed by equations (5) and (8) using GMM. To this aim, an appropriate set of kinematically feasible demonstrations generated off-line are needed. Due to the high complexity of the system and the impossibility to employ methods based on kinesthetic procedures (as done for fixed-base robotic manipulator in [3] and [4]), it has been chosen to construct the dataset through the resolution of an optimal-control problem. Through this chapter the procedure adopted to find the points composing the dataset is treated.

3.1 Newton-Euler equation in body coordinates

When open mechanic chains are considered, as in the case of a floating-base robotic manipulator, the Denavit-Hartenberg convention is a convenient way to express the kinematics. For this reason, the reference frames attached to each rigid body composing the system has been taken as depicted in fig. 3. The equations of motion are solved with respect to the inertial reference frame $\{X_0, Y_0\}$.

To obtain the Newton-Euler equation describing the overall motion of the SMS, the same procedure adopted in [5] has been employed. Neglecting the perturbation term, the resulting dynamic equation is:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}(\boldsymbol{t}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}(\boldsymbol{t}) = \boldsymbol{\tau} \tag{12}$$

Where $\boldsymbol{q}$ is a vector containing the minimal set of coordinates used to identify the configuration of the chaser system. In this case $\boldsymbol{q}$ is composed by the position and orientation of the base and the angles representing the inclinations of each link:

$$\boldsymbol{q} = \left[x_{s/c}\,, y_{s/c}\,, \alpha_{s/c}\,, q_1\,, q_2\,, q_3\right]^{\boldsymbol{T}}$$

$\boldsymbol{\tau}$ is the control input vector, $\boldsymbol{M}(\boldsymbol{q})$ and $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ are respectively the inertia matrix and the matrix containing the centrifugal and Coriolis forces/torques:

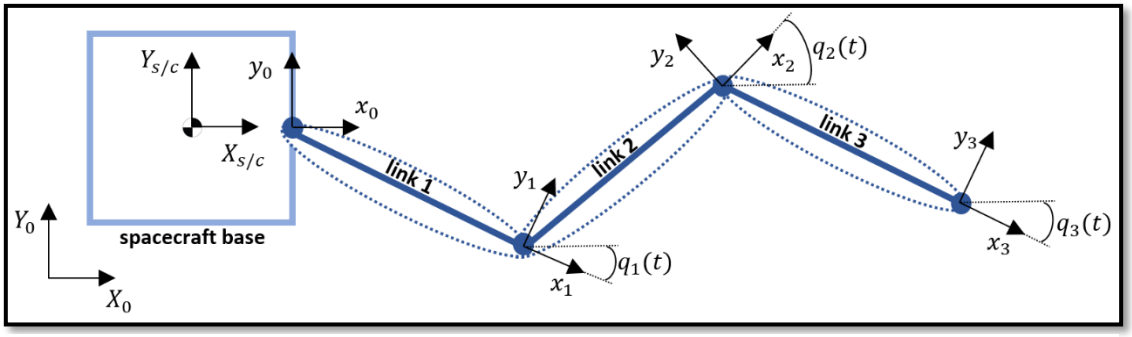


Figure 3: schematic representation of the spacecraft-manipulator system. The coordinate systems attached to the robotic links are depicted following Denavit-Hartenberg convention.

$$\boldsymbol{M}(\boldsymbol{q}) = \sum_k \boldsymbol{J}_k^T \boldsymbol{\Lambda}_k \boldsymbol{J}_k \tag{13}$$

$$\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \sum_k \boldsymbol{J}_k^T \left[\left(\boldsymbol{\Lambda}_k adj_{0,k} - adj_{0,k}^T \boldsymbol{\Lambda}_k \right) \boldsymbol{J}_k + \boldsymbol{\Lambda}_k \dot{\boldsymbol{J}}_k \right] \tag{14}$$

Here, $\boldsymbol{\Lambda}_k$ is the constant body inertia matrix and $adj_{0,k}$ is the Lie bracket matrix which uses the body twist of the Lie algebra as a linear mapping onto the Lie algebra itself. $\boldsymbol{J}_k$ and $\dot{\boldsymbol{J}}_k$ are the Jacobian matrix and its first derivative and are computed iteratively. The procedure to obtain $\boldsymbol{M}$ and $\boldsymbol{C}$ starting from $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ is described in [5].
Equation (12) can be rewritten in state-space form as:

$$\dot{\boldsymbol{x}} = \begin{Bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{Bmatrix} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{0} & -\boldsymbol{M}^{-1}\boldsymbol{C} \end{bmatrix} \begin{Bmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \end{Bmatrix} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{M}^{-1} \end{bmatrix} \boldsymbol{\tau} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{\tau} \tag{15}$$

3.2 Optimal-control problem using multiple shooting

To generate the chaser's trajectories, modeling of point-to-point motion has been considered. In such kind of motion, the final configuration $\boldsymbol{q}(t_f)$ of the system in consideration is designated, but the path used to reach it is irrelevant. Since also the initial configuration $\boldsymbol{q}(t_0)$ is known, the problem to be solved becomes then a differential Two-Point Boundary Value Problem (TPBVP). Thus, each feasible trajectory that wants to be generated can be traced back to an optimal-control problem for which a TPBVP has to be solved using numerical techniques. In particular, the optimization procedure

that has been chosen is based on the minimization of the control forces required to lead the system to the final desired position and configuration. To this aim a quadratic cost function is employed:

$$J = \sum \boldsymbol{\tau}^T \boldsymbol{W}_\tau \boldsymbol{\tau} \tag{16}$$

Where $\boldsymbol{W}_\tau$ is a weighting matrix. Moreover, to assure kinematically feasible trajectories, suitable constraints must be added to equation (12). As the attitude of the chaser is unknown during this optimization steps, an L_2 norm is used to constraint the control forces:

$$\|\boldsymbol{\tau}(t)\|_2 \leq \boldsymbol{\tau}_{max} \tag{17}$$

The desired system configuration to be reached at grasping is set as:

$$\boldsymbol{q}(t_f) = \boldsymbol{q}_f = [x_{s/c}^f, y_{s/c}^f, \alpha_{s/c}^f, q_1^f, q_2^f, q_3^f]^T \tag{18}$$

Finally, to avoid impacts between the chaser and the target systems, their relative velocity has to go to zero at t_f. Accordingly:

$$\dot{\boldsymbol{q}}(t_f) = \dot{\boldsymbol{q}}_f = [0, 0, 0, 0, 0, 0]^T \tag{19}$$

The resulting TPBVP is:

minimize: $$\min_{\boldsymbol{\tau}} J = \min_{\boldsymbol{\tau}} \sum \boldsymbol{\tau}^T \boldsymbol{W}_\tau \boldsymbol{\tau} \tag{20a}$$

subjected to: $$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}}(\boldsymbol{t}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}(\boldsymbol{t}) = \boldsymbol{\tau} \tag{20b}$$

$$\|\boldsymbol{\tau}(t)\|_2 \leq \boldsymbol{\tau}_{max} \tag{20c}$$

$$\boldsymbol{q}(t_f) = [x_{s/c}^f, y_{s/c}^f, \alpha_{s/c}^f, q_1^f, q_2^f, q_3^f]^T \tag{20d}$$

$$\dot{\boldsymbol{q}}(t_f) = [0, 0, 0, 0, 0, 0]^T \tag{20e}$$

Among the possible numerical approaches applicable to solve the TPBVP composed by equations (20), a multiple-shooting method is here used. In the direct multiple-shooting

method, the interval from $t_0 = 0$ to t_f is split into T equal subintervals by adding additional grid points:

$$t_0 < t_1 < \cdots < t_{T-1} < t_T = t_f$$

The method starts by guessing the values of $\boldsymbol{x}$ at all grid points t_i with $0 \leq i \leq T-1$. Denoting these guesses as $\boldsymbol{x}_i$, the problem is transformed to an Initial Value Problem (IVP) of the form

$$\begin{cases} \dot{\boldsymbol{x}} = f(t, \boldsymbol{x}(t)) \\ \boldsymbol{x}(t_i) = \boldsymbol{x}_i \end{cases} \tag{21}$$

that must be resolved for each subinterval. All the solutions $\boldsymbol{x}(t; t_i, \boldsymbol{x}_i)$ can then be pieced together to form a continuous trajectory if the values of $\boldsymbol{x}$ match at the grid points. Thus, solutions of the original TPBVP correspond to solutions of the following system of T equations:

$$\begin{cases} \boldsymbol{x}(t_1; t_0, \boldsymbol{x}_0) = \boldsymbol{x}_1 \\ \qquad\qquad \vdots \\ \boldsymbol{x}(t_{T-1}; t_{T-2}, \boldsymbol{x}_{T-2}) = \boldsymbol{x}_{T-1} \\ \boldsymbol{x}(t_T; t_{T-1}, \boldsymbol{x}_{T-1}) = \boldsymbol{x}_T \end{cases} \tag{22}$$

