UNIVERSIDADE DE SÃO PAULO
ESCOLA POLITÉCNICA
POST-GRADUATION PROGRAM IN MECHANICAL ENGINEERING

GUILHERME PHILLIPS FURTADO

Formulation of Impedance Control Strategy as an Optimal Control Problem

São Paulo
2018

GUILHERME PHILLIPS FURTADO

Formulation of Impedance Control Strategy as an Optimal Control Problem

Revised version that includes the remarks and changes requested by the judging committee in September 6th, 2018. The original version can be found on the Library of Polytechnic School - USP, Department of Mechanical Engineering and on the Digital Library of Thesis and Dissertations of USP (BDTD), as defined by Resolution CoPGr 6018, from October 13th, 2011.

Supervisor: Prof. Dr. Arturo Forner-Cordero

São Paulo

2018

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Furtado, Guilherme Phillips
Formulation of impedance control strategy as an optimal control problem / G. P. Furtado -- versão corr. -- São Paulo, 2018.
106 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecânica.

1.Mecânica elétrica (Controle) 2.Equações de Riccati 3.Controle ótimo 4.Programação quadrática 5.Manipuladores redundantes I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecânica II.t.

Dissertation authored by Guilherme Phillips Furtado, under the title **"Formulation of Impedance Control Strategy as an Optimal Control Problem"**, presented to the Escola Politécnica of the University of São Paulo, to obtain the title of Master of Science from the Post-Graduation Program in Mechanical Engineering, in the area of concentration Control and Mechanical Automation Engineering, approved in September 6th, 2018, by the judging committee composed by:

Prof. Dr. Arturo Forner-Cordero

Institution: Escola Politécnica da Universidade de São Paulo

President

Prof. Dr. Bruno Augusto Angélico

Institution: Escola Politécnica da Universidade de São Paulo

Titular

Prof. Dr. Jacques Elisabeth Joseph Duysens

Institution: Katholieke Universiteit Leuven

Titular

Acknowledgements

I would like to express my gratitude to my advisor, Prof. Dr. Arturo Forner-Cordero, for the continued support during my Master study, in every aspect. This work would not have been possible without his contributions and assistance.

I would also like to thank Rafael Sanchez Souza, Carlos Noriega and Prof. Dr. Rafael Traldi Moura, for helping me with the execution of the experiments. I am also grateful to Rafael Sanchez Souza for the collaborative work that has been conducted during my studies.

Additionally, I would like to thank all members of the Laboratory of Biomechatronics, for all the stimulating discussions that we had.

Last but not the least, I would like to thank my family for supporting me, not only during the my studies, but also during my life in general.

Abstract

PHILLIPS FURTADO, Guilherme. **Formulation of Impedance Control Strategy as an Optimal Control Problem**. 2018. 106 p. Dissertation (Master of Science) – Polytechnic School, University of São Paulo, São Paulo, 2018.

A formulation of impedance control for redundant manipulators is developed as a particular case of an optimal control problem. This formulation allows the planning and design of an impedance controller that benefits from the stability and efficiency of an optimal controller. Moreover, to circumvent the high computational costs of computing an optimal controller, a sub-optimal feedback controller based on the state-dependent Ricatti equation (SDRE) approach is developed. This approach is then compared with the quadratic programming (QP) control formulation, commonly used to resolve redundancy of robotic manipulators. Numerical simulations of a redundant planar 4-DOF serial link manipulator show that the SDRE control formulation offers superior performance over the control strategy based QP, in terms of stability, performance and required control effort.

Keywords: Impedance control. Ricatti equations. Optimal control. Quadratic programming. Redundant manipulators.

Resumo

PHILLIPS FURTADO, Guilherme. **Formulação da Estratégia de Controle de Impedância como um Problema de Controle Ótimo**. 2018. 106 f. Dissertação (Mestrado em Ciências) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2018.

Uma formulação do controle de impedância para manipuladores redundantes é desenvolvida como um caso particular de um problema de controle ótimo. Essa formulação permite o planejamento e projeto de um controlador de impedância que se beneficia da estabilidade e eficiência de um controlador ótimo. Para evitar lidar com os elevados custos computacionais de se computar um controlador ótimo, um controlador em malha fechada sub-ótimo, baseado na abordagem das equações de Ricatti dependentes de estado (SDRE), é desenvolvido. Essa abordagem é comparada com a formulação de um controlador baseado em programação quadrática (QP), usualmente utilizado para resolver problemas de redundância em manipuladores robóticos. Simulações numéricas de um manipulador serial plano de quatro graus de liberdade mostram que o controlador baseado em SDRE oferece performance superior em relação a um controlador baseado em programação quadratica, em termos de estabilidade, performance e esforço de controle requerido do atuador.

Palavras-chaves: Impedância elétrica (controle). Equações de Ricatti. Controle ótimo. Programação quadrática. Manipuladores redundantes.

List of Figures

List of Tables

Contents

1 Introduction and Motivation

This work presents a controller formulation, based on optimal control theory, that aims to solve the impedance control problem of redundant robotic manipulators. First proposed by Hogan (1984), impedance control is a strategy that aims to regulate the dynamic behavior of a system as it interacts with the environment. It resulted from the combination of hybrid position and force control methods (MASON, 1981). Unlike pure force or pure position control, impedance control maintains a prescribed relationship between interaction forces and displacements resulting from this interaction. A well-designed dynamical behavior enables, effectively, simultaneous control of position and contact force of an end-effector, as it interacts with an unknown environment. By formulating impedance control as a particular case of an optimal control problem, it is possible to not only generate an efficient controller according to multiple, conflicting optimization criteria, but also handle mixed input/state constraints and, more importantly, assure global stability. Assuring stability, an open problem in the context of redundancy resolution, specially for an impedance controller, has been overcame in this work by implementing a sub-optimal impedance controller. Moreover, repeatability of the joint motion in the nullspace was achieved as an indirect consequence of asymptotic convergence to optimality. The results of this work have been submitted for publication (FURTADO; FORNER-CORDERO, 2018).

The impedance control strategy has been successfully applied in a wide range of engineering problems that involve intermittent contact between distinct systems. In teleoperation, for example, impedance control has been used both for designing the compliance of the operated robot and for enabling a human operator to control it, according to haptic feedback (TAFAZOLI *et al.*, 2002; OKAMURA, 2004; FERRAGUTI *et al.*, 2015). Another important application of impedance control is the simulation of virtual dynamical interactions. These types of control problems are often encountered in the design of interactive virtual environments with haptic interfaces (ADAMS; HANNAFORD, 1999) and healthcare assistive devices (AGUIRRE-OLLINGER *et al.*, 2007; TSAGARAKIS; CALDWELL, 2003). Impedance control can also be implemented in order to control the compliance of an autonomous robot, as it interacts with unknown environments. In this context, some examples include the generation stable legged locomotion, during ground

contact (AREVALO; GARCIA, 2012; PARK, 2001; SEMINI *et al.*, 2015), and soft grasping of objects (XU, 2013; SCHLEGL *et al.*, 2001).

Over the years, several implementations of impedance control were developed. In the simplest case, it can be designed as proportional-derivative (PD) position controller, in which both the position and velocity feedback generate an apparent impedance on the manipulator. The most general case involves replacing the original dynamics of the end-effector with the desired dynamics, often selected as a set linear spring-mass-damper system on the task-space (HOGAN, 1984), due to the ease of designing and inferring its characteristics. Anderson and Spong (1988) introduced a hybrid impedance control strategy, combining hybrid position/force control with impedance control, so position and force can be controlled while impedance is carefully monitored. To address the issue of uncertainty in the robot model and estimate the environment parameters, adaptive impedance control strategies were presented by Carelli and Kelly (1991) and Singh and Popa (1995). In the same context, Lu and Goldenberg (1995) proposed a combination of sliding-mode control and impedance control to design a robust impedance controller. More recently, He, Dong and Sun (2016) developed an adaptive impedance control employing neural networks, considering uncertainties and input saturation. We briefly present the formulation of impedance control with model-reference adaptive control approach in Appendix B, which is part of a work presented at IFAC BMS 2018 (SOUZA *et al.*, 2018).

The implementation of impedance control assumes that the relationship between task-space coordinates and joint-space coordinates is known beforehand, as impedance is defined with respect to the end-effector. The problem of finding the transformation between the task-space coordinates of the end-effector to joint-space coordinates is known as the inverse kinematics problem. If the manipulator is non-redundant, the solution is straightforward. On the other hand, in the presence of redundancies, there are infinite feasible solutions that map task-space coordinates to joint-space coordinates. Whereas the number of solutions offers adaptability and flexibility to the redundant manipulator, it comes at the cost of increasing mathematical complexity. In general, for a given application, the objective is to find the best solution that solves the inverse kinematics problem.

The usual approach to address redundancy resolution in real-time is by formulating the problem as a linearly constrained quadratic optimization problem. It consists of solving an under-determined linear system, while minimizing a quadratic-like objective function involving the system variables, and may also include inequality constraints (NAKAMURA;

HANAFUSA; YOSHIKAWA, 1987; NENCHEV, 1989; KHATIB, 1987; PETERS *et al.*, 2008; ZHANG *et al.*, 2017). The linear system is obtained from the kinematic relationship between the task-space and joint-space coordinates, defined by the Jacobian of the mapping between both spaces. The Jacobian acts as a linear transformation map, and redundancy resolution consists of finding the best inverse mapping for a set of given criteria. The criteria are formulated as objective functions to be minimized, where the quantity being measured is usually the instantaneous velocity norm, instantaneous torque norm or some deviation of the manipulator with respect to a certain distance. It can be interpreted as a point-wise optimization, in the sense that the future costs for controlling the robot are not taken into consideration. The control actions, either forces or torques, to be applied on the system, are a result of this optimization. If inequality constraints are not present and a single quantity is to be optimized, the inverse problem has an explicit solution, the weighted Moore-Penrose pseudoinverse. In addition, lower priority tasks can be included by projecting them on the nullspace of the linear transformation, at the cost of optimizing the original objective function. When redundancy resolution is treated as a quadratic optimization problem that includes inequality constraints or multiple objectives, it is formulated as a quadratic programming (QP) process (ZHANG *et al.*, 2017).

However, for some given predefined optimization criteria, undesirable phenomena can appear in the nullspace of the inverse linear transformation: self-motion, joint drifting during repeated tasks and joint instability, often due to the non-integrability of the inverse map (MUSSA-IVALDI; HOGAN, 1991), even in the absence of modelling errors. The pioneering works of Hollerbach and Suh (1987), Suh and Hollerbach (1987) that attempted to optimize local joint torques already noted the presence of instabilities in the nullspace. In 2002, O'NEIL proved that for mechanisms with one degree of redundancy, quadratic optimization of velocity, torques or acceleration vectors diverges to infinity in finite time for almost all possible resolution schemes, even with linear dissipation of nullspace velocities. More critically, the author showed that some conservative schemes that are not divergent, including the dynamically consistent redundancy scheme, also known as the inertia-weighted pseudo-inverse (CHANG; KHATIB, 1995), still suffer from a rapid rise of joint velocities, resulting in large joint accelerations and torques. The author concluded that even the more conservative redundancy resolution schemes can have undesired instabilities. In fact, control of the null-space is often treated as the major control problem of redundant robots (NEMEC; ZLAJPAH, 2000).

In order to address these shortcomings, there are a number of techniques entailing a trade-off between task performance and optimality of the optimization criteria. To cite a few, these techniques generally involve 1- introducing weighting matrices extrinsic to the original optimization (MUSSA-IVALDI; HOGAN, 1991); 2- introducing secondary tasks that do not emerge from the original optimization problem, but that are still projected on the nullspace of the manipulator through nullspace projection (SADEGHIAN *et al.*, 2014); 3- controlling the nullspace directly (ORIOLO, 1994; HU; GOLDENBERG, 1993) and 4- including additional criteria in the objective function (ZHANG *et al.*, 2017). For example, Peters *et al.* (2008) resorted to a PD scheme to stabilize the nullspace trajectories of a 7-DOF manipulator, where joint torques were being optimized, pulling the joints of the robot toward a preferred configuration. On the other hand, Zhang *et al.* (2017) observed that the minimum velocity norm criterion in a 6-DOF manipulator led to drifting problems, which were remedied by including a criterion that penalized the joint deviations from a defined position.

In the context of impedance control, the aforementioned problems are exacerbated when inertia shaping is present, e.g. the target inertia of the end-effector is different from the inertia of the virtual system (OTT; DIETRICH; ALBU-SCHÄFFER, 2015). While the QP formulation has been very successful over the years, generating effective control laws for a myriad of complex robots, the fact that even simple, important objective functions, such as local joint torque norm minimization or joint velocity norm minimization, may induce internal motion instability indicates the need for an alternative formulation, from the optimization perspective. The global performance of this method is limited by the local scope of the optimization, where each instant is treated separately. The minimization of the objective function, for some given kinematic map, is a point-wise optimization, also labeled *local* optimization (PETERS *et al.*, 2008), where the evolution of the system is completely disregarded, and so are future states. Since the local choice of joint torques at an instant will affect the future configuration of the robot, and thus the availability of future feasible solutions, the cumulative effect the local optimal actions on the system may deviate it from the optimum, for a given task, over the course of time.

In contrast with an optimal solution of an inverse kinematics problem, an optimal control solution (PONTRYAGIN, 1987; KALMAN, 1963; BELLMAN, 1957) optimizes an objective function that considers the cost of an action at some instant, and all successive actions, over the entire duration of the task execution. The optimization achieved is the

sum of a quantity over an entire time interval, for every instant, present and future. By nature of the optimization, undesirable phenomena arising from a cumulative choice of actions based on point-wise optimal choices (such as self-motion and instability), would be avoided. In fact, an optimal controller guarantees global stability. This can be easily verified by observing that the associated Bellman Cost function of the controlled system satisfies Lyapunov stability criterion (SALLE; LEFSCHETZ, 2012). Another advantage is to consider the optimization of global quantities over the entire duration of the task (e.g. total energy consumed and overall control effort). Due to the global scope, drifting is unlikely to occur, and repeatability is expected in the joint coordinate motion when the end-effector repeats closed trajectories, unless there exist multiple global optimal solutions of nullspace trajectories for a given problem. In the latter case, the issue could be easily avoided by defining a priority between solutions, through the manipulation of the objective function. The superior performance of an optimal control approach over a pointwise-optimization procedure has been observed by Suh and Hollerbach (1987), where a comparison between the solutions found by local joint optimization and an optimal controller showed that the latter outperformed the former. Optimal control theory has been successfully applied in a vast number of fields, from trajectory spacecraft optimization (ENRIGHT; CONWAY, 1991) to human gait prediction (ANDERSON; PANDY, 2001). Nevertheless, due the relatively high computational cost for finding an optimal control policy, the application of a point-wise optimization has often been justified in real-time applications, despite of its drawbacks.

Finding an optimal control action in real-time is often unfeasible, as it involves solving a large scale nonlinear problem (FAHROO; ROSS, 2000) or multi-point boundary value problems (PESCH, 1989a; PESCH, 1989c). However, if it is possible to use an approximate and optimal controller that retains some desirable traits of optimal controllers, such as local stability and local optimality, then it can be expected to get rid from nullspace instability and joint drifting. One class of controller that satisfies these requirements is obtained from the State-Dependent Ricatti Equation control approach (SDRE) (CIMEN, 2008). It is essentially a sub-optimal control strategy that requires the nonlinear dynamics and objective function to be described by state-dependent matrices, which are then used to create state-dependent Ricatti equations, to be solved in real time. Even though the SDRE approach lacks the global characteristics of an optimal control solution, as long as

the system starts sufficiently close to an optimal configuration, it should exhibit superior performance over a QP formulation.

Although the SDRE control strategy has been extensively used in the aerospace industry and other fields (CIMEN, 2012), its presence in robotics is less pronounced. In 2001, Erdem and Alleyne (2001) utilized SDRE control strategy to control 2-link underactuated pendular robot in real-time, where it outperformed the linear-quadratic regulator (LQR) control strategy. In 2009, Watanabe *et al.* (2009) utilized this strategy to control a three-link snake robot moving on a plane. More recently, Korayem *et al.* (2014) have applied the concept of SDRE to different types or robotic systems, for instance, to plan the motion of a 6-DOF cable-suspened robot considering obstacles. Korayem and Nekoo (2015a) also employed the finite time horizon formulation of the SDRE strategy to control a flexible manipulator when inequality constraints were involved. A general investigation for tracking problems was also provided (KORAYEM; NEKOO, 2015b). However, an SDRE control design that address problems where the feedback law is a function of external forces, as is the case of impedance control, has yet to be formulated, and this is another contribution of the present work.

Goals and Contribution

The currently growing research interest in application fields such as autonomous service robotics and health care assistive devices leads to an increasing demand for efficient manipulators with a load to weight ratio comparable to that of human arms. These manipulators should be able to perform compliant manipulation when interacting with an unknown environment, while and guaranteeing the safety of humans in physical contact with them. In that sense, an efficient feedback control laws that is not subject to internal instability and drifting is essential. Initially, the goal of this work was to model and understand human arm motion within an optimal control framework, with the intention of subsidizing the control design of health care assistive devices, where some preliminary results are presented in Appendix A. However, as the limitation of current redundancy resolution schemes became apparent, the goal of this work evolved.

The goal of this work is to present a novel formulation of impedance control for redundant manipulators as a particular case of an optimal control problem. The solution

of this problem yields a inherently stable, efficient behavior, with repeatable motions and configurations, thus addressing, simultaneously, all the major issues that negatively affect the performance of redundancy resolution schemes based on quadratic optimization.

The method is validated with a simulated serial link 4-DOF planar manipulator with a highly nonlinear, state-dependent, impedance, based on a mathematical model of a human arm. The main contributions are:

1. The formulation of impedance control as a particular case of an optimal control problem, addressing the problem of including external forces within the control law, despite the fact that they can be measured, but not predicted;
2. The solution of an impedance control problem where the dimension of the task space is different from the number of degrees of freedom of the desired behavior;
3. The presentation of a process to formulate state-dependent coefficient (SDC) matrices related to impedance control. These SDC matrices can be used to derive a sub-optimal closed-loop control law from the SDRE control approach;
4. The validation, via numerical simulations, that the SDRE control formulation offers superior performance over the QP formulation for redundancy resolution. It is shown that the SDRE formulation does not suffer either from instabilities or joint drifting, even when the system is disturbed. The SDRE formulation guarantees performance and stability without requiring direct control of the nullspace of the manipulator nor introduction of additional tasks and optimization criteria.

2 A Brief Overview of Optimal Control

From the regulation of physiological functions (JOSHI, 2002) to modeling economic growth (BROCK; SCHEINKMAN, 1976) to minimizing the fuel consumption of powertrains (PFIFFNER; GUZZELLA; ONDER, 2003), optimal control problems (OCPs) are of great interest in a myriad of fields. Solving an OCP consists into finding the best control action that moves a system from an initial condition to a terminal, desired condition. A brief overview of an OCP, solved via Pontryagin's minimum principle, is presented in this chapter. The discussion will be restricted to the OCP in Lagrange form:

$$I = \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), t)dt \tag{1}$$

Subject to the constraints

$$\dot{x}(t) = f(x(t), u(t), t), \tag{2}$$

$$x(t_0) = x_0, \tag{3}$$

$$\psi(x(t_f)) = 0, \tag{4}$$

$$h(x(t), u(t), t) \leq 0. \tag{5}$$

The necessary conditions that the optimal solution $x^*(t)$ and the optimal control policy $u^*(t)$ must satisfy are obtained via Pontryagin's minimum principle (PMP). First, the Hamiltonian is defined as

$$H(x(t), u(t), \lambda) = \mathcal{L}(x(t), u(t), t) + \lambda^T f(x(t), u(t), t) + \mu^T h(x(t), u(t)), \tag{6}$$

and an auxiliary function as

$$\Phi = \nu^T \psi. \tag{7}$$

With the PMP (PESCH, 1989b; PONTRYAGIN, 1987), the following adjoint equations are obtained

$$\dot{x}(t)^T = H_\lambda, \tag{8}$$

$$\dot{\lambda}(t)^T = H_x, \tag{9}$$

and the following conditions hold:

$$H_u = 0, \tag{10}$$

$$H_{uu} \geq 0, \tag{11}$$

$$x(t_k) = x_0, \tag{12}$$

$$\lambda(t_f)^T = \Phi_x|_{(t=t_f)}, \tag{13}$$

$$H|_{t_f} = 0, \tag{14}$$

$$\mu^T h(x(t), u(t), t) = 0. \tag{15}$$

The PMP converts the original OCP into a two-point boundary value problem (TPBVP). In the field of differential equations, a boundary value problem (BVP) is a differential equation together with a set of additional constraints, called the boundary conditions, arising in several branches of mathematics, physics and engineering (BURTON; MILLER, 1971; UNO; KAWATO; SUZUKI, 1989; PORSA; LIN; PANDY, 2016; HUNTINGTON; BENSON; RAO, 2007; GARCÍA-HERAS; SOLER; SÁEZ, 2014). Boundary value problems are similar to initial value problem (IVP), in the sense that they involve ordinary differential equations (ODEs). An IVP has all the conditions specified at the same value of the independent variable in the equation (known as the "initial condition") while a BVP, specifically, a TPBVP, has the conditions specified at the extremes of the independent variable in the equation (boundaries). Presently, it is much easier to solve numerically IVPs than TPBVPs; algorithms that solve IVPs are faster and convergence is generally not an issue. When dealing with BVPs, however, the current available algorithms, which are mainly based on shooting methods or collocation, are not only comparably slower, but requires a solution as an initial guess to the problem, and not only the rate of convergence will depend on the quality of the initial guess, but the convergence itself will also depend on it. Solving an OCP via PMP is referred as an *indirect method*. While it is relatively reliable for offline applications, solving a TPBVP in real time for a complex system is still a formidable challenge. For online applications, besides solving the TPBVP

in real time, one has the option of computing a set of optimal solutions for different initial and terminal conditions, store them and create a map $(x, x(t_f)) \mapsto u$ that can be accessed offline to estimate the optimal control action (ITO; SCHROETER, 2001; MARKMAN; KATZ, 2000; MARKMAN; KATZ, 2002). While it can yield satisfactory results, it is unfeasible for a system with large number of states and terminal conditions.

In light of the limitations presented by numerical methods addressing TPBVP, one can also look for *direct methods*. Unlike indirect methods, that are based in PMP, in a direct method, first the continuous states, control variables and objective function are approximated using an appropriate function approximation, (e.g. polynomial approximation). Then, the coefficients of the function approximations are treated as variables and the problem is transcribed to a nonlinear optimization problem. Thus, the system is first discretized, and then optimized. Direct Methods are often preferred over indirect methods due to easier convergence rate and computation. They have been successfully used both in offline applications and real-time applications (ROSS *et al.*, 2015). While still challengening for systems with large number of states, recent advances both in numerical methods and computation hardware indicate that soon online computation may be feasible for more complex systems.

If one cannot solve the original OCP using either direct or indirect methods sufficiently fast, it is also possible to use approximate methods, such as the Iterative Linear-Quadratic Regulator (TODOROV; LI, 2005), the State-Dependent Ricatti Equations (SDRE) (CIMEN, 2008) or a power series approximations of the value function of Hamilton-Jacobi-Bellman equation (BEELER; TRAN; BANKS, 2000). In this work, the optimal control problem is solved with the SDRE method, mainly due to its local stability, asymptotic convergence to the optimal solution and ease of computation.

3 Robot dynamics and Impedance control

An impedance control strategy defines a desired behavior of the robot end-effector, when it is under the effect of some measured external action, for every relevant degree of freedom needed to execute the task. The desired behavior, that can be regarded as a virtual system, is described by a system of ordinary differential equations (ODE). The physical system is also described by another set of ODEs. Three different spaces will be used: the virtual space $V_s \in \mathbb{R}^m$, the joint space $V_j \in \mathbb{R}^n$ and the task space $V_t \in \mathbb{R}^s$.

The behavior, in virtual coordinates, may be expressed as:

$$\ddot{y} = f_y(y, \dot{y}, f_m(t), \kappa(t)), \tag{16}$$

where t denotes the independent variable (i.e. time); $y \in \mathbb{R}^m$ denotes the virtual system position vector; $\dot{y} \in \mathbb{R}^m$ denotes the virtual system velocity vector; $f_m \in \mathbb{R}^m$ denotes measured external actions acting on the physical system, usually contact forces; and κ denotes the set of parameters that govern the system behavior, such as stiffness and equilibrium position. For readability, the time dependency of the system states y and $\dot{y}$ y and $\dot{y}$ is suppressed from now on.

The dynamics of the robot end-effector (i.e. the physical system), in joint coordinates, described in matrix form, are usually expressed as

$$D(q)\ddot{q} + G(q, \dot{q}) = \tau + J_p^T(q) f_m(t), \tag{17}$$

where $q \in \mathbb{R}^n$ denotes the joint space coordinates; $D(q) \in \mathbb{R}^{n \times n}$ denotes the symmetric positive definite inertia matrix; $G(q, \dot{q}) \in \mathbb{R}^n$ denotes a vector that includes dissipative, gravitational, inertial and state-dependent effects; $\tau \in \mathbb{R}^n$ denotes the control input at joint level; and $J^T(q) \in \mathbb{R}^{n \times m}$ denotes the transpose of the Jacobian that maps joint state velocities to task space velocities.

The relationship between the task space coordinates and the joint space coordinates is defined by the function p: $q \in \mathbb{R}^n \mapsto x_q \in \mathbb{R}^s$, where $s \leq n$, expressed as

$$p(q) = x_q, \tag{18}$$

where $x_q \in \mathbb{R}^s$ denotes the task space coordinates of the end-effector. Redundancy occurs when $s < n$.

The function $r\colon y \in \mathbb{R}^m \mapsto x_y \in \mathbb{R}^s$, that maps the relationship between the task space coordinates and the virtual system coordinates, is defined as

$$r(y) = x_y, \tag{19}$$

where $x_y \in \mathbb{R}^s$ denotes the task space coordinates of the virtual system.

In the majority of cases, the task space coordinates coincide with the virtual system coordinates, so equation (19) can be omitted. In this case, y can be expressed directly as a function of q, and so can $\dot{y}$ and $\ddot{y}$. However, in the more general case, there might not be a closed form expression of $y = r^{-1}(x_y)$. More importantly, if $s \neq m$, then (19) is non-invertible. It means that for some arbitrary x_y, either there will be multiple solutions for y, or no consistent solutions at all. As the latter is the case of the example provided in chapter 7, this notation will be preserved.

The necessary control inputs to impose the desired behavior to the manipulator end-effector, at joint level, are computed as a function of the joint space coordinates. They are obtained by imposing a motion constraint consisting of the equivalence between the task space coordinates of the end-effector and the task space coordinates of the virtual system, $x_q = x_y$. Ideal impedance control is realized when the physical states of the end-effector are equivalent to the virtual system states, when under the effect of the same external measured action $f_m(t)$, for every time instant. The constraint can be stated as

$$0 = r(y) - p(q). \tag{20}$$

An ideal second-order impedance controller can be represented by a set of under-determined differential-algebraic system of equations (DAE), expressed as:

$$\begin{cases} \ddot{q} = D(q)^{-1}(-G(q,\dot{q}) + \tau + J^T(q) f_m), \\ \ddot{y} = f_y(y(t), \dot{y}(t), f_m(t), \kappa(t)), \\ 0 = r(y) - p(q). \end{cases} \tag{21}$$

To find the control inputs τ, at joint level, necessary to impose the desired behavior of the end-effector, equation (20) is differentiated twice with respect to time, yielding

$$J_r(y)\dot{y} = J_p(q)\dot{q}, \tag{22}$$

and

$$\dot{J}_r(y)\dot{y} + J_r(y)\ddot{y} = \dot{J}_p(q)\dot{q} + J_p(q)\ddot{q}, \tag{23}$$

where $J_p(q) \in \mathbb{R}^{s \times n}$ and $J_r(y) \in \mathbb{R}^{s \times m}$ are the jacobians of $p(q)$ and $r(y)$, respectively.

If there exists an unique inverse mapping $r^{-1}(x_y)$, all y dependencies of (22) and (23) can be replaced by explicit expressions of q. Otherwise, (16) has to be solved by integration in order to find the values of y and $\dot{y}$.

Equation (21) can be solved for $J_p(q)D(q)^{-1}\tau$ with equation (22) and equation (23) by straightforward substitution, yielding

$$J_p(q)D(q)^{-1}\tau = J_p(q)D(q)^{-1}(G(q,\dot{q}) - J^T(q)f_m) + \dot{J}_r(y)\dot{y} + J_r(y)\ddot{y} - \dot{J}_p(q)\dot{q} \qquad (24)$$

If an inverse transformation $J_p(q)^+$ can be found, such as $J_p(q)^+ J_p(q) = I$, where I is the identity matrix of multiplication, then τ can be obtained by multiplying both sides of (24) by $D(q)J_p(q)^+$. However, as redundant manipulators have more degrees of freedom (n) than the dimension of the task-space (s), $J_p(q)$ is a rectangular, $s \times n$ matrix that lacks an uniquely defined inverse. Thus, while the mapping of joint velocities to task velocities may be unique, the mapping of task velocities to joint velocities is not. The lack of an unique inverse of the Jacobian is the reason why designing controllers for redundant manipulators is more complex than for kinematically determined ones: there are infinitely many combinations of joint coordinates pointing to the same task-coordinates.

It is not, however, necessary to always satisfy 20 everywhere to achieve impedance control. Asymptotic convergence toward satisfying it is often sufficient in practice. To illustrate it, the next chapter provides an example of impedance control being realized with an adaptative control strategy. The idea is to achieve an specified impedance despite lacking full knowledge of the model and system parameters. The example consists of a 1-DoF exoskeleton coupled with a human arm.

4 Redundancy Resolution via Quadratic Optimization

In the context of impedance control, if redundancy resolution would consist, generally, of solving

$$J_p(q)D(q)^{-1}\tau = J_p(q)D(q)^{-1}(G(q,\dot{q}) - J^T(q)f_m) + \dot{J}_r(y)\dot{y} + J_r(y)\ddot{y} - \dot{J}_p(q)\dot{q} \tag{25}$$

for τ, solving

$$J_p(q)\dot{q} = J_r(y)\dot{y} \tag{26}$$

for $\dot{q}$, or solving

$$J_p(q)\ddot{q} = \dot{J}_r(y)\dot{y} + J_r(y)\ddot{y} - \dot{J}_p(q)\dot{q} \tag{27}$$

for $\ddot{q}$. Which equation would be solved depends on the specific criteria adopted to define the "best" solution. For instance, torque optimization would generally employ equation (25), joint velocity minimization would employ equation (26) and joint acceleration minimization would employ equation (27).

In the more general context of redundant resolution, the problem is to find at least one solution of the linear system

$$Jb = c. \tag{28}$$

The general solution of the system, in the least-square sense, is given by:

$$b = J^+c + Nb_p, \tag{29}$$

where J^+ is a generalized inverse and N is the null space projector matrix

$$N = (I - J^+J), \tag{30}$$

so that $JNb_p = \vec{0}$ for every b_p. The product $Nb_p = \vec{0}$ corresponds to the homogeneous solution of the linear system. In robotics, if $b = \dot{q}$, then $Nb_p = \vec{0}$ is referred as the self-motion, which does not affect the coordinates of the end-effector in the task-space. It can be used to project additional tasks to improve the performance of the robot, at the expense of optimizing objective function, since $||b_p|| \geq 0$.

Given that equation (28) has an infinite number of solutions, it is desirable to choose the "best" one. More specifically, the solution of equation (28) that minimizes some quantity $\sigma_{qp}(b)$.

The inverse kinematics problem can be formulated as a quadratic optimization problem by defining

$$\sigma_{qp}(b) = b^T W b + d^T b, \tag{31}$$

where W is a positive definite square weighting matrix and d is a vector with the same dimension of b. This optimization problem can be solved numerically via a quadratic programming procedure (QP). If inequality constraints are not present and $d = \vec{0}$, then the problem has an explicit solution:

$$J^+ = W^{-1} J^T (J W^{-1} J^T)^{-1}. \tag{32}$$

In particular, if W is the identity matrix, then J^+ is the Moore-Penrose pseudo-inverse. The quantity b is usually a vector representing the states of the system, accelerations or control inputs. Thus, the evolution of the robot configuration will be defined by J^+. With this formulation, it is assumed that the constraint (28) is always satisfied. However, in practice, this is difficult to realize, given the presence of modeling errors and noise. As an alternative, the constraint can be inserted in the optimization function, leading to the quadratic optimization problem where the objective function where the following quantity should be minimized

$$\beta b^T W b + (Jb - c)^T (Jb - c). \tag{33}$$

The solution of the unconstrained optimization problem is given by

$$b = W^{-1} J^T (J W^{-1} J^T + \beta I)^{-1} c. \tag{34}$$

The above method is known as the damped least-square method, and generally produces a better performance for the system near Jacobian singularities. The lower the value of β, the more accurate the solution, at the expense of robustness near singularities.

From now on, without loss of generality, it is assumed that the objective is to minimize the norm of the joint torque. This is a conceptually important objective function, as it is related to the energy consumption of acting on the system and the size of the actuators. It is also a convenient choice to compare the QP and the SDRE formulations, because it is known to present instabilities. It is emphasized that the QP approach does not generally yield a solution to an optimal control problem. Otherwise, neither stability nor drifting phenomena would be observed. Rather, it yields an optimal solution to the

algebraic problem of finding an optimal instantaneous action with regard to some particular configuration (i.e. a point-wise optimization in time). Taking this into account, it is defined:

$$b = \tau, \tag{35}$$

$$\sigma_{qp}(\tau) = \tau^T \tau, \tag{36}$$

$$J = J_p(q) D(q)^{-1}, \tag{37}$$

and

$$c = J_p(q) D(q)^{-1} (G(q, \dot{q}) - J^T(q) f_m) + \dot{J}_r(y)\dot{y} + J_r(y)\ddot{y} - \dot{J}_p(q)\dot{q}. \tag{38}$$

The point-wise optimal torque is given by $J_p(q)^+ c$, which is known to yield an unstable system over the course of time (HOLLERBACH; SUH, 1987; SUH; HOLLERBACH, 1987). Instead of compensating the shortcomings of the QP formulation with ad-hoc solutions, such as secondary optimization criteria or introduction of PID controllers in the joint-space, the optimization could be the sum of k_t future values of τ between some interval $[t_0, t_{k_t}]$. If k_t is sufficiently large, then it is expected that $||\tau||$ will be bounded and the system will not be subject to instabilities. The torque can be normalized at every instant by weighting it with the ratio:

$$\Delta t = \frac{t_{k+1} - t_k}{k_t}. \tag{39}$$

In that case, the objective function would be:

$$\sum_{k=0}^{k_t} \tau(t_k)^T \tau(t_k) \Delta t. \tag{40}$$

Taking the limit of expression (40) for $\Delta t \to 0$ as $k_t \to \infty$, the objective function becomes

$$\int_{t_k}^{t_\infty} \tau^T \tau dt. \tag{41}$$

Optimization of expression (41) is possible by formulating the impedance control strategy as an optimal control problem.

5 Formlation of Impedance control as a Particular case of an Optimal Control Problem

An optimal control problem can be summarized as finding the best control policy over a specified domain, according to some performance index (i.e. objective function), for a dynamical system. Formally, the optimal control addressed here consists of minimizing a cost function of the form:

$$V(x,u) = \int_{t_0}^{\infty} \mathcal{L}(x(t), u(t), t)dt \tag{42}$$

subject to the constraints

$$\dot{x}(t) = f(x(t), u(t), t), \tag{43}$$

$$x(t_0) = x_0, \tag{44}$$

$$h(x(t), u(t)) \leq 0, \tag{45}$$

where $t \in [t_0, \infty) \subset \mathbb{R}$ denotes the independent variable; $x(t) : t \mapsto \mathbb{R}^{n_x}$ denotes the state vector of the system; $u(t) : t \mapsto \mathbb{R}^{n_u}$ denotes the control policy; $V : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ denotes the cost function in Lagrange form, with the function $\mathcal{L} : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ assumed to be positive and twice differentiable; function $h : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_h}$ generates the path constraints; function $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_x}$ represents the system dynamics and it is assumed to be Lipschitz continuous. If equation (42) is compared with equation (41), it becomes evident that when the constraints described by equations (43) and (45) are replaced by the DAE stated by equations (21), then impedance control becomes a particular case of an optimal control problem. To ease readability, all dependencies of time for all states and control inputs will be suppressed. The optimal control problem is often solved based on the necessary conditions of optimality from Pontryagin's minimum principle (PONTRYAGIN, 1987), rather than using the necessary and sufficient conditions from Bellman's principle of optimality (BELLMAN, 1957) and Hamilton-Jacob-Bellman (HJB) equations. This comes as a practical necessity, due to the complexity of solving HJB equations via dynamic programming. By applying Pontryagin's minimum principle, the original problem is converted to a two-point boundary value problem (TPBVP), which is easier to solve. However the solution is, in the strict sense, an extreme solution, rather than a global optimal solution. Nevertheless, it is usual to address the TPBVP solution as the optimal solution, and it will be addressed as such in this work.

Within optimal control theory, it is more convenient to represent the dynamics as a system of first-order differential equations, in state-space representation. Therefore, the system of second-order differential equations (21) is converted to a system of first-order differential equation. Defining for the physical system:

$$q = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}, \tag{46}$$

$$\dot{q} = \begin{bmatrix} x_{n+1} & x_{n+2} & \cdots & x_{2n} \end{bmatrix}, \tag{47}$$

where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ x_{n+1} \\ x_{n+2} \\ \cdots \\ x_{2n} \end{bmatrix}. \tag{48}$$

For the virtual system, it is defined:

$$y = \begin{bmatrix} z_1 & z_2 & \cdots & z_m \end{bmatrix}, \tag{49}$$

$$\dot{y} = \begin{bmatrix} z_{m+1} & z_{m+2} & \cdots & z_{2m} \end{bmatrix}, \tag{50}$$

where

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \\ z_{m+1} \\ z_{m+2} \\ \cdots \\ z_{2m} \end{bmatrix}. \tag{51}$$

The equations of motion are also generalized to a control-affine system. In that case,

$$\tau = P(x)u, \tag{52}$$

where $P(x)$ is a $n \times n_u$ non-singular transmission matrix and $u \in \mathbb{R}^{n_u}$ is the control vector. It is enforced that $n \leq n_u$, in order to avoid under-actuation.

The following selection matrices are introduced:

$$S(k) = \begin{bmatrix} I_k & 0_{k,k} \end{bmatrix}, \tag{53}$$

and

$$S^{\perp}(k) = \begin{bmatrix} 0_{k,k} & I_k \end{bmatrix}, \tag{54}$$

where I_k is a $k \times k$ multiplication identity matrix and $0_{k,k}$ is a $k \times k$ null matrix. Matrix $S(k)$ extracts the positions from a state vector with span of $2k$ and matrix $S^{\perp}(k)$ extracts the velocities of a state vector with span of $2k$.