The central $T-1$ equations are the matching conditions, while the first and last equations are the boundary conditions $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ and $\boldsymbol{x}(t_f) = \boldsymbol{x}_f$ of the original problem. The Runge-Kutta direct method is been chosen as numerical method to solve the system of IVP.

Once solved, the problem returns the state of the system $\boldsymbol{q}_t = [\boldsymbol{q}_0 \ldots \boldsymbol{q}_T]^T$ in the boundary points of each subinterval. Here $t_f = 15s$ and $T = 36$ has been chosen, meaning that the state vector is known every $0.417s$.

3.3 Dataset

It is now possible to construct the dataset from the solution of the system of equations (20). As mentioned in chapter 2, the state $\xi(t)$ of the LPV system composed by equations (5) and (8) considers the position and the orientation of the spacecraft base and of the end-effector.

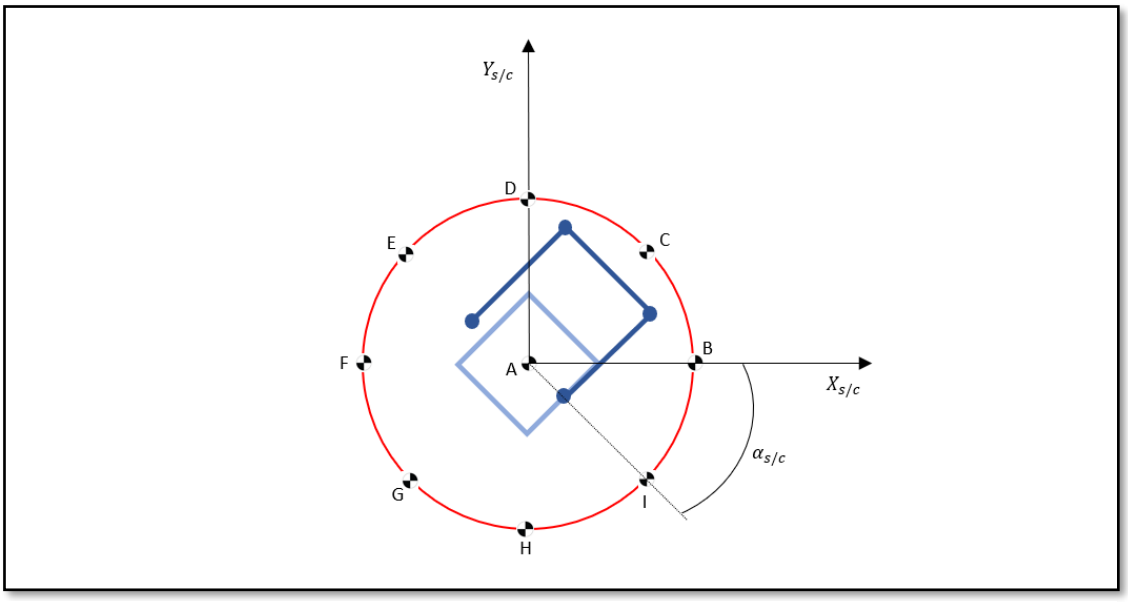


Figure 4: scheme of the various c.o.m. positions chosen as initial points to initiate the TPBVP.

For the base, $\boldsymbol{r}_b = [x_{s/c}, y_{s/c}, \alpha_{s/c}]$ is already known, while $\boldsymbol{r}_{e.e} = [x_{e.e}, y_{e.e}, \alpha_{e.e}]$ can be obtained directly from $\boldsymbol{q}$:

$$\boldsymbol{r}_{e.e} = \boldsymbol{r}_b + [R]_{\alpha_{s/c}} \boldsymbol{r}_0^{(x_{s/c})} + [R]_{\beta} \boldsymbol{r}_1^{(x_0)} + [R]_{\gamma} \boldsymbol{r}_2^{(x_1)} + [R]_{\delta} \boldsymbol{r}_3^{(x_2)} \tag{23}$$

Where in $\boldsymbol{r}_j^i$ the apex i refers to the reference system that is considered. $[R]_i$ is the transformation matrix between reference frames:

$$[R]_i = \begin{bmatrix} \cos(i) & -\sin(i) \\ \sin(i) & \cos(i) \end{bmatrix} \tag{24}$$

and

$$\begin{cases} \beta = \alpha_{s/c} + q_1 \\ \gamma = \beta + q_2 \\ \delta = \gamma + q_3 \end{cases} \tag{25}$$

The velocity and acceleration of the end effector $\dot{\boldsymbol{r}}_{e.e}$ and $\ddot{\boldsymbol{r}}_{e.e}$ are the first and second derivatives of equation (23) respectively.

With the goal of creating a dataset as complete as possible, the TPBVP (20) has been solved for $N = 27$ different initial configurations. Fig. 4 shows the different positions of the spacecraft center of mass, for each of which three different inclinations has been

considered ($\alpha_{s/c} = 0°$, $\alpha_{s/c} = -45°$ and $\alpha_{s/c} = 45°$). This results in $N = 27$ different feasible trajectories, each divided into $T = 36$ intervals for a total of $M = N \times T = 972$ points in which the state of the system is known. The dataset can then be written as:

$$\left\{\xi^{t,n}, \dot{\xi}^{t,n}, \ddot{\xi}^{t,n}\right\}_{t=0,n=1}^{T,N} = \left\{\xi^{m}, \dot{\xi}^{m}, \ddot{\xi}^{m}\right\}_{m=1}^{M} \tag{26}$$

4. Multivariate regression

This chapter first introduces the concept of GMM, then describes how the parameters of the LPV system composed by equations (5) and (8) can be approximated via a GMM from the training dataset (26), maintaining the stability properties described in section 2.3.

4.1 Gaussian Mixture Models

Before defining what a GMM is, lets introduce the concept of Gaussian distribution. In probability theory, a Gaussian (or normal) distribution is a type of continuous probability distribution for real-valued random variable. The general form of its Probability Density Function (PDF) is:

$$\mathcal{N}(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{27}$$

Here μ is the mean while σ is the standard deviation, i.e. the square root of the variance σ^2. A random variable x with a Gaussian distribution is said to be normally distributed, and is called a normal deviate. The normal distribution generalizes to $\mathbb{R}^n$, in which case it is known as multivariate Gaussian distribution and it has the form:

$$\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sqrt{\frac{1}{(2\pi)^d \det(\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right) \tag{28}$$

Where $\boldsymbol{\Sigma}$ is the covariance matrix and $\boldsymbol{\mu}$ represent still the mean, thought now it is vector-valued. The importance of normal distributions in statistics is mainly due to the central limit theorem. This last states that, under some conditions, the average of many observations of a random variable with finite mean and variance is itself a random variable, whose distribution converges to a normal distribution as the number of samples increases. Therefore, physical quantities that are expected to be the sum of many

independent processes, as in the studied case, often have distributions that are nearly normal.