The mappings given by equations (18) and (19) are converted to a nonlinear mapping $r \in \mathbb{R}^m \mapsto p \in \mathbb{R}^n$ representing equation (20), given by the equation:

$$r(S(m)z) = p(S(n)x). \tag{55}$$

The dynamical equations of the robot end-effector is given by:

$$\dot{x} = f(x, u, f_m(t)), \tag{56}$$

where

$$f(x, u, f_m(t)) = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \vdots \\ x_{2n} \\ f_p(x, u, f_m(t)), \end{bmatrix}, \tag{57}$$

and

$$f_p(x, u, f_m(t)) = D^{-1}(x)(J_p^T(x)f_m(t) + P(x)u - G(x)). \tag{58}$$

For the virtual system, it is defined that:

$$f_v(z, f_m(t), \kappa(t)) = \begin{bmatrix} z_{m+1} \\ z_{m+2} \\ \vdots \\ z_{2m} \\ f_y(z, f_m(t), \kappa(t)) \end{bmatrix}, \tag{59}$$

and

$$\dot{z} = f_v(z, f_m(t), \kappa(t)). \tag{60}$$

After collecting equations (56), (59) and (55), the optimal control problem can be formulated as

Minimize

$$V[u, x] = \int_{t_0}^{t_{k_t}} \mathcal{L}(x, u, t)dt \tag{61}$$

subject to

$$\dot{x} = f(x, u, f_m(t)), \tag{62}$$

$$\dot{z} = f_v(z, f_m(t), \kappa(t)), \tag{63}$$

$$x(t_0) = x_0, \tag{64}$$

$$p(S(n)x) - r(S(m)z) = 0, \tag{65}$$

$$z(t_0) = \begin{bmatrix} p(S(n)x_0) \\ J(x)S^{\perp}(n)x_0 \end{bmatrix}, \tag{66}$$

$$h(x, u) \leq 0. \tag{67}$$

Note that equation (56) must be solved together with equation (59). In order to find the optimal control policy, the entire structure of the virtual behavior must be known, and explicitly stated. That is because the optimal control policy must consider how the virtual system will evolve over the course of time. It is the main characteristic of the impedance control strategy being formulated as an optimal control problem: the configuration of the physical system at some specified time will be selected based on the anticipation of all future states it must track.

Inserting the virtual system states as constraints in an optimal control problem may be simple from the theoretical point of view. However, there are important practical drawbacks: it increases the difficulty of computing the solution; it reduces the space of feasible solutions; and, more importantly, due to modelling errors and disturbances, the constraints will very likely be violated. If the constraints cannot be *always* satisfied, it is not possible to guarantee neither a consistent numerical solution nor system stability.

If, instead, it is assume that the constraints can be violated in practice, it is expedient to include them as another criteria in the objective function. However, in order to avoid interference between satisfying the constraint function and optimizing the original function, the constraint function must have a significant large weight, which may lead to numerical ill-conditioning. Also, since the original constraint will no longer be always satisfied, for every t, it is also prudent to include higher derivatives of the constraint in the objective function, as the end-effector should be as close as possible to the intended behavior with respect to position, velocities and accelerations. To that end, a component of the cost function is defined as some metric $\sigma : (x, z, t) \mapsto [0, \infty)$, measuring a distance composed by the states of end-effector and the states of the virtual system, when under the effect of some action $f_m(t)$, during the time interval $[t_0, \infty)$. When that distance is minimized, the end-effector will be as close as possible to the desired behavior. Without loss of generality, the squared L^2-norm will be utilized as the metric function.

The jacobians of (55) and their derivatives with respect to time must be obtained. The Jacobian of p becomes

$$J_p(x) = \frac{\partial p}{\partial (S(n)x)}, \tag{68}$$

and the Jacobian of r becomes

$$J_r(z) = \frac{\partial r}{\partial (S(m)z)}. \tag{69}$$

Noticing that

$$S(n)\dot{x} = S^{\perp}(n)x, \tag{70}$$

and

$$S(m)\dot{z} = S^{\perp}(m)z, \tag{71}$$

the derivative of Jacobians (68) and (69) with respect to time are given by

$$\dot{J}_p(x) = \frac{\partial J_p(x)}{\partial (S(n)x)} S^{\perp}(n)x, \tag{72}$$

$$\dot{J}_r(z) = \frac{\partial J_r(z)}{\partial (S(m)z)} S^{\perp}(m)z, \tag{73}$$

The position error vector is defined as the function

$$\epsilon_0(x, z) = p(S(n)x) - r(S(n)z). \tag{74}$$

The first derivative of expression (74) with respect to time is the velocity error vector, given by

$$\epsilon_1(x,z) = J(x)S^{\perp}(n)x - J_r(z)S^{\perp}(m)z, \tag{75}$$

and the second derivative with respect to time is the acceleration error vector, given by

$$\epsilon_2(x,z,t) = \dot{J}(x)S^{\perp}(n)x + J(x)f_p(x,u,f_m(t)) - J_r(z)f_v(z,f_m(t),\kappa(t)) - \dot{J}_r(z)S^{\perp}(m)z. \tag{76}$$

If either ϵ_0, ϵ_1 or ϵ_2 equals $\vec{0}$ for every $t \geq t_0$, the task space behavior of the physical system (i. e. end-effector) becomes identical to the task space behavior of the virtual system. The distance metric σ is defined as

$$\sigma(x,z) = \lambda_0\epsilon_0(x,z)^T\epsilon_0(x,z) + \lambda_1\epsilon_1(x,z)^T\epsilon_1(x,z) + \lambda_2\epsilon_2(x,z,t)^T\epsilon_2(x,z,t) \tag{77}$$

where $[\lambda_0, \lambda_1, \lambda_2] \in \mathbb{R}^3_+$, leading to the optimal control problem

Minimize

$$V = \int_{t_0}^{t_f} \beta\,\mathcal{L}(x,u,t) + \sigma(x,z,t)dt \tag{78}$$

subject to

$$\begin{aligned}
&\dot{x} = f(x,u,t),\\
&\dot{z} = f_v(z,t),\\
&x(t_0) = x_0,\\
&z(t_0) = z_0,\\
&\beta > 0 \mid \beta \in \mathbb{R}\\
&h(x,u) \leq 0.
\end{aligned}$$

where β is the weight given for a secondary optimization criteria, e.g. control effort minimization, joint compliance, etc.. As long as $\|[\lambda_0, \lambda_1, \lambda_2]\| \gg \beta$, the behavior of the end-effector will be similar to the desired behavior. Notice that the initial condition $z(t_0)$ can be selected freely, as the end-effector is no longer constrained. If multiple tasks, with hierarchical relevance, are to be executed, it is possible to dismember $\mathcal{L}(x,u,t)$ into

successive sub-functions, each representing a sub-task, where the weights are selected according to task priority.

An optimal controller has the feature of taking advantage of known external actions to optimize the objective function, for the whole time interval considered. Therefore, the optimal solution is not only a function of the instantaneous value of $f_m(t_k)$ (at time t_k), but also of its future values. Evidently, the future time-history of f_m is not known, so in principle it is not possible to obtain an actual optimal solution. Luckily, the future history of f_m does not necessarily intervene with the minimization of $\sigma(x, z, t)$. Since f_m has an algebraic relationship with u, there always exists some u_c that can nullify the effect of f_m on the system. Under the assumption that the transmission matrix $P(x)$ is non-singular, the expression of this counter-action is given by

$$u_c = -P(x)^+ J_p^T(x) f_m(t). \tag{79}$$

That, of course, is only true if u does not saturate (HÄRKEGÅRD; GLAD, 2005).

Even if the minimization of $\sigma(x, z, t)$ is not influenced by the values of $f_m(t)$ for $t > t_k$, the same is not true for the minimization of $\mathcal{L}(x, u, t)$. Indeed, the internal motions of the redundant degrees of freedom at time t_k will depend on what it is assumed about the future of $f_m(t)$. An optimally controlled system (56) is not causal: the present values of x will depend on the future values of f_m. Therefore, in practice it is impossible to develop truly optimal impedance controller that can minimize some arbitrary function $\mathcal{L}(x, u, t)$, even when satisfying $\sigma(x, z, t) = 0$. Even so, an impedance controller can still be obtained by assuming that the future expectation of $f_m(t)$ is a constant equal to $f_m(t_k)$, at the present instant t_k, since the optimality of $\sigma(x, z, t)$ does not depend on the future values of $f_m(t)$.

So far, the formulation of impedance control as an optimal control problem presented here yields an open-loop solution for u. An open-loop controller cannot be used for applications that use impedance control, as the system is always under the effect of non-negligible external actions. A proper closed-loop optimal controller can be realized as a recursive open-loop optimal controller (ROSS *et al.*, 2015), extending the time horizon towards infinity (i.e. $t_f \to \infty$). The initial conditions $x(t_k)$ and $z(t_k)$ are set as current end-effector and virtual states, respectively, while initial time t_k is set as the current time. The problem is solved from t_k towards infinity, and the computed values of $u(t_k)$ are

applied on the system at time t_k. At time $t_{k+1} = t_k + \Delta t$, the current physical states $x(t_{k+1})$ and virtual states $z(t_{k+1})$ are set as the initial condition; t_{k+1} is set as the initial time; once again, the problem is solved, from t_{k+1} towards infinity.

If the physical system is autonomous, the controller can still achieve the task successfully. In this case, initial time can be forgone and the optimal control problem can be solved at every step from $t_k = 0$ towards infinity. In practice, a large value for the terminal time is sufficient to compute an accurate infinite-time horizon solution. If solving the optimal control problem in real-time is unfeasible, as it is often the case, offline computation can be carried out and stored; u may be approximated by a pre-computed look-up table (BEELER; TRAN; BANKS, 2000), a task that may require very long computations for a system with multiple degrees of freedom. In this work, the problem will be solved with the state-dependent Ricatti equation (SDRE) approach, presented in the next chapter, as it is a method that retains local asymptotic stability of an optimal control policy and asymptotically optimizes the objective function. It has the necessary characteristics to overcome the issues associated with the QP control formulation.

6 State-Dependent Ricatti Equations

The SDRE strategy, first proposed by Pearson (1962), provides an algorithm for solving nonlinear optimal control problems. The method entails the factorization of equations (56), (59) and (42) into the product of a matrix with state-dependent coefficients (SDC) and the state vector. This factorization brings the system to a (non-unique) linear structure having state-dependent coefficient (SDC) matrices, with the objective function described by a quadratic-like structure. An algebraic Riccati equation (ARE) using the SDC matrices is then solved on-line to give a sub-optimal control law. Cloutier (1997) shows that the SDRE feedback scheme for the infinite-time nonlinear optimal control problem, with control terms that appear affine in the dynamics and quadratic in the cost, is locally asymptotically stable. It is also shown that the Pontryagin necessary conditions for optimality are satisfied asymptotically by the algorithm. So, if the system starts at a position where $\sigma(x, z, t)$ is sufficiently close to zero, it can be expected that the SDRE approach will be able to solve the optimal control problem satisfactorily. The application of the SDRE strategy requires the system to be described with the following structure:

$$\dot{x}_{Ri} = F(x_{Ri}, u_{Ri}), \tag{80}$$

$$F(x_{Ri}, u_{Ri}) = A(x_{Ri})x_{Ri} + B(x_{Ri})U, \tag{81}$$

$$v(x_{Ri}, u_{Ri}) = x_{Ri}^T Q(x_{Ri})x_{Ri} + U^T R(x_{Ri})U + 2x_{Ri}^T N(x_{Ri})U, \tag{82}$$

where x_{Ri} denotes the state vector; u_{Ri} denotes the control vector; $A(x_{Ri})$ and $B(x_{Ri})$ are SDC matrices associated with the system equations; $v(x_{Ri}, u_{Ri})$ is the instantaneous cost-function, $Q(x_{Ri})$ is a positive definite weighting matrix and $R(x_{Ri})$ is a positive definite weighting matrix.

Additionally, the equations of the system must be factorized in a way that no singularities are generated in either $A(x_{Ri})$, $B(x_{Ri})$, $Q(x_{Ri})$ or $R(x_{Ri})$ (e.g. if, for a single state system, $f_k(x_k, u_k) = \cos x_k + u_k$, then $A_k(x_k) = \cos(x_k)/x_k$ is not a valid factorization when $x_k = 0$). Cimen (2012) provides some techniques to help design the SDC matrices, so that it is compatible with the SDRE framework. The SDC matrices are non-unique, and the performance of the controller will depend on the choice of parametrization. The

selection of the parametrization for a general system, and the analysis of how it affects the performance of the system, is outside the scope of this work.

If bias terms are present (i.e. $F(0,0) \neq 0$), they are factorized with the help of a newly-created state $\gamma \in \mathbb{R}$ satisfying

$$\dot{\gamma} = -\alpha\gamma, \tag{83}$$

and

$$\gamma(t_0) > 0 \tag{84}$$

where $\alpha \in \mathbb{R}^+$.

As an example, if there is a function such as

$$f(x_a) = \cos(x_a) + \sin(x_a) + x_a u_a, \tag{85}$$

it can be factorized as

$$f(x_a) = a_1 x_a + a_2 \gamma + b_a u_a, \tag{86}$$

where

$$a_1 = \sin(x_a)/x_a, \tag{87}$$

$$a_2 = \cos(x_a)/\gamma, \tag{88}$$

and

$$b_a = x_a. \tag{89}$$

The SDC matrices will be, in this trivial example,

$$A_a(x_{Ri_a}) = \begin{bmatrix} a_1 & a_2 \\ 0 & -\alpha \end{bmatrix}, \tag{90}$$

and

$$B_a(x_{Ri_a}) = \begin{bmatrix} b_a \\ 0 \end{bmatrix}, \tag{91}$$

with the state vector

$$x_{Ri_a} = \begin{bmatrix} x_a \\ \gamma \end{bmatrix}. \tag{92}$$

In order to define the appropriate weight matrices for the objective function, first an additional "error" state vector is defined, expressed as

$$e_0 = p(S(n)x) - r(S(m)z), \tag{93}$$

Then, an extended state vector is defined as

$$x_{Ri} = \begin{bmatrix} x \\ z \\ e_0 \\ \gamma \end{bmatrix}. \tag{94}$$

Differentiating expression (93) with respect to time, the following equation is obtained:

$$\dot{e}_0 = J(x)S^{\perp}(n)x - J_r(z)S^{\perp}(m)z. \tag{95}$$

Since, in order to solve ARE, the system must be stabilizable at every point, sham controls are created acting on the error state ($u_e \in \mathbb{R}^m$) and virtual system equations ($u_v \in \mathbb{R}^m$).

Defining:

$$E_0 = \begin{bmatrix} 0_{m,n} & J(x) & 0_{m,m} & -J_r(z) & 0_{m,m} & 0_{m,1} \end{bmatrix}, \tag{96}$$

the equations of the error-states will be given by

$$\dot{e}_0(x,u) = E_0 \begin{bmatrix} x \\ z \\ e_0 \\ \gamma \end{bmatrix} + \epsilon u_e, \tag{97}$$

where ϵ is a very small number that aids the numerical computation of the SDRE solution. With the appropriate value of ϵ and appropriate weight $r(x)$, the effect of the controls on the virtual system and error functions should be negligible. The extended control vector becomes:

$$u_{Ri} = \begin{bmatrix} u \\ u_v \\ u_e \end{bmatrix}. \tag{98}$$

The extended system equation is given by:

$$F(x_{Ri}, u_{Ri}) = \begin{bmatrix} f_p(x_{Ri}, u_{Ri}) \\ f_v(x_{Ri}) + \epsilon u_v \\ \dot{e}_0(x,u) \\ -\gamma \end{bmatrix}. \tag{99}$$

An SDC matrix $A(x_{Ri})$, specific to the system, is composed by the submatrices:

$$A(x_{Ri}) = \begin{bmatrix} A_x(x_{Ri}) & 0_{2n,2m} & 0_{n,m} & A_{x_\gamma}(x_{Ri}) \\ 0_{2m,2n} & A_z(x_{Ri}) & 0_{m,m} & A_{z_\gamma}(x_{Ri}) \\ & E_0 & & \\ 0_{1,2n} & 0_{1,2m} & 0_{1,m} & -\alpha \end{bmatrix}, \tag{100}$$

where $A_x(x_{Ri})$ is a SDC matrix including exclusively the physical system equations; $A_z(x_{Ri})$ is a SDC matrix including exclusively the virtual system equations; $A_{x_\gamma}(x_{Ri})$ and $A_{z_\gamma}(x_{Ri})$ contain all bias terms multiplied by $1/\gamma$. None of the sub-matrices contain ill-defined coefficient for $x_{Ri} = 0$.

Since the system is control-affine, the SDC matrix $B(x_{Ri})$ is a straightforward factorization with respect to U:

$$B(x_{Ri}) = \begin{bmatrix} D^{-1}(x_{Ri})P(x) & 0_{n,m} & 0_{n,m} \\ 0_{m,n} & \epsilon I_m & 0_{m,m} \\ 0_{m,n} & 0_{m,m} & \epsilon I_m \end{bmatrix}. \tag{101}$$

The cost function $\sigma(x, u)$, responsible for imposing the desired behavior, is defined as

$$\sigma(x_{Ri}, u_{Ri}) = x_{Ri}^T Q_{imp}(x_{Ri}) x_{Ri} + u_{Ri}^T R_{imp}(x_{Ri}) u_{Ri} + 2x_{Ri}^T N_{imp}(x_{Ri}) u_{Ri}, \tag{102}$$

where

$$Q_{imp}(x_{Ri}) = \lambda_0 Q_{e_0}(x_{Ri}) + \lambda_1 Q_{e_1}(x_{Ri}) + \lambda_2 Q_{e_2}(x_{Ri}), \tag{103}$$

$$N_{imp}(x_{Ri}) = \lambda_2 N_{e_2}, \tag{104}$$

$$R_{imp}(x_{Ri}) = \lambda_2 R_{e_2}. \tag{105}$$

Since the position error, e_0, is a system state, Q_{e_0} is defined as

$$Q_{e_0} = \begin{bmatrix} 0_{2n+2m,2n+2m} & 0_{2n+2m,m} & 0_{2n+2m,1} \\ 0_{m,2n+2m} & I_m & 0_{m,1} \\ 0_{1,2n+2m} & 0_{1,m} & 0 \end{bmatrix}. \tag{106}$$

Now, consider that

$$e_1 = J(x)S^\perp(n)x - J_r(z)S^\perp(m)z = E_0 x_{Ri}. \tag{107}$$

Therefore

$$e_1^T e_1 = x_{Ri}^T E_0^T E_0 x_{Ri}. \tag{108}$$

Thus, Q_{e_1} is given by

$$Q_{e_1} = E_0^T E_0. \tag{109}$$

For e_2, consider that

$$e_2 = \dot{E}_0 x_{Ri} + E_0 F. \tag{110}$$

After some manipulation, the following expression is obtained:

$$e_2^T e_2 = x_{Ri}^T Q_{e_2} x_{Ri} + 2x_{Ri}^T N_{e_2} u_{Ri} + u_{Ri}^T R_{e_2} u_{Ri}, \tag{111}$$

where

$$Q_{e_2} = \dot{E}_0^T \dot{E}_0 + \dot{E}_0^T E_0 A + A^T E_0^T \dot{E}_0 + A^T E_0^T E_0 A, \tag{112}$$

$$R_{e_2} = B^T E_0^T E_0 B, \tag{113}$$

$$N_{e_2} = \dot{E}_0^T E_0 B. \tag{114}$$

In the simulation provided in section 7, the following control-effort cost, to be optimized over the course of time, is considered:

$$\mathcal{L}(u_{Ri}) = u_{Ri}^T \begin{bmatrix} \beta I_n & 0_{n,2m} \\ 0_{2m,n} & \eta I_{2m} \end{bmatrix} u_{Ri}, \tag{115}$$

where β is a small, positive real number and η is a large, positive real number. As η becomes larger, optimizing the above expression becomes equivalent to optimize expression (36). In this work, it is set: $\eta = 1/\epsilon$. The instantaneous cost function is expressed as:

$$v(x_{Ri}, u_{Ri}) = \sigma(x_{Ri}, u_{Ri}) + \mathcal{L}(u_{Ri}). \tag{116}$$

The SDRE is expressed as

$$\begin{aligned} &A(x_{Ri})^T Z(x_{Ri}) + Z(x_{Ri}) A(x_{Ri}) - (Z(x_{Ri}) B(x_{Ri}) + \\ &N(x_{Ri})) R^{-1}(x_{Ri}) (B^T(x_{Ri}) Z(x_{Ri}) + N^T(x_{Ri})) + Q(x_{Ri}) = 0 \end{aligned} \tag{117}$$

The feedback gain that minimizes the expression

$$V(x_{Ri}, u_{Ri}) = \int_0^\infty v(x_{Ri}, u_{Ri}) dt \tag{118}$$

is

$$u_{Ri} = -K(x_{Ri})x_{Ri}, \tag{119}$$

where

$$K(x_{Ri}) = R^{-1}(x_{Ri})(B^T(x_{Ri})Z(x_{Ri}) + N^T(x_{Ri})), \tag{120}$$

and $Z(x_{Ri})$ is a stabilizing solution of equation (117), for a given x_{Ri}.

7 Implementation of QP and SDRE in a Redundant Manipulator

In order to compare the performance of both approach with a practical application, a redundant manipulator behaving as a simplified human arm model (BURDET; FRANKLIN; MILNER, 2013) was modelled and simulated under different conditions. The robot consisted of a planar 4-DOF serial link manipulator, as depicted on fig. 3b. The desired behavior of the robot end-effector is inspired on human arm motion. It consists of a virtual 2-DOF serial manipulator representing the shoulder-elbow joints (depicted on fig. 3a), with asymmetrical joint stiffness and damping, that varies according to external disturbances applied on the limbs. The wrist and hand are rigidly attached to the forearm, and which will be referred as the virtual end-effector. The virtual elbow and shoulder equilibrium positions, denoted as θ_{et} and θ_{st}, respectively, move in a cyclic fashion. The position profile of each virtual joint equilibrium position is depicted in fig. 1. The generation of the position profile of each joint angle is based on the experiments and numerical simulations presented in Appendix A. A sinusoidal force is applied on the robot end-effector, with a profile according to fig. 2. The robot end-effector translates the forces to the virtual end-effector, interfering with the arm motion. The resulting nonlinear impedance of the virtual human arm resists the interference. The robot end-effector task is to behave exactly like the virtual human arm, as it moves under the external forces (fig. 3c). The optimization criteria is to minimize the sum of squared joint torques and the sum of the squared error between the desired and the actual behavior. Additionally, an inertia-weighted torque optimization is also considered for the QP approach, as it is known to yield better than pure torque optimization.

The task space for both the robot end-effector and virtual end-effector is an horizontal Cartesian plane. The task-coordinates are the horizontal position and end-effector orientation, defined by $\begin{bmatrix} x_q & y_q & \phi_q \end{bmatrix}^T$ for the robot end-effector and $\begin{bmatrix} x_\theta & y_\theta & \phi_\theta \end{bmatrix}^T$ for the virtual end-effector.

The mapping $p(q) : q \mapsto \begin{bmatrix} x_q & y_q & \phi_q \end{bmatrix}^T$, between the joint coordinates and the task-space coordinates of the robot end-effector, is expressed as

$$p(q) = \begin{bmatrix} L_1 \cos(q_1) + L_2 \cos(q_2) + L_3 \cos(q_3) + L_4 \cos(q_4) \\ L_1 \sin(q_1) + L_2 \sin(q_2) + L_3 \sin(q_3) + L_4 \sin(q_4) \\ q_4 \end{bmatrix} \tag{121}$$

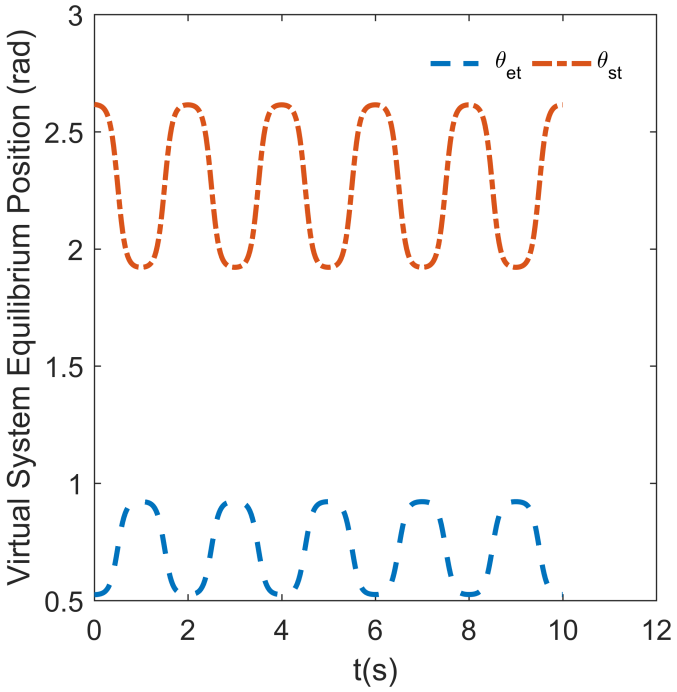


Figure 1 – *The equilibrium positions of the virtual shoulder and virtual elbow, over time.*

Source: *Guilherme Phillips Furtado and Arturo Forner-Cordero, 2018*

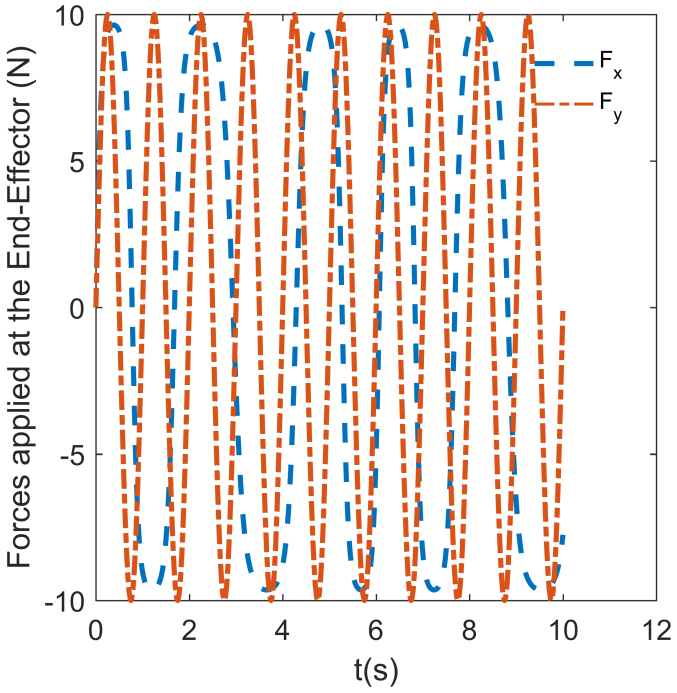
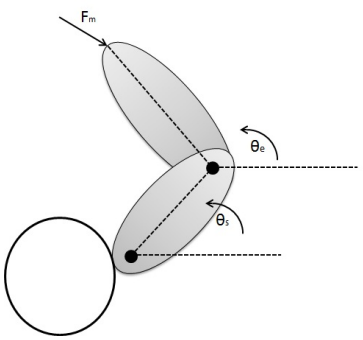
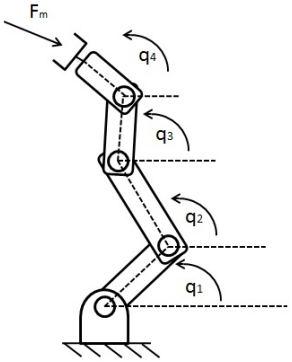


Figure 2 – *The external force applied at the robot end-effector, f_m, which is "transmitted" to the virtual end-effector. F_x and F_y correspond, respectively, to the horizontal and vertical force directions.*

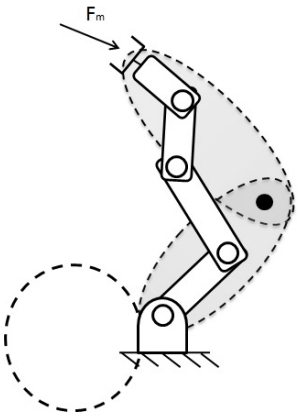
Source: *Guilherme Phillips Furtado and Arturo Forner-Cordero, 2018*



(a) *Illustration of the adopted 2-DOF human arm model.*



(b) *Illustration of a 4-DOF manipulator.*



(c) *Illustration indicating how the 2-DOF human arm model is constrained to the 4-DOF manipulator.*

Figure 3 – *An external force, f_m, is applied at the robot end-effector. This force is transferred to the virtual model that reacts according to its defined dynamics. This response is translated into motion of the robot end-effector. Since the robot end-effector is constrained to the virtual end-effector, both will present the same apparent impedance (or admittance) to the environment.*

Source: *Guilherme Phillips Furtado and Arturo Forner-Cordero, 2018*

where L_k and q_k denote, respectively, link lengths and joint coordinates for $k = 1, 2, 3, 4$ (fig. 3b)

The mapping $r(\theta) : \theta \mapsto \begin{bmatrix} x_\theta & y_\theta & \phi_\theta \end{bmatrix}^T$, between the virtual joint coordinates and virtual task-space coordinates of the virtual end-effector, is expressed as

$$r(\theta) = \begin{bmatrix} L_s \cos(\theta_s) + L_e \cos(\theta_e) \\ L_s \sin(\theta_s) + L_e \sin(\theta_e) \\ \theta_e \end{bmatrix}, \tag{122}$$

where L_s and L_e represent, respectively, upper and lower arm lengths, with θ_s and θ_e denoting, respectively, shoulder and elbow angles (fig. 3a).

Note that in the ideal case, both equations (121) and (122) map their arguments to the same point. In state-space representation, the virtual joint angles and velocities are denoted as

$$\begin{bmatrix} \theta_s \\ \theta_e \\ \dot{\theta}_s \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} \tag{123}$$

The dynamical equation of the virtual system is

$$\dot{z} = \begin{bmatrix} z_3 \\ z_4 \\ M_v^{-1}(-K_v \begin{bmatrix} z_1 - \theta_{s_t} \\ z_2 - \theta_{e_t} \end{bmatrix} - C_v \begin{bmatrix} z_3 - \dot{\theta}_{s_t} \\ z_4 - \dot{\theta}_{e_t} \end{bmatrix} - G_v + J_r^T f_m + \begin{bmatrix} \ddot{\theta}_{s_t} \\ \ddot{\theta}_{e_t} \end{bmatrix}) \end{bmatrix}, \tag{124}$$

where

$$M_v = \begin{bmatrix} 0.263 & 0.120\cos(z_2 - z_1) \\ 0.120\cos(z_2 - z_1) & 0.134 \end{bmatrix} \tag{125}$$

$$K_v = \begin{bmatrix} 10.8 + 3.18|\tau_s| & 2.83 + 2.15|\tau_e| \\ 2.51 + 2.34|\tau_e| & 8.67 + 6.18|\tau_e| \end{bmatrix} \tag{126}$$

$$C_v = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \tag{127}$$

$$c_{11} = 0.52\sqrt{2.84 + 0.836|\tau_s|}, \tag{128}$$

$$c_{12} = 0.52\sqrt{\cos(z_2 - z_1)(0.334 + 0.258|\tau_e|)}, \tag{129}$$

$$c_{21} = 0.52\sqrt{\cos(z_2 - z_1)(0.301 + 0.281|\tau_e|)}, \tag{130}$$

$$c_{22} = 0.52\sqrt{1.162 + 0.828|\tau_e|}. \tag{131}$$

$$\begin{bmatrix} \tau_s \\ \tau_e \end{bmatrix} = J_r^T f_m \tag{132}$$

and

$$G_v = \begin{bmatrix} 0.1328\sin(\theta_s - \theta_e)\dot{\theta}_e^2 \\ -0.1328\sin(\theta_s - \theta_e)\dot{\theta}_s^2 \end{bmatrix}. \tag{133}$$

The physical system is a 4-DOF ideal serial manipulator with diagonal moment of inertia tensor and negligible joint stiffness. All parameters used in the simulations are presented on table 1.

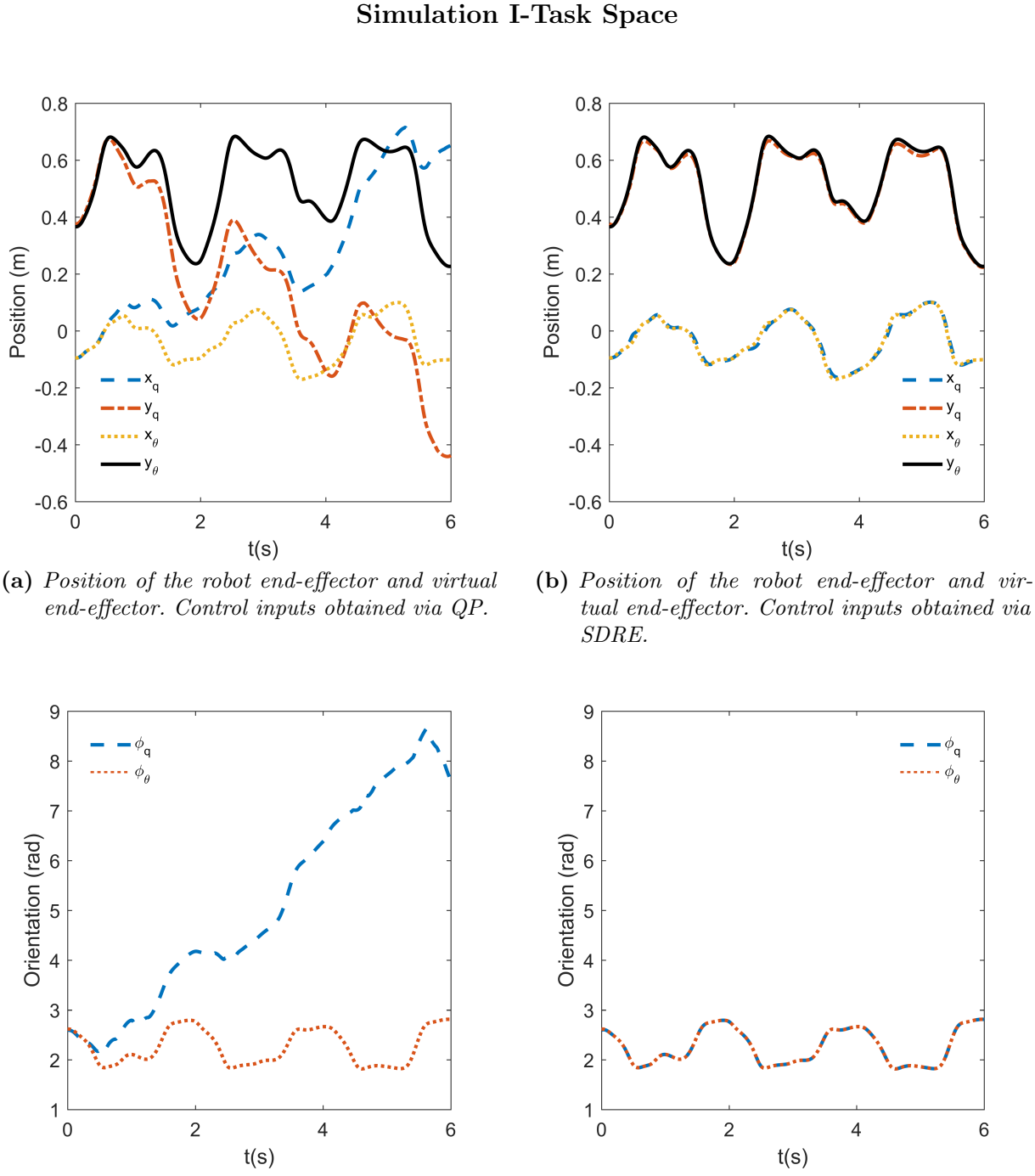
Table 1 – *Parameter Values of the Human Arm Model and 4-DoF Serial Manipulator*

Parameter	Value	unit
$[\lambda_0\ \lambda_1\lambda_2]$	[500 1 0]	
ϵ	0.001	
α	0.001	
β	0.05	
$[L_1\ L_2\ L_3\ L_4]$	[0.25 0.25 0.34 0.08]	m
$[L_s\ L_e]$	[0.31 0.42]	m
Linear friction at robot joints	0.4	Ns/m
Mass of L_1	0.2	kg
Mass of L_2	0.2	kg
Mass of L_3	0.272	kg
Mass of L_4	0.064	kg
Moment of inertia of L_1	0.0031	kgm^2
Moment of inertia of L_2	0.0031	kgm^2
Moment of inertia of L_3	0.0079	kgm^2
Moment of inertia of L_4	1.0240e-04	kgm^2

7.1 Results: Comparison Between QP and SDRE approaches

Four simulations were carried out in order to compare the performance between QP formulation and SDRE formulation. They used MATLAB (The Mathworks, Inc.,

Natwick, MA, USA) running in a computer with an Intel i7-6700HQ processor at 2.60GHz clock rate. A fixed step 4th-order Runge-Kutta algorithm was used to find the numerical solutions of equations (17) and (124), with a stepsize of 10^{-4}. The solution of equation (117) is found with the MATLAB function *care()*. The gain matrix $K(x_{Ri})$ was updated every 10^{-2} s. The solution of equation (117) is found with the MATLAB function *care()*. Simulation I (fig. 8 and 5) compares the performance between the QP formulation and the SDRE formulation. Simulation II (fig. 6) shows the performance the QP formulation with an inertia-weighted torque optimization. Simulation III (fig. 7) presents the performance of the SDRE formulation when the system starts outside the desired behavior, under random uniformly distributed torque disturbance of $0.1Nm$. Simulation IV (fig. 8) aims to investigate the repeatability of joint motions, when the motion in the task space is cyclic.