Thus, a GMM is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. One hint that the data might follow a mixture model is that the data looks multimodal, i.e. there is more than one "peak" in the distribution of data. Trying to fit a multimodal distribution with a unimodal (one "peak") model will generally give a poor fit. For this reason, using a mixture of Gaussian distributions makes intuitive sense for the studied case. Furthermore, GMMs maintain many of the theoretical and computational benefits of Gaussian models, making them practical for efficiently modeling large datasets.

A GMM with K Gaussians in the multivariate case is parametrized by two types of values, the mixture components weights π^k and a mean $\boldsymbol{\mu}^k$ and covariance matrix $\boldsymbol{\Sigma}^k$ for each Gaussian component. The mixture component weights satisfy the relation $\sum_{k=1}^{K} \pi^k = 1$, so that the total probability distribution normalizes to 1. The multi-dimensional GMM can then be written as:

$$P(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi^k \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{\mu}^k, \boldsymbol{\Sigma}^k\right) \tag{29}$$

The procedure to obtain the parameters $\theta^k = \left\{\pi_1^k, \pi_2^k, \mu^k, \Sigma^k\right\}$ is introduced and explained in the following chapters.

4.2 LPV system through GMM

To estimate the new DS, the parameters of the LPV system becomes the priors $\pi_j^k \; \forall j \in \{1,2\}$, the means $\boldsymbol{\mu}^k$ and the covariance matrices $\boldsymbol{\Sigma}^k$ of the $k \in \{1 \dots K\}$ Gaussian functions. $\boldsymbol{\mu}^k$ and $\boldsymbol{\Sigma}^k$ for a Gaussian k are defined by:

$$\boldsymbol{\mu}^k = \begin{pmatrix} \mu_{\xi}^k \\ \mu_{\dot{\xi}}^k \\ \mu_{\ddot{\xi}}^k \end{pmatrix} \qquad \boldsymbol{\Sigma}^k = \begin{pmatrix} \Sigma_{\xi}^k & \Sigma_{\xi\dot{\xi}}^k & \Sigma_{\xi\ddot{\xi}}^k \\ \Sigma_{\dot{\xi}\xi}^k & \Sigma_{\dot{\xi}}^k & \Sigma_{\dot{\xi}\ddot{\xi}}^k \\ \Sigma_{\ddot{\xi}\xi}^k & \Sigma_{\ddot{\xi}\dot{\xi}}^k & \Sigma_{\ddot{\xi}}^k \end{pmatrix} \tag{30}$$

The parameters are collected into a matrix $\boldsymbol{\theta} = \{\theta^1 \dots \theta^K\}$ where $\theta^k = \left\{\pi_1^k, \pi_2^k, \mu^k, \Sigma^k\right\}$.

Given a set of N demonstrations $\{\xi^{t,n}, \dot{\xi}^{t,n}, \ddot{\xi}^{t,n}\}_{t=0,n=1}^{T,N}$, each recorded point in the trajectories $[\xi^m, \dot{\xi}^m, \ddot{\xi}^m]$ is associated with two Probability Density Functions (PDF) $P_j(x;\boldsymbol{\theta})$, one related to ξ and the other one to $\dot{\xi}$:

$$P_j(x;\boldsymbol{\theta}) = \sum_{k=1}^{K_j} \pi_j^k \, P_j(x|k) \qquad \forall (j,x) \in \{(1,\xi),(2,\dot{\xi})\} \tag{31}$$

Here $P_j(x|k)$ is the normal or Gaussian conditional PDF (28) corresponding to the k^{th} Gaussian function:

$$P_j(x|k) = \mathcal{N}(x|\mu_x^k, \Sigma_x^k) = \frac{1}{\sqrt{(2\pi)^6|\Sigma_x^k|}} \exp\left[-\frac{1}{2}([x]-\mu_x^k)^T(\Sigma_x^k)^{-1}([x]-\mu_x^k)\right] \tag{32}$$

$\forall x \in \mathbb{R}^4$. Taking the posterior mean estimate of $P(\ddot{\xi}|\xi,\dot{\xi})$ yields (as done in [6]):

$$\begin{aligned}\ddot{\xi} = \sum_{k=1}^{K_1} \frac{\pi_1^k P_1(\xi|k)}{\sum_{i=1}^{K_1} \pi_1^i P_1(\xi|i)} \left(\mu_{\ddot{\xi}}^k + \Sigma_{\ddot{\xi}\xi}^k \left(\Sigma_\xi^k\right)^{-1}\left(\xi - \mu_\xi^k\right)\right) + \\ + \sum_{k=1}^{K_2} \frac{\pi_2^k P_2(\dot{\xi}|k)}{\sum_{i=1}^{K_2} \pi_2^i P_2(\dot{\xi}|i)} \left(\mu_{\ddot{\xi}}^k + \Sigma_{\ddot{\xi}\dot{\xi}}^k \left(\Sigma_{\dot{\xi}}^k\right)^{-1}\left(\dot{\xi} - \mu_{\dot{\xi}}^k\right)\right)\end{aligned} \tag{33}$$

To simplify eq. (33) the following change of notation is used:

$$\begin{cases} A_k^1 = \Sigma_{\ddot{\xi}\xi}^k \left(\Sigma_\xi^k\right)^{-1} & A_k^2 = \Sigma_{\ddot{\xi}\dot{\xi}}^k \left(\Sigma_{\dot{\xi}}^k\right)^{-1} \\ b_k^1 = \mu_{\ddot{\xi}}^k - A_k^1 \mu_\xi^k & b_k^2 = \mu_{\ddot{\xi}}^k - A_k^2 \mu_{\dot{\xi}}^k \\ h_k^j(x) = \dfrac{\pi_j^k P_j(x|k)}{\sum_{i=1}^{K_j} \pi_j^i P_j(x|i)} & \forall (j,x) \in \{(1,\xi),(2,\dot{\xi})\} \end{cases} \tag{34}$$

$0 \leq h_k^j(x) \leq 1$ here is a continuous and continuously differentiable function that gives a measure of the relative influence of each Gaussian locally. Substituting eq. (34) into (33) it is obtained:

$$\ddot{\xi} = \hat{f}(\xi,\dot{\xi}) = \sum_{k=1}^{K_1} h_k^1(\xi)\left(A_k^1\xi + b_k^1\right) + \sum_{k=1}^{K_2} h_k^2(\dot{\xi})(A_k^2\dot{\xi} + b_k^2) \quad \forall \xi,\dot{\xi},\ddot{\xi} \in \mathbb{R}^D \tag{35}$$

That is the representation of the unforced DS (1) with GMM. $\hat{f}(\xi, \dot{\xi})$ is now expressed as a nonlinear sum of linear dynamical systems. To achieve convergence, the control input vector (4) becomes simply:

$$u(\xi, \dot{\xi}) = \ddot{\xi}^* \tag{36}$$

Adding eq. (36) to eq. (35) it is obtained the GMM formulation of the LPV system (5):

$$\ddot{\xi} - \ddot{\xi}^* = \sum_{k=1}^{K_1} h_k^1(\xi)(A_k^1 \xi + b_k^1) + \sum_{k=1}^{K_2} h_k^2(\dot{\xi})(A_k^2 \dot{\xi} + b_k^2) \tag{37}$$

4.3 Stability analysis

The DS (37) asymptotically converges to $\left[\xi^* \;\; \dot{\xi}^*\right]^T$ if it meets the following constraints:

$$\begin{cases} A_k^j + \left(A_k^j\right)^T \prec 0 & \forall j \in \{1,2\} \\ A_k^1 = \left(A_k^1\right)^T & \forall k \in \{1 \dots K^1\} \\ b_k^1 = -A_k^1 \xi^* \quad ; & b_k^2 = -A_k^2 \dot{\xi}^* \end{cases} \tag{38}$$