(a) *Position of the robot end-effector and virtual end-effector. Control inputs obtained via QP.*

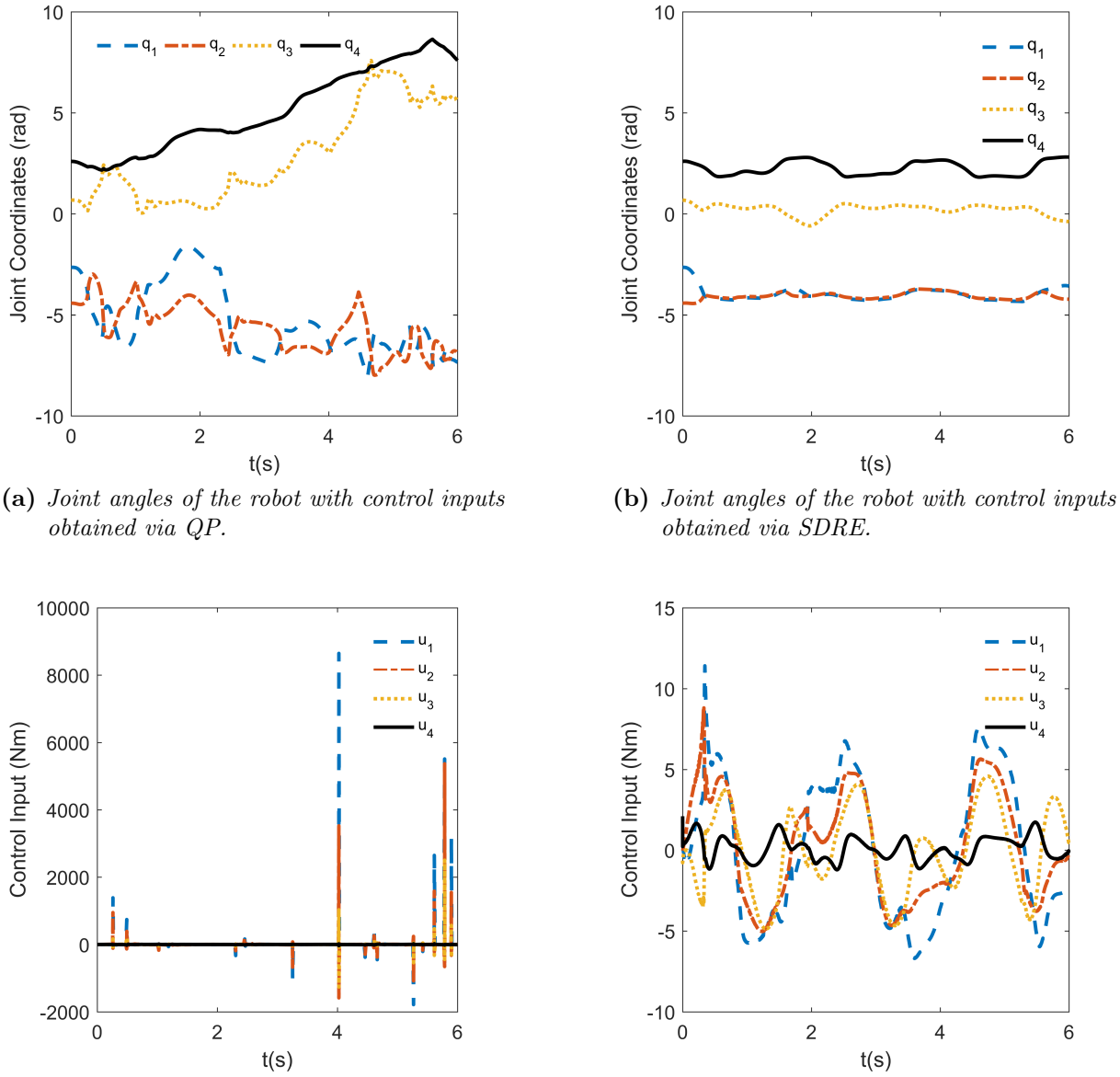
(b) *Position of the robot end-effector and virtual end-effector. Control inputs obtained via SDRE.*

(c) *Orientation of the robot end-effector and virtual end-effector. Control inputs obtained via QP.*

(d) *Orientation of the robot end-effector and virtual end-effector. Control inputs obtained via SDRE.*

Figure 4 – *Simulation I: Comparison between performances of the system with the QP (a and c) and SDRE (b and d) strategies. The robot starts at* $q(0) = [-2.65 \ -4.42 \ 0.68 \ 2.60]^T$, *and the virtual system starts at* $\theta(0) = [0.52 \ 2.62]^T$. *Under QP formulation, the system demonstrates instability, which is not observed when the controls are obtained via the SDRE approach.*

Source: *Guilherme Phillips Furtado and Arturo Forner-Cordero, 2018*



(a) *Joint angles of the robot with control inputs obtained via QP.*

(b) *Joint angles of the robot with control inputs obtained via SDRE.*

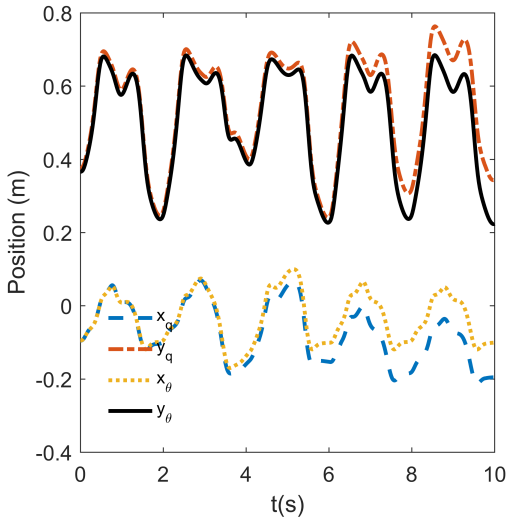
(c) *Control inputs computed via QP formulation.*

(d) *Control inputs computed via SDRE formulation.*

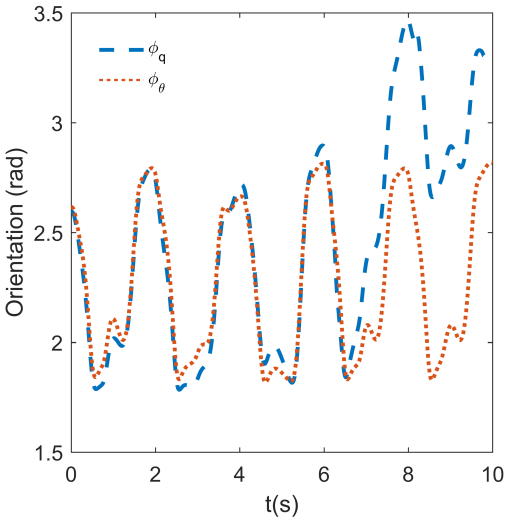
Figure 5 – *Simulation I: Comparison between performances of the system with the QP (a and c) and SDRE (b and d) strategies. The robot starts at* $q(0) = [-2.65 \ -4.42 \ 0.68 \ 2.60]^T$, *and the virtual system starts at* $\theta(0) = [0.52 \ 2.62]^T$. *When the controls are obtained via QP formulation, the joint angles drift away, and very high torque norms are observed. On the other hand, when the system is controlled via SDRE formulation, the joint angles and torque norms remain bounded.*

Source: *Guilherme Phillips Furtado and Arturo Forner-Cordero, 2018*

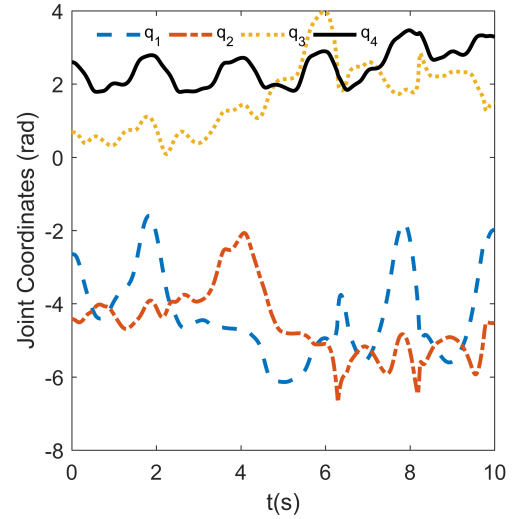
Simulation II



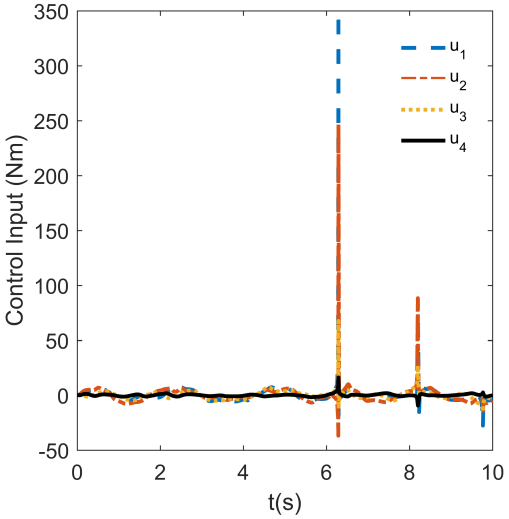
(a) *Position of the robot end-effector and virtual end-effector. QP, with inertia-weighted torque optimization.*



(b) *Orientation of the robot end-effector and virtual end-effector. QP, with inertia-weighted torque optimization.*



(c) *Joint angles of the robot with control inputs obtained via QP, with inertia-weighted torque optimization.*

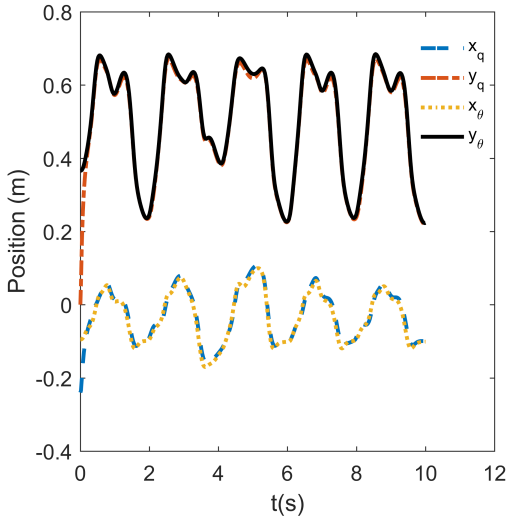


(d) *Control inputs computed via QP formulation, with inertia-weighted torque optimization.*

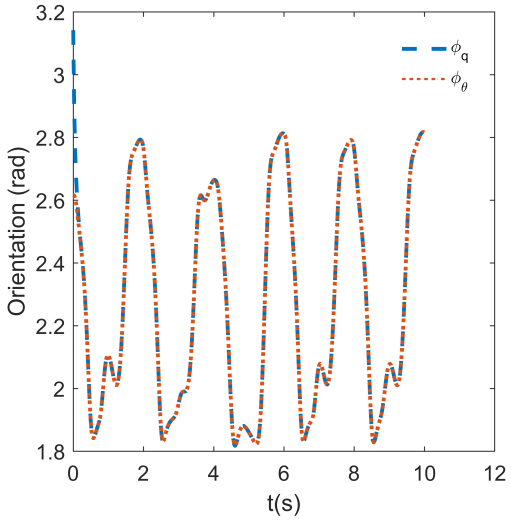
Figure 6 – *Simulation II: Behavior of the system with QP formulation, with inertia-weighted torque optimization criterium. Initial conditions are identical to Simulation I.*

Source: *Guilherme Phillips Furtado and Arturo Forner-Cordero, 2018*

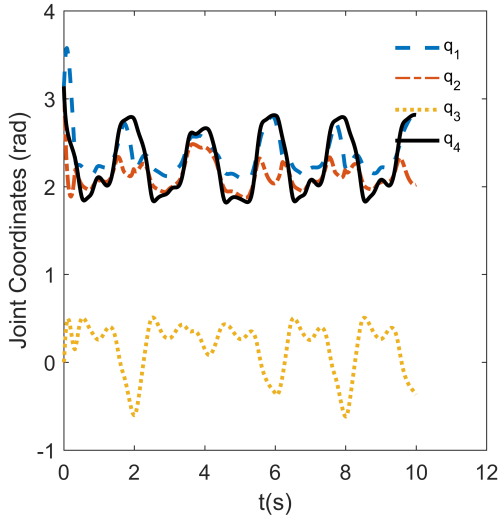
Simulation III



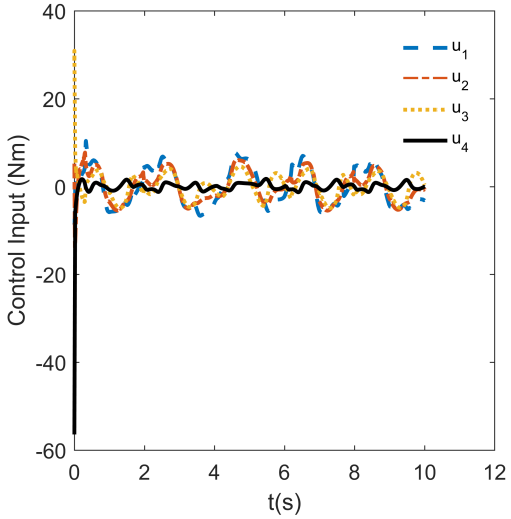
(a) *Position of the robot end-effector and virtual end-effector, when the robot starts far from an optimal initial condition and its joints are disturbed, under SDRE formulation.*



(b) *Orientation of the robot end-effector and virtual end-effector, when the robot starts far from an optimal initial condition and its joints are disturbed, under SDRE formulation.*



(c) *Joint angles of the robot end-effector, when it starts far from an optimal initial condition and its joints are disturbed, under SDRE formulation.*

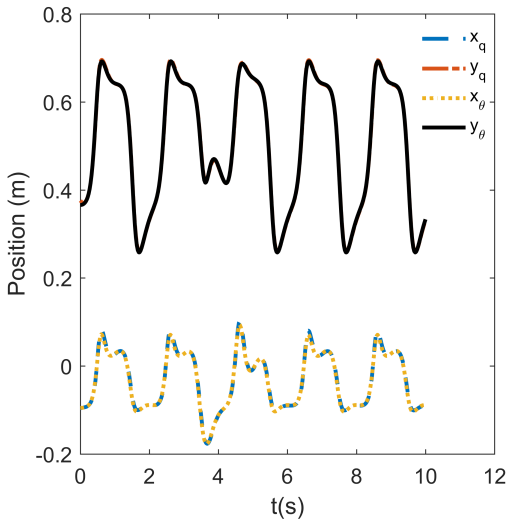


(d) *Computed control inputs via SDRE formulation, when the robot is disturbed and far from an optimal initial condition.*

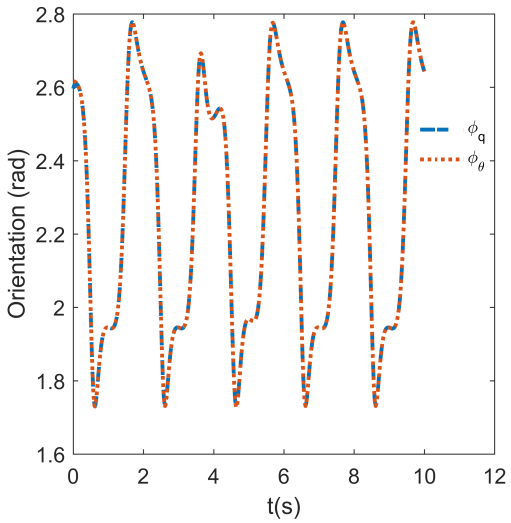
Figure 7 – *Simulation III: Behavior of the system under perturbations, with initial conditions away from the optimal* ($q(0) = [\pi\ \pi\ 0\ \pi]^T$ *and* $\theta(0) = [0.52\ 2.62]^T$*). The joints are disturbed with a random uniformly distributed torque of amplitude* $0.1Nm$ *and zero mean.*

Source: *Guilherme Phillips Furtado and Arturo Forner-Cordero, 2018*

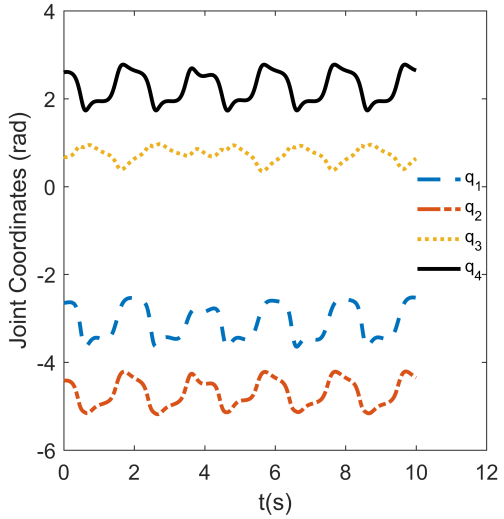
Simulation IV



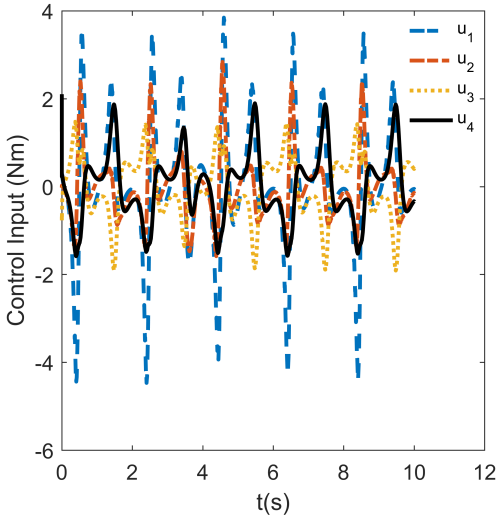
(a) *Position of the robot end-effector and virtual end-effector, when no external forces or disturbances are applied, under SDRE formulation.*



(b) *Orientation of the robot end-effector and virtual end-effector, when no external forces or disturbances are applied, under SDRE formulation.*



(c) *Joint angles of the robot end-effector, when no external forces or disturbances are applied, under SDRE formulation.*



(d) *Computed control inputs when no external forces or disturbances are applied, under SDRE formulation.*

Figure 8 – *Simulation IV: Behavior of the system when when no external forces or disturbances are applied with initial conditions:* $q(0) = [-2.65 \;\; -4.42 \;\; 0.68 \;\; 2.60]^T$ *and* $\theta(0) = [0.52 \;\; 2.62]^T$*. The robot executes repeatable motions on the joint space, even without considering a repeatability criterion in the objective function.*

Source: *Guilherme Phillips Furtado and Arturo Forner-Cordero, 2018*

7.2 Discussions

The first simulation compared the method proposed in this work, based on the SDRE, with the more frequently used quadratic programming (QP) approach. In figures 4 and 5, the comparison between system performance with the QP (a and c) and SDRE (b and d) formulation strategies is presented. The robot starts at $q(0) = [-2.65\ -4.42\ 0.68\ 2.60]^T$, and the virtual system starts at $\theta(0) = [0.52\ 2.62]^T$. Under QP formulation, the system demonstrates instability, which is not observed when the controls are obtained via the SDRE approach. While in the SDRE approach the virtual and robot end-effector positions follow the same paths, in the QP approach they diverge. Moreover, the SDRE joint angles and torques (control inputs) are bounded while in the QP approach the joint angles drift and the torques have very large peaks (fig. 5).

The instability of joint torque optimization with QP control formulation is a well-known problem. In order to alleviate it, a dynamically consistent pseudo-inverse can be used instead of the Moore-Penrose pseudo-inverse. It is equivalent to a inertia-weighted Moore-Penrose pseudoinverse, where $W = D^{-1}(q)$. The second simulation (fig. 6) aimed to provide the results of the inertia-weighted torque optimization. Despite not being the original optimization criterium, it offers better performance than pure torque optimization, in the context of QP. However, it still presented very high peak torques, and slowly diverged from the desired behavior. It also did not display repeatability, as it can be seen in figure 6.

The third simulation was performed to illustrate the behavior of the SDRE approach when it starts far from the optimal initial condition ($q(0) = [\pi\ \pi\ 0\ \pi]^T$ and $\theta(0) = [0.52\ 2.62]^T$). In addition, the joints are disturbed by a random uniformly distributed torque of amplitude $0.1Nm$ and zero mean. The robot remained stable during the entire duration of the simulation, and no drifting was observed, as shown in figure 7. These results shows a starking contrast with the QP formulation. Stability was assured, even in the presence of significant disturbances, without requiring any specific nullspace control. It can be said that nullspace control, a major control problem of redundant robots, is completely unnecessary with the SDRE formulation. Note that these results were obtained with nonlinear inertia-shaping of the robot. In essence, the SDRE formulation *trivializes* redundancy resolution.