The above system is the equivalent GMM's formulation of (8). As done in section 2.3, Lyapunov stability theorem is utilized to prove convergence of the system of equations composed by (37) and (38). Consider the following Lyapunov function:

$$V(\xi, \dot{\xi}) = \frac{1}{2}\left(\dot{\xi} - \dot{\xi}^*\right)^T \left(\dot{\xi} - \dot{\xi}^*\right) - \frac{1}{2}(\xi - \xi^*)^T \sum_{k=1}^{K_1} h_k^1(\xi) A_k^1 (\xi - \xi^*) \tag{39}$$

Thanks to the first constraint of (38), V results positive $V > 0$. In addition, $V(\xi^*, \dot{\xi}^*) = 0$. The first derivative of the Lyapunov function results:

$$\begin{aligned} \dot{V} = \frac{dV}{dt} &= \left(\dot{\xi} - \dot{\xi}^*\right)^T \left(\ddot{\xi} - \ddot{\xi}^*\right) - \sum_{k=1}^{K_1} h_k^1(\xi)(\xi - \xi^*)^T A_k^1 \left(\dot{\xi} - \dot{\xi}^*\right) \\ &= \left(\dot{\xi} - \dot{\xi}^*\right)^T \sum_{k=1}^{K_2} \theta_{A_k^2} A_k^2 \left(\dot{\xi} - \dot{\xi}^*\right) \end{aligned} \tag{40}$$

Where equation (37) and the constraints in the last row of (38) has been substituted. Again, a quadratic form multiplied a negative definite matrix is obtained. Consequently, $\dot{V} < 0$. Furthermore $\dot{V}(\xi^*, \dot{\xi}^*) = 0$. The conditions of Lyapunov stability theorem are therefore satisfied. The DS

$$\begin{cases} \ddot{\xi} = \sum_{k=1}^{K_1} h_k^1(\xi)\left(A_k^1 \xi + b_k^1\right) + \sum_{k=1}^{K_2} h_k^2(\dot{\xi})(A_k^2 \dot{\xi} + b_k^2) + u \\ u = \ddot{\xi}^* \\ \qquad \begin{cases} A_k^j + \left(A_k^j\right)^T \prec 0 & \forall j \in \{1,2\} \\ A_k^1 = \left(A_k^1\right)^T & \forall k \in \{1 \dots K^1\} \\ b_k^1 = -A_k^1 \xi^* \quad ; \quad b_k^2 = -A_k^2 \dot{\xi}^* \end{cases} \end{cases} \tag{41}$$

ss stable and exhibits convergence behavior to $\left[\xi^* \ \dot{\xi}^*\right]^T$.

5. Learning algorithm

In this chapter, a learning algorithm needed to estimate the parameters $\boldsymbol{\theta}$ of the DS given in (41) is presented. As training procedure, an optimization approach based on the minimization of a Mean Square Error (MSE) has been chosen. Moreover, an alternative simplified solution for this last is given.

5.1 MSE-based approach

This method uses the Mean Square Error (MSE) as a means to quantify the accuracy of the estimation. The system to be optimized is the following:

minimize:
$$\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \sum_{m=1}^{M} \left(\ddot{\hat{\xi}} - \ddot{\xi}^m\right)^T \left(\ddot{\hat{\xi}} - \ddot{\xi}^m\right) \tag{42}$$

subjected to:
$$\begin{cases} A_{k_i}^i + \left(A_{k_i}^i\right)^T \prec 0 \\ A_{k_i}^1 = \left(A_{k_i}^1\right)^T \\ b_{k_i}^1 = -A_{k_i}^1 \xi^* \\ b_{k_i}^2 = -A_{k_i}^2 \dot{\xi}^* \end{cases} \qquad \begin{cases} 0 \prec \begin{pmatrix} \Sigma_{\dot{\xi}}^{k_i} & \Sigma_{\dot{\xi}\ddot{\xi}}^{k_i} \\ \Sigma_{\ddot{\xi}\dot{\xi}}^{k_i} & \Sigma_{\ddot{\xi}}^{k_i} \end{pmatrix} \\ 0 \prec \begin{pmatrix} \Sigma_{\dot{\xi}}^{k_i} & \Sigma_{\dot{\xi}\ddot{\xi}}^{k_i} \\ \Sigma_{\ddot{\xi}\dot{\xi}}^{k_i} & \Sigma_{\ddot{\xi}}^{k_i} \end{pmatrix} \\ 0 \leq \pi_i^{k_i} \leq 1 \\ \sum_{k_i}^{K_i} \pi_i^{k_i} = 1 \end{cases} \tag{43}$$

for $\forall k_i \in \{1 \dots K_i\}$ and $\forall i \in \{1,2\}$. $C(\boldsymbol{\theta})$ is the cost function, $\boldsymbol{\theta}$ is the vector of unknown parameters and M is the number of training data-points. $\ddot{\hat{\xi}}$ is computed directly from (35) and $\ddot{\xi}^m$ is taken from the training-set (24). The left-side constraints in (43) assure asymptotical stability, while those on the right follows from the definition of positiveness and bounded integrality for the GMM density; see [4]. The number of optimization parameters required is $K(2 + 3d + 4d^2)$, i.e. the sum of priors, means and covariance matrices, respectively. Considering a number $K = 4$ of Gaussians, the total number of parameters results $P = 656$. It can be noted that the learning grows linearly with the number of Gaussians and quadratically with the dimension.

5.2 Alternative optimization methods

To simplify the optimization procedure and increase its performances, a change of optimization parameters can be performed [8]. Defining:

$$\begin{cases} \tilde{\pi}_j^k = \ln\left(\pi_j^k\right) \\ L_\xi^k = Chol(\Sigma_\xi^k) \\ L_{\dot{\xi}}^k = Chol(\Sigma_{\dot{\xi}}^k) \end{cases} \tag{44}$$

where L_x^k are the Cholesky decomposition of Σ_x^k. L_x^k always exist since Σ_x^k are positive definite matrices. Furthermore, substituting the last two constraints on the left-side of (43) into (35), the equation describing the evolution of the motion becomes:

$$\ddot{\xi} = \hat{f}(\xi, \dot{\xi}) = \sum_{k=1}^{K_1} h_k^1(\xi) A_k^1 (\xi - \xi^*) + \sum_{k=1}^{K_2} h_k^2(\dot{\xi}) A_k^2 (\dot{\xi} - \dot{\xi}^*) \tag{45}$$

Considering equations (44) and (45) and defining the new optimization parameters as:

$$\theta^k = \left\{\tilde{\pi}_1^k, \tilde{\pi}_2^k, \mu^k, L_\xi^k, L_{\dot{\xi}}^k, A_1^k, A_2^k\right\}$$

the alternative MSE optimization can be expressed as:

minimize:
$$\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \sum_{m=1}^{M} \left(\ddot{\hat{\xi}}_m - \ddot{\xi}\right)^T \left(\ddot{\hat{\xi}}_m - \ddot{\xi}\right) \tag{46}$$

subjected to:
$$\begin{cases} A_{k_i}^i + \left(A_{k_i}^i\right)^T \prec 0 \\ A_{k_i}^1 = \left(A_{k_i}^1\right)^T \end{cases} \tag{47}$$

$\ddot{\hat{\xi}}$ are computed directly from (45). The proposed change permits to automatically satisfy the right-hand side constraints of (43) and the last two of the left-side are already embedded into the LPV equation of motion. The number of optimization parameters in this case is reduced to $P = K(2 + 2d + 4d^2 - 30) = 512$. In fact, in this case $\mu_{\dot{\xi}}^k$ might be not considered and the presence of two lower triangular matrices eliminate others $2K\left({}^{d^2}/_2 - {}^{d}/_2\right)$ variables.