The fourth simulation is aimed at validating the behavior of the SDRE during the execution of repetitive motions. In this simulation, no external forces or disturbances were applied (Initial conditions: $q(0) = [-2.65 \ -4.42 \ 0.68 \ 2.60]^T$ and $\theta(0) = [0.52 \ 2.62]^T$). The robot executes repeatable motions in the joint space, even without considering a repeatability criterion in the objective function. This is equivalent to an infinite-horizon state-tracking problem, as the desired trajectory of the end-effector is predefined, with no external forces acting on the virtual system. Numerous works have emerged over the years developing methods capable of imposing repeatable joint motions with the QP formulation (MUSSA-IVALDI; HOGAN, 1991; ROBERTS; MACIEJEWSKI, 1993; ROBERTS; MACIEJEWSKI, 1994; KLEIN; AHMED, 1995; BOWLING; HARMEYER, 2010; ORIOLO; CEFALO; VENDITTELLI, 2017). Unlike with the SDRE formulation, repeatability in QP formulation can only be achieved by exploiting the structure of the dynamic equations of the system or imposing additional optimization criteria. Here, once again, it can be seen that the SDRE formulation completely overshadows the QP formulation.

Due to the high computational costs of obtaining an optimal control policy in real time, obtaining an approximate optimal control policy via SDRE formulation was opted. In comparison with the QP formulation, the SDRE approach requires the additional computation of four weighting matrices and appropriate factorization of the system dynamics. During the simulations, the evaluation of the gain matrix $K(x_{Ri})$ took, on average, 5×10^{-3}s. However, the gain matrix can be updated at a slower rate than the input computations ($u = -K(x_{Ri})x_{Ri}$). In the simulations, the gain matrix was updated every 10^{-2} s, while input computation was done at every time step (10^{-4}). This setup can immediately benefit from a parallel computation scheme, as $K(x_{Ri})$ is computed independently, and at a slower rate, than $u = -K(x_{Ri})x_{Ri}$.

The main advantages of the SDRE over QP formulation are local asymptotic stability of the algorithm and the asymptotic convergence to an optimal solution. Even though the SDRE control is not truly optimal, it still takes into consideration the future costs of the instantaneous choice of action. In contrast, the QP approach, being a point-wise optimization with respect to time, chooses the instantaneous actions without regarding to where the system is heading. As a consequence, in the simulations, when the control inputs were computed via QP formulation, the system became unstable, failing to fulfill the task (fig. 5). On the other hand, when the control inputs were computed via the SDRE

formulation, the system retained stability in every scenario tested, even when the it was under external disturbances and started relatively far from an optimal initial condition (fig. 7). The required torques were also bounded in every scenario, remaining below $12Nm$ most of the time (fig. 5d, fig. 7d and fig. 8d). However, in fig. 7d, when the system started far away from an optimal position, an initial peak of $60Nm$ was observed, decaying to $0Nm$ in less than $0.2s$. This would require a torque rate of change above $300Nm/s$, which might be difficult to realize in practice. This rate of change is due the large number of relation $\frac{\|[\lambda_0,\lambda_1,\lambda_2]\|}{\beta}$, which greatly penalizes the system when it is outside the optimal position. This issue could be alleviated by adopting a time-varying value of β, starting close to $\|[\lambda_0, \lambda_1, \lambda_2]\|$ and then decaying to the desired value. By comparing fig. 5b and fig. 7c, it can be noted that the configuration of the robot is also a function of the initial position. In practice, if there is some preferred configuration, then it is recommended to start close to that configuration. In simulation III (fig. 8), all external measured forces and disturbances were removed, in order to demonstrate that the SDRE formulation also achieves repeatable, bounded joint motion if the task is also cyclic.

There are, however, some drawbacks intrinsic to the SDRE formulation. They are:

1. $r(x_{Ri})$ must be always positive semi-definite. It means that optimizing some quantity related to the control inputs is compulsory. For instance, if the optimization criteria were solely the minimization of the L^2-norm of the joint velocities (together with error minimization), then the SDRE approach could not be used.
2. Because $r(x_{Ri})$ must be positive semi-definite, the optimization weights were chosen such as $\|[\lambda_0, \lambda_1, \lambda_2]\| \gg \beta$, in order to better satisfy equation (21). However, this relation tends to leave the problem ill-posed. Thus, the maximal tracking accuracy of the desired behavior is limited to the smallest relationship $\frac{\beta}{\|[\lambda_0,\lambda_1,\lambda_2]\|}$ achievable with numerical computation of the solution of (117).
3. The pair $A(x_{Ri})$ and $B(x_{Ri})$ must be point-wise stabilizable. In the application provided, the non-linear virtual system and error state equations did not originally satisfy this condition. In order to address this issue, an extended control vector (98) that made them stabilizable everywhere was created. It was necessary assure, however, that their effect on the overall system was virtually negligible.

If any of the aforementioned issues cannot be addressed satisfactorily, the SDRE approach cannot be used. In the overwhelming majority of cases, however, it should be

possible to circumvent them by adopting strategies similar to the ones presented in this work.

8 Conclusions and Future Works

The first part of this dissertation deals with the formulation of an impedance control problem as an optimal control problem. In this case, the impedance control of a redundant manipulator is transformed into an optimal control problem, where the main objective is to minimize a distance between the actual states of the robot end-effector and its desired states, the latter characterized by a virtual system. In essence, the controller forces the physical system to track some desired time-varying set of states, which are generated by simulating a virtual system.

The second part of this dissertation presents a procedure to convert the resulting optimal control problem into a structure compatible with the SDRE control approach. It consists of: 1- creating additional states and control inputs that facilitates the factorization and solution of the SDRE; and 2- presenting the SDC matrices that translate the optimal control problem into an impedance controller. The proposed formulation can also address redundancy resolution in infinite-horizon state-tracking problems, simply by prescribing a predefined trajectory to the virtual system.

The comparison between the QP formulation and SDRE formulation was illustrated with the numerical example of a 4-DOF redundant serial manipulator. The task was defined to be a nonlinear impedance that was also a function of the external forces applied to the system. The optimization criterion, sum of the squared joint torque, was chosen because of its importance, both from theoretical and practical points of view. From a theoretical perspective, the the sum of squared joint torques represents a measure of the energy consumption, whose minimization is a principle often found in biological motion. It is also associated with the size, power and cost of actuators required to control the system. This choice of criterium is also motivated to illustrate the shortcomings of the QP approach, which presented joint and task space instability. This result is compatible with the observations of Suh and Hollerbach (1987) and the interpretations of the theoretical results from O'Neil (2002).

Unfortunately, the performance of an optimal controller is strongly tied to the quality of parameter estimation. The application of SDRE in a system where the parameters governing the dynamical system are unaccounted, unknown or change over time (e.g. dry friction), will not assure the desired characteristics of an optimal controller. Under

these circumstances, it will be necessary to combine different types of controller together. Future work should focus on how to improve and combine different methods to assure the robustness of the manipulators. Depending on the system, one attractive solution would be the adoption of a cascated control approach. For example, in a conventional manipulator actuated with direct drive motors, the ideal states could be generated by the optimal or sub-optimal controller, and then be tracked by a conventional PID controller with adjustable gains. In more complex situations, where redundant actuation is present, such as with variable stiffness actuators, a combination of an optimal controller with a robust nonlinear controller could be more appropriate. One attempt to do so is presented in (PUKDEBOON; KUMAM, 2015), where the authors combined sliding-mode control with an optimal controller for spacecraft position control and attitude tracking, and could be readily adapted to the case presented on this dissertation.

In summary, a new formulation of impedance control of redundant manipulators was presented as a particular case of an optimal control problem. This formulation takes advantage of the benefits of an optimal controller to plan and design efficient impedance controllers. It is a systematic approach that yields a control law encompassing all the following characteristics:

- Repeatability of motion in joint space for cyclic tasks;
- Stability;
- Efficiency.

Essentially, it trivializes the problem of redundancy resolution of manipulators, when compared to quadratic optimization. The high computational costs of finding a solution to an optimal control problem are circumvented through the SDRE approach. Although departing from an optimal solution, it allows reaching sub-optimal solutions that converge asymptotically to it. This approach maintains the desirable properties of optimal controllers, which was verified through numerical simulations of a redundant 4-DOF serial link manipulator. The simulations presented promising results and demonstrated the advantages of the SDRE approach over QP approach, in terms of stability, required control effort and the quality of task execution. Future work should focus on how to assure robustness of the method in circumstances where parameters changes and unmodelled dynamics cannot be disregarded.

Bibliography

ADAMS, R. J.; HANNAFORD, B. Stable haptic interaction with virtual environments. *IEEE Transactions on robotics and Automation*, IEEE, v. 15, n. 3, p. 465–474, 1999. Cited on page 12.

AGUIRRE-OLLINGER, G. *Active impedance control of a lower-limb assistive exoskeleton*. Tese (Doutorado) — NORTHWESTERN UNIVERSITY, 2009. Cited on page 79.

AGUIRRE-OLLINGER, G.; COLGATE, J. E.; PESHKIN, M. A.; GOSWAMI, A. Active-impedance control of a lower-limb assistive exoskeleton. In: IEEE. *Rehabilitation Robotics, 2007. ICORR 2007. IEEE 10th International Conference on*. [S.l.], 2007. p. 188–195. Cited on page 12.

ANDERSON, F. C.; PANDY, M. G. Dynamic optimization of human walking. *Journal of biomechanical engineering*, American Society of Mechanical Engineers, v. 123, n. 5, p. 381–390, 2001. Cited on page 16.

ANDERSON, R. J.; SPONG, M. W. Hybrid impedance control of robotic manipulators. *IEEE Journal on Robotics and Automation*, IEEE, v. 4, n. 5, p. 549–556, 1988. Cited on page 13.

AREVALO, J. C.; GARCIA, E. Impedance control for legged robots: An insight into the concepts involved. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, IEEE, v. 42, n. 6, p. 1400–1411, 2012. Cited on page 13.

BEELER, S.; TRAN, H.; BANKS, H. Feedback control methodologies for nonlinear systems. *Journal of optimization theory and applications*, Springer, v. 107, n. 1, p. 1–33, 2000. Cited 2 times on pages 21 and 35.

BELLMAN, R. Dynamic programming and the numerical solution of variational problems. *Operations Research*, JSTOR, p. 277–288, 1957. Cited 2 times on pages 15 and 28.

BERRET, B.; DARLOT, C.; JEAN, F.; POZZO, T.; PAPAXANTHIS, C.; GAUTHIER, J. P. The inactivation principle: mathematical solutions minimizing the absolute work and biological implications for the planning of arm movements. *PLoS computational biology*, Public Library of Science, v. 4, n. 10, p. e1000194, 2008. Cited on page 66.

BERRET, B.; JEAN, F. Why don't we move slower? the value of time in the neural control of action. *Journal of neuroscience*, Soc Neuroscience, v. 36, n. 4, p. 1056–1070, 2016. Cited on page 66.

BOWLING, A.; HARMEYER, S. Repeatable redundant manipulator control using nullspace quasivelocities. *Journal of dynamic systems, measurement, and control*, American Society of Mechanical Engineers, v. 132, n. 3, p. 031007, 2010. Cited on page 54.

BROCK, W. A.; SCHEINKMAN, J. Global asymptotic stability of optimal control systems with applications to the theory of economic growth. *Journal of Economic Theory*, Elsevier, v. 12, n. 1, p. 164–190, 1976. Cited on page 19.

BURDET, E.; FRANKLIN, D. W.; MILNER, T. E. *Human robotics: neuromechanics and motor control*. [S.l.]: MIT press, 2013. Cited 2 times on pages 42 and 66.

BURTON, A.; MILLER, G. The application of integral equation methods to the numerical solution of some exterior boundary-value problems. In: THE ROYAL SOCIETY. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. [S.l.], 1971. v. 323, n. 1553, p. 201–210. Cited on page 20.

CARELLI, R.; KELLY, R. An adaptive impedance/force controller for robot manipulators. *IEEE Transactions on Automatic Control*, IEEE, v. 36, n. 8, p. 967–971, 1991. Cited on page 13.

CHANG, K.-S.; KHATIB, O. Manipulator control at kinematic singularities: A dynamically consistent strategy. In: IEEE. *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*. [S.l.], 1995. v. 3, p. 84–88. Cited on page 14.

CIMEN, T. State-dependent riccati equation (sdre) control: A survey. *IFAC Proceedings Volumes*, Elsevier, v. 41, n. 2, p. 3761–3775, 2008. Cited 2 times on pages 16 and 21.

CIMEN, T. Survey of state-dependent riccati equation in nonlinear optimal feedback control synthesis. *Journal of Guidance, Control, and Dynamics*, v. 35, n. 4, p. 1025–1047, 2012. Cited 2 times on pages 17 and 36.

CLOUTIER, J. R. State-dependent riccati equation techniques: An overview. In: IEEE. *American Control Conference, 1997. Proceedings of the 1997*. [S.l.], 1997. v. 2, p. 932–936. Cited on page 36.

COLBAUGH, R.; SERAJI, H.; GLASS, K. Adaptive impedance control of redundant manipulators. In: IEEE. *Decision and Control, 1990., Proceedings of the 29th IEEE Conference on*. [S.l.], 1990. p. 2661–2666. Cited on page 84.

ENRIGHT, P. J.; CONWAY, B. A. Optimal finite-thrust spacecraft trajectories using collocation and nonlinear programming. *Journal of Guidance, Control, and Dynamics*, v. 14, n. 5, p. 981–985, 1991. Cited on page 16.

ERDEM, E. B.; ALLEYNE, A. G. Experimental real-time sdre control of an underactuated robot. In: IEEE. *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*. [S.l.], 2001. v. 3, p. 2986–2991. Cited on page 17.

FAHROO, F.; ROSS, I. Trajectory optimization by indirect spectral collocation methods. In: *Astrodynamics Specialist Conference*. [S.l.: s.n.], 2000. p. 4028. Cited on page 16.

FERRAGUTI, F.; PREDA, N.; MANURUNG, A.; BONFE, M.; LAMBERCY, O.; GASSERT, R.; MURADORE, R.; FIORINI, P.; SECCHI, C. An energy tank-based interactive control architecture for autonomous and teleoperated robotic surgery. *IEEE Transactions on Robotics*, IEEE, v. 31, n. 5, p. 1073–1088, 2015. Cited on page 12.

FLASH, T.; HOGAN, N. The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of neuroscience*, Soc Neuroscience, v. 5, n. 7, p. 1688–1703, 1985. Cited on page 66.

FURTADO, G. P.; FORNER-CORDERO, A. Impedance control as an optimal control problem. 2018. Cited on page 12.

GARCÍA-HERAS, J.; SOLER, M.; SÁEZ, F. J. A comparison of optimal control methods for minimum fuel cruise at constant altitude and course with fixed arrival time. *Procedia Engineering*, Elsevier, v. 80, p. 231–244, 2014. Cited on page 20.

HÄRKEGÅRD, O.; GLAD, S. T. Resolving actuator redundancy—optimal control vs. control allocation. *Automatica*, Elsevier, v. 41, n. 1, p. 137–144, 2005. Cited on page 34.

HE, W.; DONG, Y.; SUN, C. Adaptive neural impedance control of a robotic manipulator with input saturation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, IEEE, v. 46, n. 3, p. 334–344, 2016. Cited on page 13.

HOGAN, N. Impedance control: An approach to manipulation. In: IEEE. *American Control Conference, 1984*. [S.l.], 1984. p. 304–313. Cited 2 times on pages 12 and 13.

HOLLERBACH, J.; SUH, K. Redundancy resolution of manipulators through torque optimization. *IEEE Journal on Robotics and Automation*, IEEE, v. 3, n. 4, p. 308–316, 1987. Cited 2 times on pages 14 and 27.

HU, Y.-R.; GOLDENBERG, A. A. Dynamic control of coordinated redundant robots with torque optimization. *Automatica*, Elsevier, v. 29, n. 6, p. 1411–1424, 1993. Cited on page 15.

HUNTINGTON, G.; BENSON, D.; RAO, A. A comparison of accuracy and computational efficiency of three pseudospectral methods. In: *AIAA guidance, navigation and control conference and exhibit*. [S.l.: s.n.], 2007. p. 6405. Cited on page 20.

ITO, K.; SCHROETER, J. D. Reduced order feedback synthesis for viscous incompressible flows. *Mathematical and computer modelling*, Elsevier, v. 33, n. 1-3, p. 173–192, 2001. Cited on page 21.

JOSHI, H. R. Optimal control of an hiv immunology model. *Optimal control applications and methods*, Wiley Online Library, v. 23, n. 4, p. 199–213, 2002. Cited on page 19.

KALMAN, R. The theory of optimal control and the calculus of variations. *Mathematical optimization techniques*, University of California Press Berkeley and Los Angels, p. 309–331, 1963. Cited on page 15.

KHATIB, O. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, IEEE, v. 3, n. 1, p. 43–53, 1987. Cited on page 14.

KLEIN, C. A.; AHMED, S. Repeatable pseudoinverse control for planar kinematically redundant manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, IEEE, v. 25, n. 12, p. 1657–1662, 1995. Cited on page 54.

KORAYEM, M.; ZEHFROOSH, A.; TOURAJIZADEH, H.; MANTEGHI, S. Optimal motion planning of non-linear dynamic systems in the presence of obstacles and moving boundaries using sdre: application on cable-suspended robot. *Nonlinear Dynamics*, Springer, v. 76, n. 2, p. 1423–1441, 2014. Cited on page 17.

KORAYEM, M. H.; NEKOO, S. R. Finite-time state-dependent riccati equation for time-varying nonaffine systems: Rigid and flexible joint manipulator control. *ISA transactions*, Elsevier, v. 54, p. 125–144, 2015. Cited on page 17.

KORAYEM, M. H.; NEKOO, S. R. Suboptimal tracking control of nonlinear systems via state-dependent differential riccati equation for robotic manipulators. In: IEEE. *Robotics and Mechatronics (ICROM), 2015 3rd RSI International Conference on*. [S.l.], 2015. p. 025–030. Cited on page 17.

LIU, D.; TODOROV, E. Evidence for the flexible sensorimotor strategies predicted by optimal feedback control. *Journal of Neuroscience*, Soc Neuroscience, v. 27, n. 35, p. 9354–9368, 2007. Cited on page 66.

LU, Z.; GOLDENBERG, A. A. Robust impedance control and force regulation: Theory and experiments. *The International journal of robotics research*, Sage Publications Sage CA: Thousand Oaks, CA, v. 14, n. 3, p. 225–254, 1995. Cited on page 13.

MARKMAN, J.; KATZ, I. Convergence of an iterative algorithm for solving hamilton-jacobi type equations. *Mathematics of computation*, v. 71, n. 237, p. 77–103, 2002. Cited on page 21.

MARKMAN, J.; KATZ, I. N. An iterative algorithm for solving hamilton–jacobi type equations. *SIAM Journal on Scientific Computing*, SIAM, v. 22, n. 1, p. 312–329, 2000. Cited on page 21.

MASON, M. T. Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, v. 11, n. 6, p. 418–432, June 1981. ISSN 0018-9472. Cited on page 12.

MORASSO, P. Spatial control of arm movements. *Experimental brain research*, Springer, v. 42, n. 2, p. 223–227, 1981. Cited on page 66.

MUSSA-IVALDI, F. A.; HOGAN, N. Integrable solutions of kinematic redundancy via impedance control. *The International Journal of Robotics Research*, Sage Publications Sage CA: Thousand Oaks, CA, v. 10, n. 5, p. 481–491, 1991. Cited 3 times on pages 14, 15, and 54.

NAKAMURA, Y.; HANAFUSA, H.; YOSHIKAWA, T. Task-priority based redundancy control of robot manipulators. *The International Journal of Robotics Research*, Sage Publications Sage UK: London, England, v. 6, n. 2, p. 3–15, 1987. Cited on page 14.

NEMEC, B.; ZLAJPAH, L. Null space velocity control with dynamically consistent pseudo-inverse. *Robotica*, Cambridge University Press, v. 18, n. 5, p. 513–518, 2000. Cited on page 14.