A Sequential Quadratic Programming (SQP) approach has been used to solve the optimization problem algorithm. In addition, since the Non-Linear Programming (NLP) problem (46) is non-convex, the initial guess θ_0^k for the optimization problem affects the quality of the solutions. The following procedure provides a simple and efficient way to obtain a feasible initial guess:

1. Given the dataset $\left\{\xi^{t,n}, \dot{\xi}^{t,n}, \ddot{\xi}^{t,n}\right\}_{t=0,n=1}^{T,N}$, separate the trajectories into a number of subsets equal to the chosen number of gaussians K; i.e., for each trajectory create:

$$\begin{cases} \left\{\xi^{t,n}, \dot{\xi}^{t,n}\right\}_{t=0,n=1}^{T/K,N} = k_1 \\ \left\{\xi^{t,n}, \dot{\xi}^{t,n}\right\}_{t=T/K,n=1}^{2T/K,N} = k_2 \\ \vdots \\ \left\{\xi^{t,n}, \dot{\xi}^{t,n}\right\}_{t=(K-1)T/K,n=1}^{T,N} = k_K \end{cases}$$

2. Calculate the mean and covariance matrix for the K subsets:

$$\begin{cases} \check{\mu}^k = mean(k_k) \\ \check{\Sigma}^k = cov(k_k) \end{cases}$$

   Here the accent denotes that the parameters are the initial guesses.

3. Take the priors as:

$$\check{\pi}_j^k = \frac{1}{K}$$

However, the proposed initialization method cannot ensure to find the globally optimal solution due to the non-convexity of the NLP problem. Solvers are usually very sensitive to initialization of the parameters and will often converge to some local minima of the objective function.

Once the optimization process is completed, the GMM's parameters can be reconstructed as:

$$\begin{cases} \pi_j^k = e^{\tilde{\pi}_j^k} / \left(\sum_{k=1}^{K_1} e^{\tilde{\pi}_j^k}\right) \\ \Sigma_\xi^k = L_\xi^k \left(L_\xi^k\right)^T \\ \Sigma_{\dot{\xi}}^k = L_{\dot{\xi}}^k \left(L_{\dot{\xi}}^k\right)^T \\ \Sigma_{\ddot{\xi}\xi}^k = A_1^k \left(\Sigma_\xi^k\right)^T \\ \Sigma_{\ddot{\xi}\dot{\xi}}^k = A_2^k \left(\Sigma_{\dot{\xi}}^k\right)^T \end{cases} \tag{48}$$

6. Results

Parameter	Value
Number of dimensions	$d = 6$
Number of training data-points	$M = 972$
Number of Gaussians	$K = 4$

Table 2: parameters used to create the DS (49).

Given the presented framework, the DS

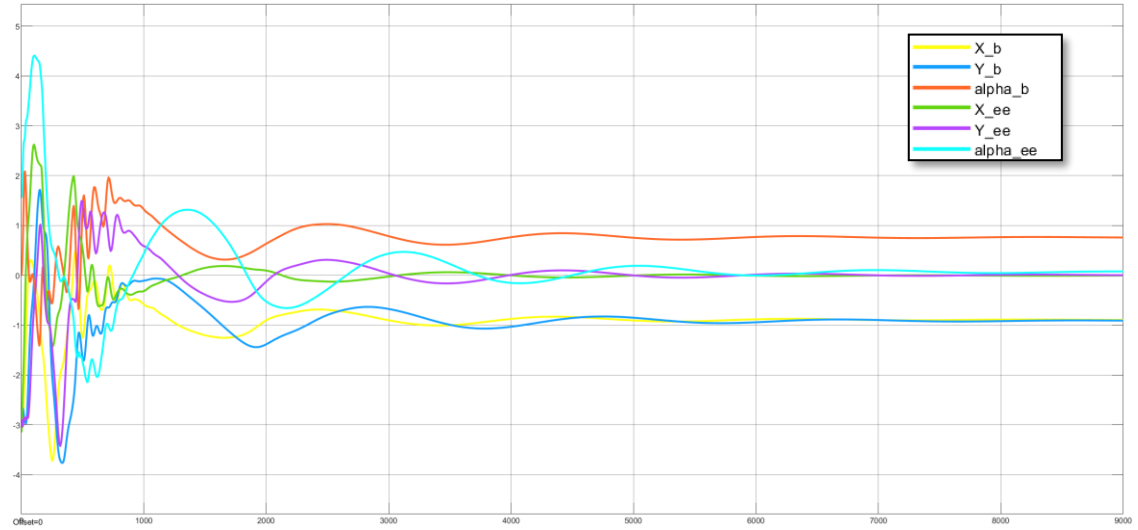
$$\ddot{\xi} = \hat{f}(\xi, \dot{\xi}) + u = \sum_{k=1}^{K_1} h_k^1(\xi) A_k^1(\xi - \xi^*) + \sum_{k=1}^{K_2} h_k^2(\dot{\xi}) A_k^2(\dot{\xi} - \dot{\xi}^*) + \ddot{\xi}^* \tag{49}$$

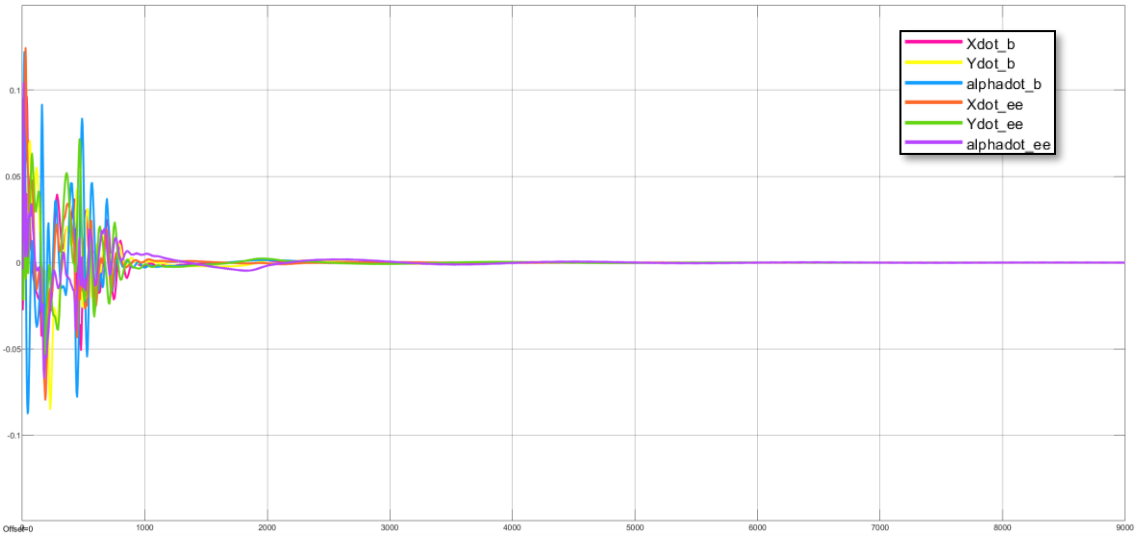
has been constructed utilizing the parameters in table 2. Additionally, the final configuration, velocity, and acceleration of the chaser system, used also inside (45) to run the training algorithm, has been chosen as:

$$\begin{cases} \xi^* = [-0.89\,;\, -0.89\,;\, \frac{\pi}{4}\,;\, 0\,;\, 0\,;\, \frac{\pi}{30}] \\ \dot{\xi}^* = [0\,;0\,;0\,;0\,;0\,;0] \\ \ddot{\xi}^* = [0\,;0\,;0\,;0\,;0\,;0] \end{cases} \tag{50}$$

The values are taken with respect to a Cartesian coordinate system centered on the target grasping fixture, so that the final position of the end-effector, in both the x and y coordinates, has always to converge to zero, i.e. $[x_{e.e}\,;\, y_{e.e}] = [0\,;0]$. Positions are measured in meters, while the unit measure chosen for the angles are the radians. The positions and orientations in (50) correspond to the configuration depicted on the right-side of figure 2, while velocity and acceleration vectors are null to have zero relative velocity of the end-effector with respect to the target. However, the choice for the final configuration is arbitrary within the physical constraints of the system.

The schematic of the control flow employed to test stability and robustness behaviors of the DS (49) is reported in figure 1.The tests have been carried on considering four scenarios, characterized by different initial ($\xi(t_0) = \xi^0$) and final ($\xi(t_f) = \xi^*$) positions of the chaser system. Velocity and acceleration vectors, for both initial and final states, are instead always taken as vectors of zeros for the reasons discussed in the previous chapters.





Figures 5a, 5b: time evolution of DS (49). In the first figure positions and orientations are depicted, while on the second one the velocity vector is considered. The initial and final state are respectively:

- $\xi^0 = \left[-2.89\ ;\ -2.89\ ;\ 0\ ;\ -3.19\ ;\ -2.89\ ;\ \frac{\pi}{2}\right]$.
- $\xi^* = [-0.89\ ;\ 0.-89\ ;\ \frac{\pi}{4}\ ;\ 0\ ;\ 0\ ;\ \frac{\pi}{30}]$.

1st case: initially both ξ^0 and ξ^* have been chosen to coincide with vectors belonging to the dataset used to create the model. Figures 5a and 5b shows the time evolution of the vector containing positions and orientations, and the vector containing the velocities, respectively. Since, in this case, the values that the state has to reach are represented by the system (50), it can be seen that the system converges to it.

A remark has to be done on both the amplitude of the initial oscillations of the state and the velocity of convergence. The presented work has been carried on with the aim of constructing a DS stable and robust in face of uncertainties on the boundary conditions. No type of assumptions was made to limit the amplitude of the state vector components or to fastener the convergence. Conversely, the dataset has been created minimizing the vector of control forces. In any case, with a proper tuning of the control vector both the aspects can be modified depending on the final needs of the user.

All the 27 different initial configurations used to create the dataset (figure 4) have been tested. The time evolution showed convergence to the desired state, similar to the case reported in figure 5a and 5b, in all the scenarios.

2nd case: for the second case it has been chosen to keep the final configuration as in (50), while vary the initial configuration, considering, this time, points out of the dataset. The results in terms of position, orientation, and velocity for the case

$$\xi^0 = \left[-2.49\ ;\ -2.49\ ;\ \frac{\pi}{6}\ ;\ -2.79\ ;\ -2.49\ ;\ \frac{\pi}{2}\right]$$

are reported in figures 6a and 6b. Again, the system shows good stability and converges to the desired state. For the amplitudes of the state's variations and for the convergence time hold the same considerations done for the previous case.

20 different out-of-dataset initial configurations has been tried; in all the cases the state vector converged to the wanted values. The results obtained prove that the system is robust in front of changes in the initial boundary conditions. However, to achieve good convergence behavior, the chosen ξ^0 must be close to a vector belonging to the dataset. In particular, the variation range of the initial state vector ξ^0 to stay within, in order to maintain good asymptotic stability of the DS, has been proven to be:

- For the positions:

$$\xi^0 = [-2.89 \pm 1.0\ ;\ -2.89 \pm 1.0\ ;\ -3.19 \pm 1.0\ ;\ -2.89 \pm 1.0\ ]\mathrm{m}$$

- For the orientations:

$$\xi^0 = \left[0 \pm \frac{\pi}{2}\ ;\ \frac{\pi}{2} \pm 0\ \right]\mathrm{rad}$$

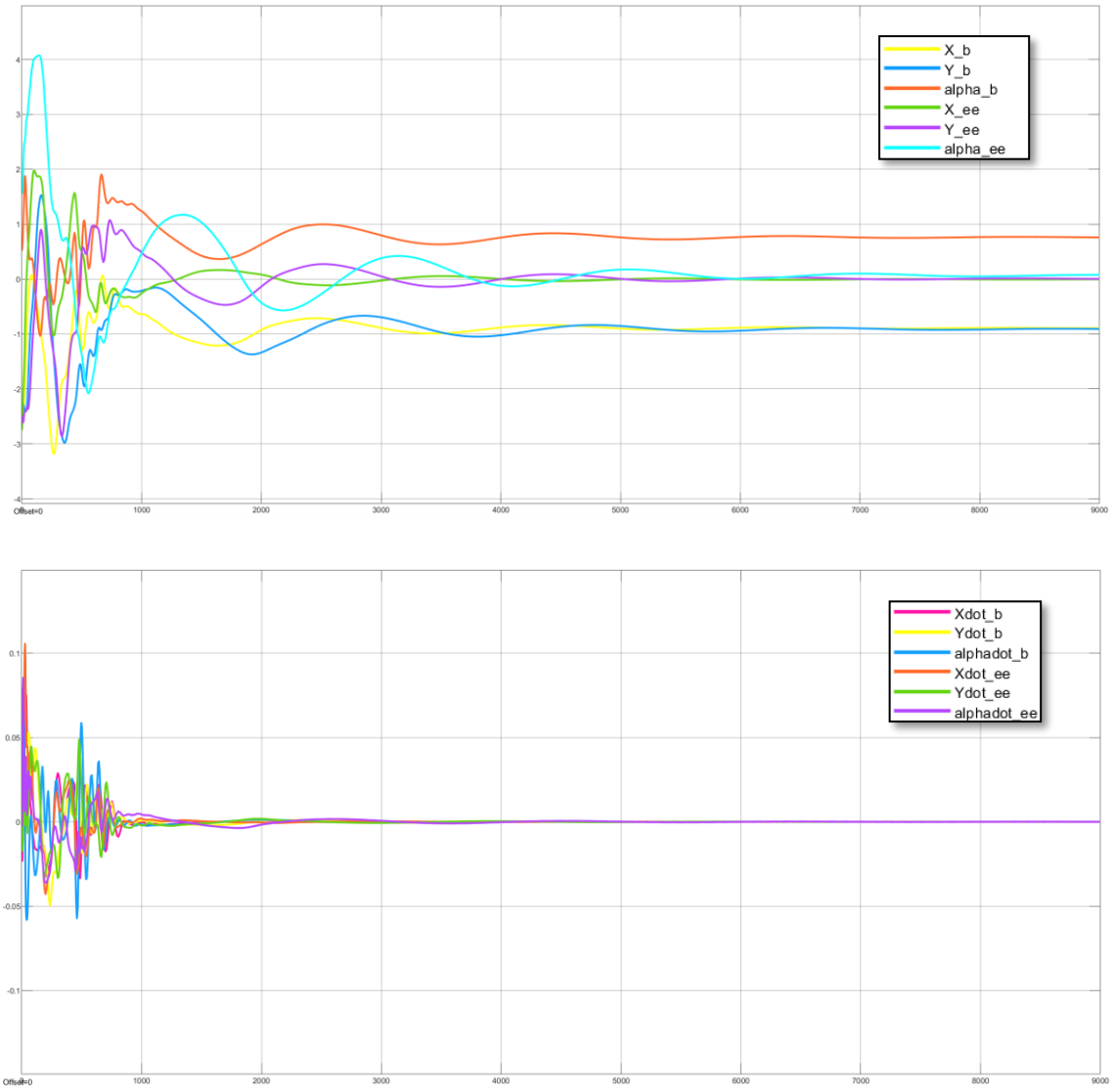


Figure 6a, 6b: ξ^* is taken from (50), while the initial configuration is:

- $\xi^0 = \left[-2.49\ ;\ -2.49\ ;\ \frac{\pi}{6}\ ;\ -2.79\ ;\ -2.49\ ;\ \frac{\pi}{2}\right]$.