NENCHEV, D. N. Redundancy resolution through local optimization: A review. *Journal of Field Robotics*, Wiley Online Library, v. 6, n. 6, p. 769–798, 1989. Cited on page 14.

OKAMURA, A. M. Methods for haptic feedback in teleoperated robot-assisted surgery. *Industrial Robot: An International Journal*, Emerald Group Publishing Limited, v. 31, n. 6, p. 499–508, 2004. Cited on page 12.

O'NEIL, K. A. Divergence of linear acceleration-based redundancy resolution schemes. *IEEE Transactions on Robotics and Automation*, IEEE, v. 18, n. 4, p. 625–631, 2002. Cited 2 times on pages 14 and 57.

ORIOLO, G. Stabilization of self-motions in redundant robots. In: IEEE. *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*. [S.l.], 1994. p. 704–709. Cited on page 15.

ORIOLO, G.; CEFALO, M.; VENDITTELLI, M. Repeatable motion planning for redundant robots over cyclic tasks. *IEEE Transactions on Robotics*, IEEE, v. 33, n. 5, p. 1170–1183, 2017. Cited on page 54.

OTT, C.; DIETRICH, A.; ALBU-SCHÄFFER, A. Prioritized multi-task compliance control of redundant manipulators. *Automatica*, Elsevier, v. 53, p. 416–423, 2015. Cited on page 15.

PACHLER, C.; YABUKI, D. K. Implementação de um controle de impedâncias modular para exoesqueleto robótico de membro superior. 2014. Cited on page 79.

PARK, J. H. Impedance control for biped robot locomotion. *IEEE Transactions on Robotics and Automation*, IEEE, v. 17, n. 6, p. 870–882, 2001. Cited on page 13.

PEARSON, J. Approximation methods in optimal control i. sub-optimal control. *International Journal of Electronics*, Taylor & Francis, v. 13, n. 5, p. 453–469, 1962. Cited on page 36.

PESCH, H. J. Real-time computation of feedback controls for constrained optimal control problems. part 1: Neighbouring extremals. *Optimal Control Applications and Methods*, Wiley Online Library, v. 10, n. 2, p. 129–145, 1989. Cited on page 16.

PESCH, H. J. Real-time computation of feedback controls for constrained optimal control problems. part 1: Neighbouring extremals. *Optimal Control Applications and Methods*, Wiley Online Library, v. 10, n. 2, p. 129–145, 1989. Cited 2 times on pages 19 and 68.

PESCH, H. J. Real-time computation of feedback controls for constrained optimal control problems. part 2: A correction method based on multiple shooting. *Optimal Control Applications and Methods*, Wiley Online Library, v. 10, n. 2, p. 147–171, 1989. Cited on page 16.

PETERS, J.; MISTRY, M.; UDWADIA, F.; NAKANISHI, J.; SCHAAL, S. A unifying framework for robot control with redundant dofs. *Autonomous Robots*, Springer, v. 24, n. 1, p. 1–12, 2008. Cited 2 times on pages 14 and 15.

PFIFFNER, R.; GUZZELLA, L.; ONDER, C. Fuel-optimal control of cvt powertrains. *Control engineering practice*, Elsevier, v. 11, n. 3, p. 329–336, 2003. Cited on page 19.

PONTRYAGIN, L. S. *Mathematical theory of optimal processes*. [S.l.]: CRC Press, 1987. Cited 4 times on pages 15, 19, 28, and 68.

PORSA, S.; LIN, Y.-C.; PANDY, M. G. Direct methods for predicting movement biomechanics based upon optimal control theory with implementation in opensim. *Annals of biomedical engineering*, Springer, v. 44, n. 8, p. 2542–2557, 2016. Cited on page 20.

PUKDEBOON, C.; KUMAM, P. Robust optimal sliding mode control for spacecraft position and attitude maneuvers. *Aerospace Science and Technology*, Elsevier, v. 43, p. 329–342, 2015. Cited on page 58.

ROBERTS, R. G.; MACIEJEWSKI, A. A. Repeatable generalized inverse control strategies for kinematically redundant manipulators. *IEEE Transactions on Automatic Control*, IEEE, v. 38, n. 5, p. 689–699, 1993. Cited on page 54.

ROBERTS, R. G.; MACIEJEWSKI, A. A. Singularities, stable surfaces, and the repeatable behavior of kinematically redundant manipulators. *The International journal of robotics research*, Sage Publications Sage CA: Thousand Oaks, CA, v. 13, n. 1, p. 70–81, 1994. Cited on page 54.

ROSS, S. M.; COBB, R. G.; BAKER, W. P.; HARMON, F. G. Implementation lessons and pitfalls for real-time optimal control with stochastic systems. *Optimal Control Applications and Methods*, Wiley Online Library, v. 36, n. 2, p. 198–217, 2015. Cited 2 times on pages 21 and 34.

SADEGHIAN, H.; VILLANI, L.; KESHMIRI, M.; SICILIANO, B. Task-space control of robot manipulators with null-space compliance. *IEEE Transactions on Robotics*, IEEE, v. 30, n. 2, p. 493–506, 2014. Cited on page 15.

SALLE, J. L.; LEFSCHETZ, S. *Stability by Liapunov's Direct Method with Applications by Joseph L Salle and Solomon Lefschetz*. [S.l.]: Elsevier, 2012. v. 4. Cited on page 16.

SCHLEGL, T.; BUSS, M.; OMATA, T.; SCHMIDT, G. Fast dextrous re-grasping with optimal contact forces and contact sensor-based impedance control. In: IEEE. *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*. [S.l.], 2001. v. 1, p. 103–108. Cited on page 13.

SEMINI, C.; BARASUOL, V.; BOAVENTURA, T.; FRIGERIO, M.; FOCCHI, M.; CALDWELL, D. G.; BUCHLI, J. Towards versatile legged robots through active impedance control. *The International Journal of Robotics Research*, SAGE Publications Sage UK: London, England, v. 34, n. 7, p. 1003–1020, 2015. Cited on page 13.

SINGH, S. K.; POPA, D. O. An analysis of some fundamental problems in adaptive control of force and impedance behavior: Theory and experiments. *IEEE Transactions on Robotics and Automation*, IEEE, v. 11, n. 6, p. 912–921, 1995. Cited on page 13.

SLOTINE, J.-J. E.; LI, W. *et al. Applied nonlinear control*. [S.l.]: prentice-Hall Englewood Cliffs, NJ, 1991. v. 199. Cited 2 times on pages 76 and 77.

SOUZA, R. S.; MARTINS, T. d. C.; FURTADO, G. P.; FORNER-CORDERO, A. Model-reference adaptive impedance controller design for modular exoskeleton. In: ELSEVIER SCIENCE. *IFAC Symposium on Biological and Medical Systems*. [S.l.], 2018. Cited on page 13.

SUH, K.; HOLLERBACH, J. Local versus global torque optimization of redundant manipulators. In: IEEE. *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*. [S.l.], 1987. v. 4, p. 619–624. Cited 4 times on pages 14, 16, 27, and 57.

TAFAZOLI, S.; SALCUDEAN, S. E.; HASHTRUDI-ZAAD, K.; LAWRENCE, P. D. Impedance control of a teleoperated excavator. *IEEE Transactions on Control Systems Technology*, IEEE, v. 10, n. 3, p. 355–367, 2002. Cited on page 12.

TODOROV, E.; LI, W. Optimal control methods suitable for biomechanical systems. In: IEEE. *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE*. [S.l.], 2003. v. 2, p. 1758–1761. Cited on page 66.

TODOROV, E.; LI, W. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In: IEEE. *American Control Conference, 2005. Proceedings of the 2005*. [S.l.], 2005. p. 300–306. Cited on page 21.

TSAGARAKIS, N. G.; CALDWELL, D. G. Development and control of a 'soft-actuated'exoskeleton for use in physiotherapy and training. *Autonomous Robots*, Springer, v. 15, n. 1, p. 21–33, 2003. Cited on page 12.

UNO, Y.; KAWATO, M.; SUZUKI, R. Formation and control of optimal trajectory in human multijoint arm movement. *Biological cybernetics*, Springer, v. 61, n. 2, p. 89–101, 1989. Cited on page 20.

WATANABE, K.; IWASE, M.; HATAKEYAMA, S.; MARUYAMA, T. Control strategy for a snake-like robot based on constraint force and verification by experiment. *Advanced Robotics*, Taylor & Francis, v. 23, n. 7-8, p. 907–937, 2009. Cited on page 17.

WINTER, D. A. *Biomechanics and motor control of human movement*. [S.l.]: John Wiley & Sons, 2009. Cited on page 67.

XU, Q. Adaptive discrete-time sliding mode impedance control of a piezoelectric microgripper. *IEEE Transactions on Robotics*, IEEE, v. 29, n. 3, p. 663–673, 2013. Cited on page 13.

ZHANG, L.-Q.; RYMER, W. Z. Simultaneous and nonlinear identification of mechanical and reflex properties of human elbow joint muscles. *IEEE Transactions on Biomedical Engineering*, IEEE, v. 44, n. 12, p. 1192–1209, 1997. Cited on page 67.

ZHANG, Z.; LIN, Y.; LI, S.; LI, Y.; YU, Z.; LUO, Y. Tricriteria optimization-coordination motion of dual-redundant-robot manipulators for complex path planning. *IEEE Transactions on Control Systems Technology*, IEEE, 2017. Cited 2 times on pages 14 and 15.

APPENDIX A – Generating Human-like Motion with optimal control

When designing controllers for robotic devices that should interact closely with the human movement it is crucial to have a criterion to determine which movements are desirable or not (BURDET; FRANKLIN; MILNER, 2013). The first aspect to consider would be related to safety and this is usually solved, from a controller perspective, with the use of controller techniques that avoid the application too large torques for certain angles or angular velocities. The second aspect to be regarded is the actual execution of the movement. Despite of the redundancy of possible solutions for arm reaching (the arm has 7 DOF), the trajectory of the hand is straight and smooth with a typical bell-shaped velocity profile (MORASSO, 1981; FLASH; HOGAN, 1985). Therefore, despite of the availability of infinite solutions to perform a certain task, humans tend to perform the same strategy, following some kind of "invariance principles". One of the most successful control theories that have been used to generate human-like motion patterns is based on optimal control theory (TODOROV; LI, 2003; LIU; TODOROV, 2007). More recent work considered a cost function based not only in energy consumption but also on accuracy, endpoint stability and movement duration. However, the definition of the optimal time is not completely solved and it has been subject of research in the recent years (BERRET *et al.*, 2008; BERRET; JEAN, 2016). In this respect, it would be very interesting to determine the optimal time to perform the task. Here, the movement of an arm is simulated, using optimal control to find the optimal time. The idea is to find an underlying objective function which minimization yields a solution that resembles the motion that a human arm executes. In order to do so, a simple 1-dof model of the human arm is derived. Based on experimental data, an objective function is generated. The behavior displayed by the human arm motion is used as a basis to define the shape of fig. 1. The motion in question is the upper limb flexion against gravity, while different weights are held. It allows the evaluation motion changes according to an external force.

A.1 Upper Limb flexion extension against gravity

The upper limb movement is considered with one degree of freedom (1-DoF), modeled as an articulated segment mechanism moving in a gravity field and controlled

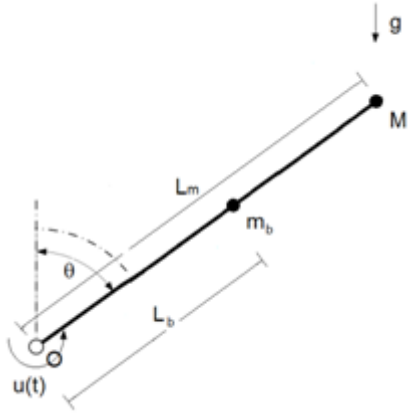


Figure 9 – *1-DOF Upper Limb free-body diagram*

Source: *Guilherme Phillips Furtado et al, 2018*

by a torque produced by the muscles. The free-body diagram of the model considered is shown in fig. 9.

In this brief study, the aim of the subject is to lift a mass by flexing the elbow, maintaining the arm fixed in the vertical position while rotating the forearm in sagittal plane. The equations of the system are:

$$\dot{x}(t) = f(x(t)) + B\ u(t), \quad f : \mathbb{R}^3 \to \mathbb{R}^2, B \in \mathbb{R}^2, \tag{134}$$

with:

$$x(t) =: \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \end{bmatrix} : [0, T] \to \mathbb{R}^2, \tag{135}$$

$$f(x(t)) = \begin{bmatrix} x_2 \\ \frac{2(-gL_m M sin(x_1) - gL_b m_b sin(x_1) - c_b x_2(t))}{(2I_b + L_m^2 M + 2L_b m_b^2)} \end{bmatrix}, \tag{136}$$

$$B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1/(2I_b + L_m^2 M + 2L_b m_b^2) \end{bmatrix}. \tag{137}$$

Table 2 describes what each parameter stands for. The friction coefficient for the elbow is based on (ZHANG; RYMER, 1997), while the moment of inertia, mass and center of mass of the forearm are based on (WINTER, 2009). No frictions other than viscous friction are considered in the model.

Table 2 – *List of Parameters of the 1-DoF Human Arm*

Parameter	Description
g	Gravity acceleration
L_m	Distance between elbow and mass to be lifted
M	Mass to be lifted
L_b	Center of mass of the forearm
m_b	Mass of the forearm
c_b	Elbow flexion friction coefficient
I_b	Moment of inertia of the forearm
$\theta(t)$	Forearm angular displacement
$u(t)$	Torque applied at the elbow

A.2 Objective Function

The method consists of finding an objective function assuming that the behavior of the system is optimal. The optimal solution is the motion of a human subject flexing the elbow along the sagittal plane, from an initial angle to a final angle. The optimal control problem is formulated using the Euler-Lagrange equation of the system supplied by Pontryagin's minimum principle (PONTRYAGIN, 1987; PESCH, 1989b).

For the optimal control problem, let us consider the following functional:

$$\mathcal{L}(x(t), u(t)) = \alpha + \beta u(t)^2 + \gamma\ u(t)^2 x_2(t)^2, \quad \alpha, \beta, \gamma \in \mathbb{R}\,. \tag{138}$$

The main idea is to find parameters α, β and γ. The parameter α can be interpreted as the weight for time interval T, β weights the squared control effort term, associated with energy expenditure, and γ is the weight for a quadratic pseudo-power term.

The objective function to be minimized is

$$J[u, x] = \int_0^T \mathcal{L}(x(t), u(t))dt. \tag{139}$$

The terminal state constraint is given by:

$$\psi(x(T), T) = x(T) - x_f = 0. \tag{140}$$

The Hamiltonian H and an auxiliary function Φ are defined as:

$$H(x(t), u(t), \lambda(t)) = \mathcal{L}(x(t), u(t)) + \lambda(t)^T f(x(t), u(t)), \tag{141}$$

$$\Phi(x(t), t, v) = v\psi(x(t), t), \tag{142}$$

For the problem stated by equations (138)-(142), the following conditions can be obtained, considering the final time is unspecified:

$$x(t) = \partial H/\partial \lambda^T, \tag{143}$$

$$\lambda(t) = -\partial H/\partial x^T, \tag{144}$$

$$-\partial H/\partial u = 0, \tag{145}$$

$$\lambda(T) = -\partial \Phi/\partial x|_T, \tag{146}$$

$$x(T) = x_f, \tag{147}$$

$$H(x(t), u(t), \lambda(t)) = 0, \tag{148}$$

where $\lambda : [0, T] \mapsto U \subset \mathbb{R}^2$ and $v \in \mathbb{R}^2$ denote Lagrange multipliers. Since the final time is unknown, the equations (143)-(148) will be non-dimensionalized in respect to time. Defining

$$t = \tau T, \quad \tau \in \mathbb{R}, 0 \leq \tau \leq 1, \tag{149}$$

and replacing it on the system equations,

$$\dot{x}(\tau) = Tf(x(\tau)) + g(x(\tau))u(\tau), \tag{150}$$

$$\lambda(\tau) = Tf(-\lambda(\tau)^T)(\partial f/\partial x - \partial \mathcal{L}/\partial x), \tag{151}$$

$$u(\tau) = (-\lambda^T B_2)/(2\beta + 2\gamma x_2), \tag{152}$$

$$\lambda(1) = v, \tag{153}$$

$$x(1) = x_f, \tag{154}$$

$$x(0) = x_0, \tag{155}$$

$$H(x(\tau), u(\tau), \lambda(\tau)) = 0. \tag{156}$$

The solution to the two-point boundary value problem (TPBVP) enumerated by equations (150)-(156) yields an optimal trajectory, along with the terminal time T, which will be compared with the movement of a human subject.

A.3 Description of the experiment

In order to compare the results of the optimal controller with experimental data, a series of tests were carried out. One male healthy volunteer participated in the tests, that were approved by the Local Ethical Committee. The volunteer height was $1.74m$ a body mass of $91kg$. The distance from elbow joint center to the fingertips $L_b = 45cm$, and the distance from the elbow joint center to the center of the hand $L_m = 35cm$. The experiment consisted of lifting different weights with a flexion-extension of the elbow. The instruction to the volunteer was to lift the weight with only a flexion of the elbow. The experiment lasted $20s$, during the first 10 s the task was performed with clear pauses with the elbow was flexed and extended and for the following $10s$ the same task was repeated without pausing. The elbow flexion was performed with the subject standing comfortably with the arms hanging at both sides in the standard anatomical position. The task was performed without weight and with weights of $2kg$ and $3kg$. The movement was recorded with an inertial measurement unit (VN-100, VectorNav Technologies, TX, USA) placed at the dorsum of the hand with a sampling frequency of $200Hz$. The data analyzed corresponded to the elbow flexion (lifting he load) as it agrees with the simulation objectives.

Table 3 – *List of Parameter Values for the Optimal Time Experiment*

Parameter	Value
g	$9.78m/s^2$
L_m	$0.35cm$
M	$0,2kgor3kg$
L_b	$0.1935m$
m_b	$2.29kg$
c_b	$0.025Nms$
I_b	$0.0099kgm^2$
α	13.4
β	1
γ	0.15

A.4 Results and Discussion

The TPBVP was solved in MATLAB, using the parameters from table 2.

Table 4 – *Comparison Between Experimental and Simulation Results*

Mass (kg)	Mean experimental time (s)	Simulation time (s)
0	0.7	0.7
2	0.89	0.9
3	0.97	0.95
Mass (kg)	Mean initial angle (rad)	Mean final angle (rad)
0	0.41	2.48203
2	0.833275	2.63815
3	0.864375	2.79575

The initial and final position for the simulation were taken from the valleys and peaks of the measured angular position. The values of the parameters α, β and γ were adjusted based on the elbow flexion when no mass was being lifted. In Table 4, the average time intervals for three different masses are confronted with the simulation results. Figures 10, 11 and 12 compare the simulation results and the experimental data for selected initial and final states.