3rd case: this case is conceptually the opposite of the previous case. In fact, the initial configuration is fixed and belongs to the dataset meanwhile ξ^* has been made to change. In figures 7a and 7b the time evolution for the case

$$\xi^* = [-1.27\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0]$$

is depicted. The showed behavior is similar to the one demonstrated for the previous cases, the states reach convergence and stabilize to the desired values. 20 different ξ^*

has been considered also for this scenario. The test proved that, if the variation limits are inside the range

$$\xi^* = \left[-0.89 \pm 1.0\ ;\ -0.89 \pm 1.0\ ;\ \frac{\pi}{4} \pm \frac{\pi}{2}\ ;\ 0\ ;\ 0\ ;\ \frac{\pi}{30} \pm \frac{\pi}{4}\right]$$

, the solutions always converge to the new desired state. The results obtained prove that the created DS is robust in front of changes in the final boundary conditions.

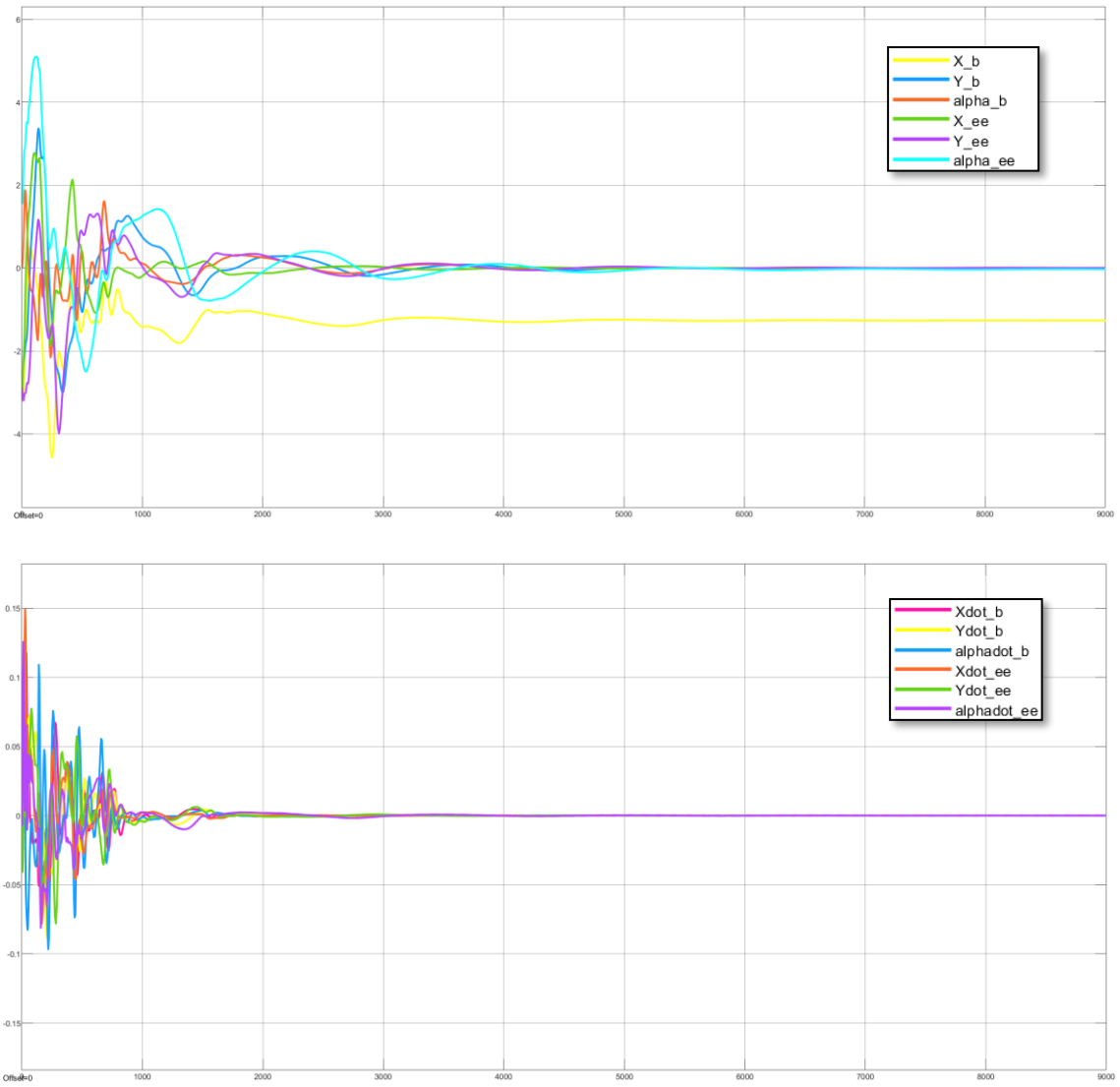


Figure 7a, 7b: the initial and final configurations are respectively:

- $\xi^0 = \left[-2.89\ ;\ -2.89\ ;\ 0\ ;\ -3.19\ ;\ -2.89\ ;\ \frac{\pi}{2}\right]$.
- $\xi^* = [-1.27\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0]$.

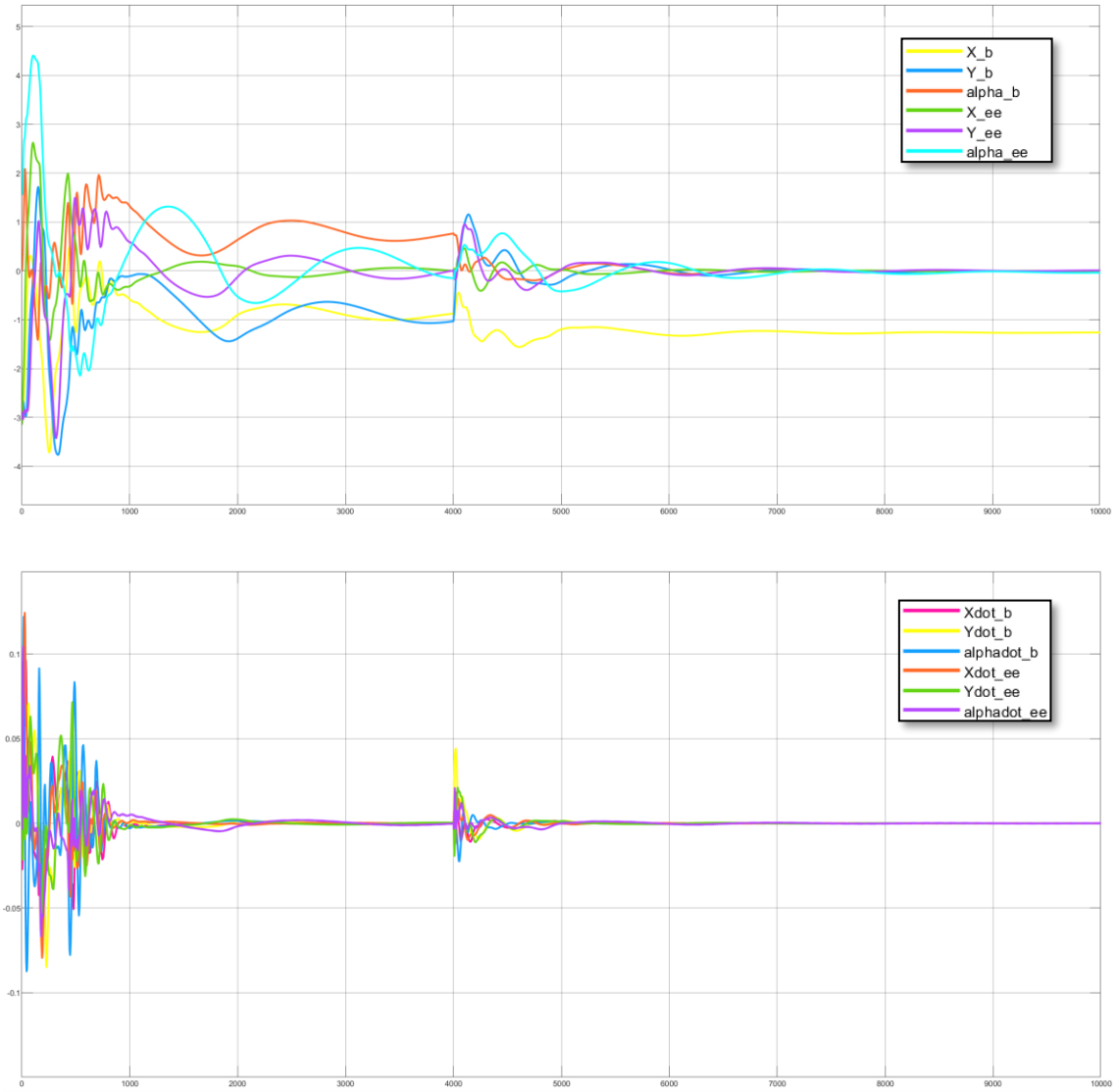


Figure 8a, 8b: the state vector has been chosen as:

- $\xi^0 = \left[-2.89\ ;\ -2.89\ ;\ 0\ ;\ -3.19\ ;\ -2.89\ ;\ \frac{\pi}{2}\right]$;
- $\xi^*(t < 4000) = \left[-0.89\ ;\ -0.89\ ;\ \frac{\pi}{4}\ ;\ 0\ ;\ 0\ ;\ \frac{\pi}{30}\right]$;
- $\xi^*(t \geq 4000) = [-1.27\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0]$.

4th case: as introduced in Chapter 2, a control algorithm based on a visual system mounted on the chaser spacecraft, continuously upgrades the final configuration ξ^* depending on the target unknown dynamics. For this reason, in this last case, starting from any initial configuration taken in the range described for case 2, the final position is made to change multiple times during the system motion.

Figures 8a and 8b shows the time evolution of the DS if ξ^* changes at $t = 4000s$, in particular:

$$\begin{cases} \xi^*(t < 4000) = \left[-0.89\ ;\ -0.89\ ;\ \frac{\pi}{4}\ ;\ 0\ ;\ 0\ ;\ \frac{\pi}{30}\right] \\ \xi^*(t \geq 4000) = [-1.27\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0\ ;\ 0] \end{cases}$$

It can be seen that, in correspondence of the upgrade, the states describe an oscillation, traceable to a re-set of the trajectory, to then converge to the desired solution.

The tests have been carried on upgrading ξ^* up to 5 times for each trajectory. The results are comforting since in all the proves the DS showed asymptotical stability and robustness in face of uncertainties in both the final and initial boundary conditions.

7. Conclusions

The capture of a tumbling Resident Space Object (RSO) by a spacecraft equipped with a robotic manipulator is expected to be a key maneuver in many future space missions. Given the increased number of degrees-of-freedom and the presence of non-convex constraints, successfully guiding such maneuver is a challenge.

The proposed approach is able to find a solution to the guidance problem by solving a DS in the form of an LPV system, whose parameters are approximated using GMMs. The successful demonstrations of this given guidance algorithm, during the proposed set of numerical simulations, open to the possibility of applying the proposed approach for onboard and real-time use. However, it has to be remembered that these results represent an “ideal” environment because of the multiple initial assumptions introduced in paragraph 2.1. For this reason, the numerical simulations provided may underestimates the amount of impulse required to complete the capture maneuver as well as modelling errors may make the chaser deviate from the nominal trajectory during real applications. Moreover, the control vector must be tuned to satisfies, apart from convergence and robustness properties, other important aspects such as the convergence velocity or additional constraints on the generated trajectory.

Despite these considerations the tests show consistent results and appear to indicate a certain degree of robustness of the created framework.

7.1 Future Works

In this paragraph are presented possible implementations for future works. In order to apply the given framework to real scenarios, the following points should be satisfied or, at least, considered:

- Surely, the procedure has to be extended to the 3D case. This implies the increment of the state vector dimension among with the increased complexity of the algorithms presented to construct the final DS. However, this extension should not involve too much the quality of the solutions since the main changes are related to the off-line procedure.

- The robotic arm may be composed by more links with the consequence of augmenting the redundancy of the system. The considerations related to this are similar to those done for the previous point.
- As previously mentioned, the control may be tuned to consider the proper velocity of convergence as well as to feasibly limit the components of the state vector.
- Additional constraints, like the keep-out-zones related to tumbling targets, should be considered during the dataset creation.
- If the variation range of the initial and final state vector wants to be increased, a possible solution is to increment the number of demonstrations to insert inside the dataset. This point coincide with an increased degree of robustness of the created DS.

Bibliography

1. R. Zappulla, J. Virgili-Llop, M. Romano (2017). “Convex Optimization for Proximity Maneuvering of a Spacecraft with a Robotic Manipulator”, AAS/AIAA Spaceflight Mechanics Meeting, San Antonio, TX, Feb. 6-9, 2017. (Advances in the Astronautical Sciences, Volume 160, pp 1059-1078).
2. J. Virgili-Llop, M. Romano (2017). “Laboratory Experiments on the Capture of a Tumbling Object by a Spacecraft-Manipulator System Using a Convex-Programming-Based Guidance”, Research Gate, Conference Paper.
3. S. S. M. Salehian, M. Khoramshahi, A. Billard (2018). “A Dynamical System Approach for Catching Softly a Flying Object: Theory and Experiment”, IEEE Transaction on Robotics. Volume 32, Issue 2. DOI: 10.1109/TRO.2016.2536749.
4. S. M. Khansari-Zadeh, A. Billard (2011). “Learning Stable Non-Linear Dynamical Systems with Gaussian Mixture Models”, IEEE Transaction on Robotics. Volume 27, Issue 5. DOI: 10.1109/TRO.2011.2159412.
5. G. Garofalo, C. Ott, A. Albu-Schäffer (2013). “On the Closed Form Computation of the Dynamic Matrices and their Differentiations”, IEEE/RSJ International Conference on Intelligent Robots and Systems. DOI: 10.1109/IROS.2013.6696688.
6. D. Cohn and z. Ghahramani, (1996). “Active Learning with Statistical Models,” Artificial intelligence research, vol. 4, pp. 129–145.
7. Aghili, F. (2008). “Optimal Control for Robotic Capturing and Passivation of a Tumbling Satellite with Unknown Dynamics,” in AIAA Guidance, Navigation, and Control Conference and Exhibit, vol. 21 (Honolulu, HI). DOI: 10.2514/6.2008-7274
8. S. Khansari-Zade, A. Billard, (2011). “The Derivatives of the SEDS Optimization Cost Function and Constraints with respect to the Learning Parameters”, Learning Algorithms and Systems Laboratory (Lasa), Technical Report.
9. Goodman, J. L. (2006). “History of Space Shuttle Rendezvous and Proximity Operations”, J. Spacecraft rockets 43, 944–959. Doi: 10.2514/1.19653.

10. Yoshida, K., Dimitrov, D., and Nakanishi, H. (2006). “On the Capture of Tumbling Satellite by a Space Robot,” IEEE/RSI International Conference on Intelligent Robots and Systems (Beijing), 4127–4132.
11. Flores-Abad, A., Zhang, l., Wei, Z., and Ma, O. (2016). “Optimal Capture of a Tumbling Object in Orbit Using a Space Manipulator”, J. Intell. Robot. Syst. 86:199. DOI: 10.1007/s10846-016-0417-1
12. Gasbarri, P., and Pisculli, A. (2015). “Dynamic/Control Interactions between Flexible Orbiting Space-Robot during Grasping, Docking and Post-Docking Maneuvers”, Acta Astronautical 110 (suppl. C): 225–238. DOI: 10.1016/j.actaastro.2015. 01.024