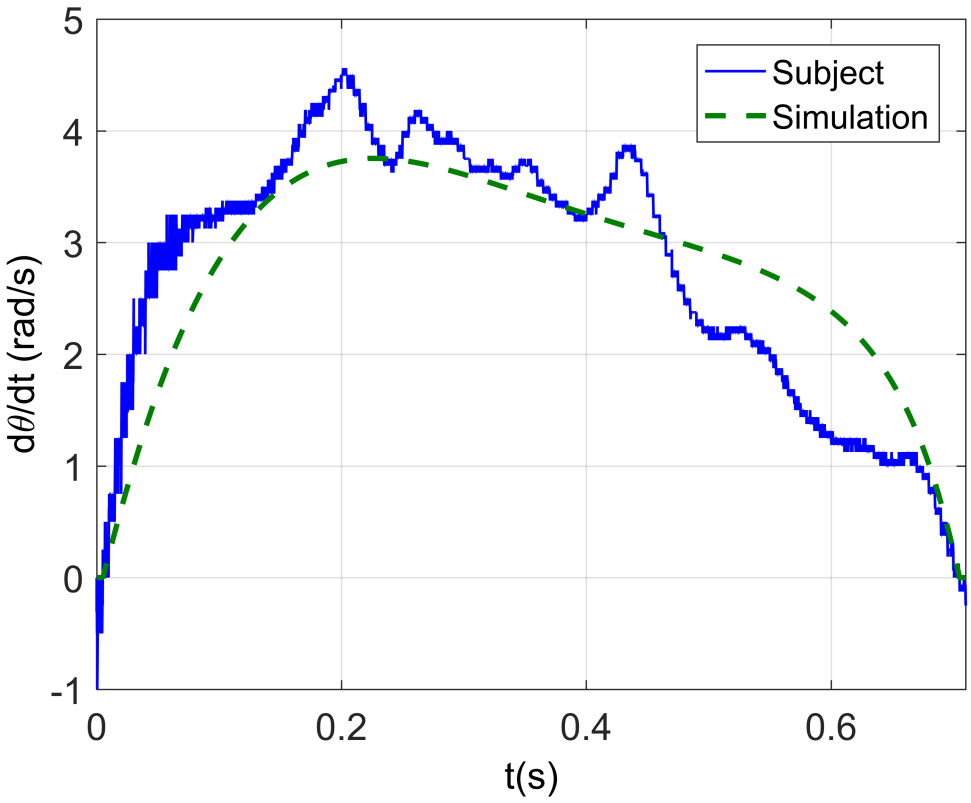


Figure 10 – *Comparison between experimental and simulated angular velocity profile when M= 0kg.*

Source: *Guilherme Phillips Furtado et al, 2018*

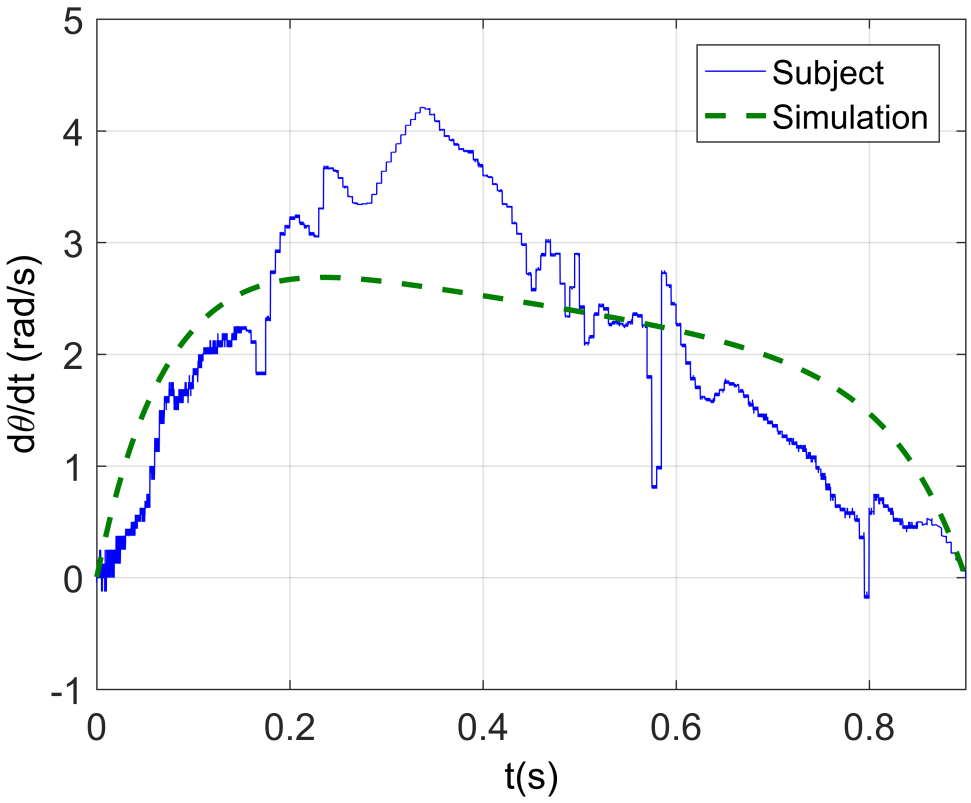


Figure 11 – *Comparison between experimental and simulated angular velocity profile when M= 2kg.*

Source: *Guilherme Phillips Furtado et al, 2018*

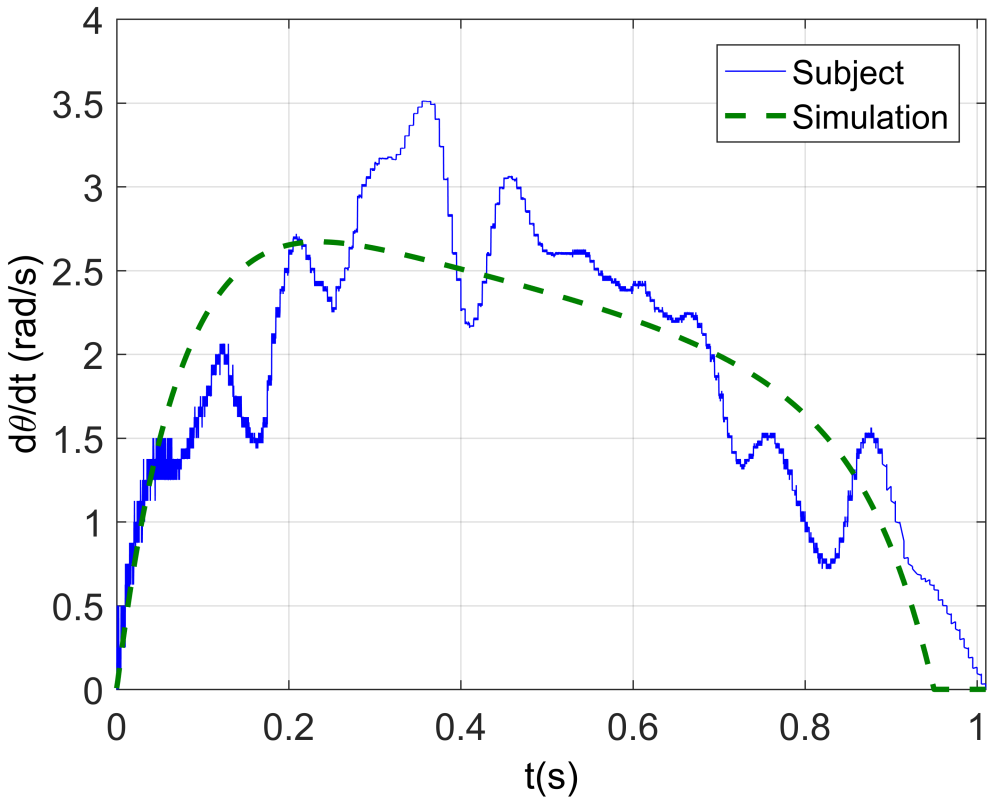


Figure 12 – *Comparison between experimental and simulated angular velocity profile when M= 3kg.*

Source: *Guilherme Phillips Furtado et al, 2018*

From the results, it can be concluded that the optimal controller adjusted for the arm motion without any load was still valid with increasing mass (figs 2 and 3). The angular velocities found in the simulations resemble the experimental ones and they have the same trend with increasing weight. In addition, the movement time emerged as a part of the optimal solution.

APPENDIX B – Implementation of Impedance Control via Adaptive Control

In this appendix, it is illustrated how impedance control can be used to control wearable assistive devices (ie. exoskeletons). One of the major challenges in exoskeleton design is to implement a desired type of human-robot cooperation. The relation between robotic and human joints can be defined by the robot controller, depending on the nature of the task and the user needs. An exoskeleton is an active robotic orthosis acting in parallel with the body segments. It could, for example, be used to assist a patient either in daily life activities, or to assist a therapist when applying a physical therapy to a patient. Impedance control is commonly implemented in exoskeletons because it allows the designer to predefine the dynamic relationship that comprises the human-robot cooperation.

However, obtaining a precise model of the human body is often not possible. The muscle dynamics have non-linear, time-varying behavior, with parameters which are difficult to measure. Given such complexity, it is often impractical to design a model-based controller that fits a large range of human motions and tasks. It can be considered that the controller should not rely on a precise knowledge of system parameters. Under these circumstances, simply finding a solution of 21 based on approximate parameter values may not be enough to obtain an effective impedance controller.

Adaptive Control has been used to handle parameter uncertainty (SLOTINE; LI *et al.*, 1991) and has the advantage of parameter estimation, which is very useful for motor control research and performance assessment in rehabilitation. The adaptation mechanism compares the expected input/output relation in order to minimize the tracking error by adjusting the parameters estimation.

Here, an implementation of Model Reference Adaptive Control (MRAC) that aims to imposes a desired behavior to both human limb and exoskeleton as a whole system will be presented as an example of an impedance controller implementation. The exoskeleton controller should guarantee that, for given a torque input, the coupled system follows the motion determined by the desired behavior. This approach has the aim to simultaneously control the system and provide parameter identification.

For simplicity, human and exoskeleton are assumed to be perfectly coupled, with no displacement between them. Therefore, the position, velocity and acceleration are the

same while inertial parameters are simply summed; e.g. the equivalent inertia becomes $J = J_H + J_e$, to finally obtain the system dynamics represented by figure 14.

The equation of the coupled system is given by

$$\tau_h + \tau_e = (J_h + J_e)\,\ddot{\theta} + (b_h + b_e)\,\dot{\theta} + (k_h + k_e)\,\theta \tag{157}$$

where J_H denotes the human limb inertia, J_E denotes the exoskeleton inertia, θ denotes the angular displacement, b_H denotes the viscous friction of the human limb, b_E denotes the viscous friction of the exoskeleton, k_H denotes the linear stiffness of the human limb, k_E denotes the linear stiffness of the exoskeleton k_E and τ_H denotes the torque acting on the system.

The coupled system behavior must be controlled in such way that when the user applies some effort, the system must perform a certain movement. However, the coupled plant parameters are not precisely known, due to two reasons: 1- a precise model to represent its dynamics is not available; 2- the plant parameters change due to muscle contraction according to the user's intention. In addition, the controller must also satisfy user safety and comfort requirements (SLOTINE; LI *et al.*, 1991).

B.1 Model-Reference Adaptive Control (MRAC)

The MRAC offers the possibility of defining a reference model to specify the desired system impedance and has the structure shown in figure 13.

The system with the simplified coupled dynamics is given by Eq.(157), representing the displacement resulting from an applied torque to the human joint. There are uncertainties with respect to the parameters J_H, b_H and k_H.

The reference model consists the desired system behavior 16, and depends on the nature of the exoskeleton task. For a rehabilitation exoskeleton, for example, the desired system could have smaller impedance parameters, resulting in larger displacement with same applied torque, or the same displacement with less torque.

The controller is responsible to maintain the system tracking the desired dynamics. For instance, in the case of a rehabilitation exoskeleton, it might be possible to apply this controller in such a way that the result will be to assist the limb movement of a disabled patient.

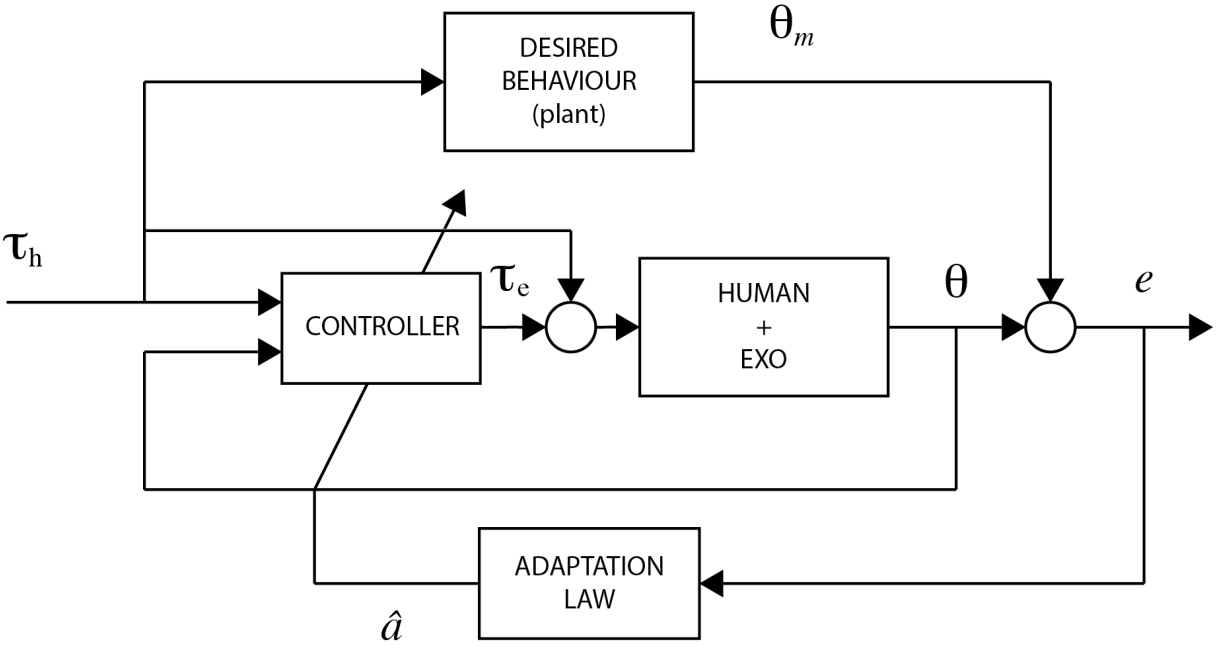


Figure 13 – *Model Reference Adaptive Control.*

Source: *Rafael Sanchez Souza et al, 2018*

The design approach considers the simplest problem: the system full state is measurable with a linear mode. The second-order linear system is represented by:

$$a_2\ddot{\theta} + a_1\dot{\theta} + a_0\theta = u, \tag{158}$$

where $\boldsymbol{a} = [\ J \quad b \quad K\ ]^T$ is the unknown coefficient vector and $\boldsymbol{\alpha} = [\ J_v \quad b_v \quad k_v\ ]^T$ the desired (virtual) coefficients for the reference-model to be tracked by the controller:

$$\alpha_2\ddot{\theta}_m + \alpha_1\dot{\theta}_m + \alpha_0\theta_m = \tau_h \tag{159}$$

with τ_h being the torque applied to the joints by the patient muscles. The control law is chosen as

$$u = \hat{a}_2\ddot{\theta} + \hat{a}_1\dot{\theta} + \hat{a}_0\theta = \mathbf{v}^T(t)\hat{\mathbf{a}}(t), \tag{160}$$

where

$$z(t) = \ddot{\theta}_m - \beta_1\dot{e} - \beta_0 e$$
$$\mathbf{v}(t) = \begin{bmatrix} z(t) & \dot{\theta}_m & \theta_m \end{bmatrix}^T$$
$$\hat{\mathbf{a}}(t) = [\ \hat{a}_2 \quad \hat{a}_1 \quad \hat{a}_0\ ]^T$$

with the tracking error given by $e = \theta - \theta_m$.

The adaptation law is given by

$$\dot{\hat{\mathbf{a}}} = -\mathbf{\Gamma v B}^t\mathbf{P e}, \tag{161}$$

and

$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

B.2 Simulations and Results

In order to verify the control design, simulations were made using MATLAB for the coupled plant represented by the diagram in figure 14. The vector of the plant parameters is defined a sum of the two coupled inertias J, the equivalent damping b and equivalent stiffness K. The values were taken from (PACHLER; YABUKI, 2014; AGUIRRE-OLLINGER, 2009):

$$a = \begin{bmatrix} 0.115 + 0.199 \;\; Kgm^2 \\ 2.52 + 1.32 \;\; Nms/rad \\ 8.6 + 5.12 \;\; Nm/rad \end{bmatrix}.$$

The tracking error parameters are:

$$\boldsymbol{\beta} = [\;\; 10 \quad 40 \;\;]. \tag{162}$$

And the adaptation law parameters Γ and P:

$$\Gamma = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 5 & 0 \\ 0.6 & 0 & 40 \end{bmatrix},$$

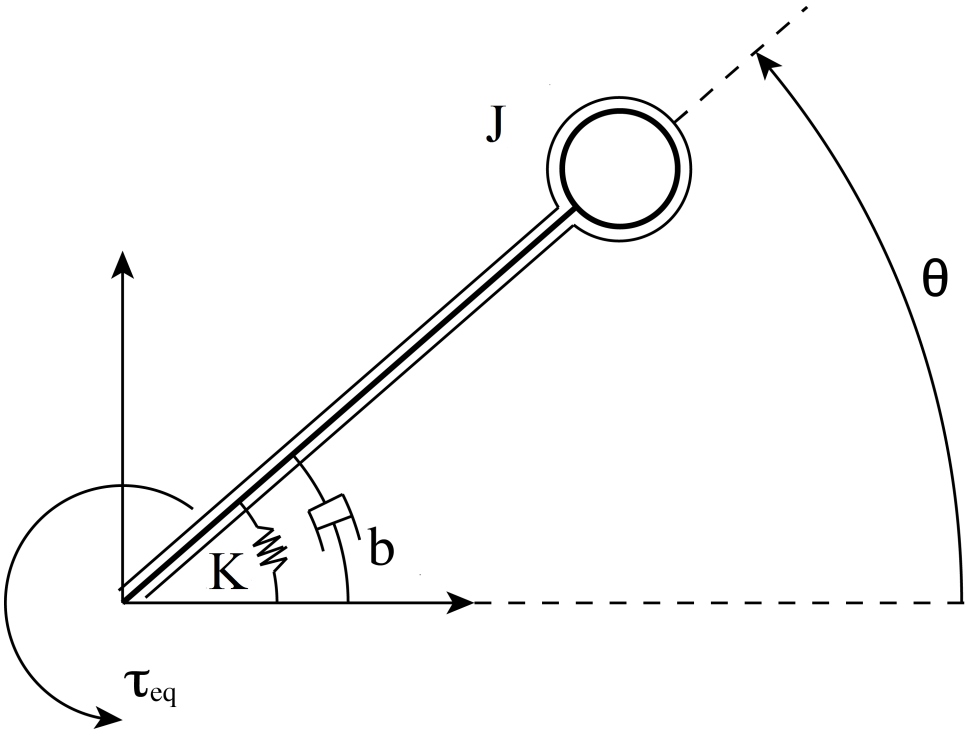


Figure 14 – *Diagram representing workspace and coupled coupled plant.*

Source: *Rafael Sanchez Souza et al, 2018*

$$P = \begin{bmatrix} 0 & 0 \\ 3 & 1 \end{bmatrix}.$$

For a preliminary verification of the controller performance, it is necessary to check whether the controller is able to compensate the presence of the exoskeleton. In order to do that, the Model-Reference vector (or desired behavior) is set as $\alpha = [\ 0.112 \quad 2.52 \quad 8.6\ ]$: that corresponds to the aforementioned modeled human arm parameters. Also, the initial parameter estimation is set to a value 20% smaller than the defined system parameters. Figure 15 shows the simulation for a step entrance tau_h, the plant angular displacement θ and the reference model response θ_m.

The final goal of the control design is to assist the arm movement of the user. In order to achieve an effective assistance, all the model reference parameters must be changed according to the desired behavior. Figure 16 shows the controller performance for $\boldsymbol{\alpha} = [\ 0.115 \quad 0.95 \quad 2.58\ ]^T$ in opposition to the previously set behavior.

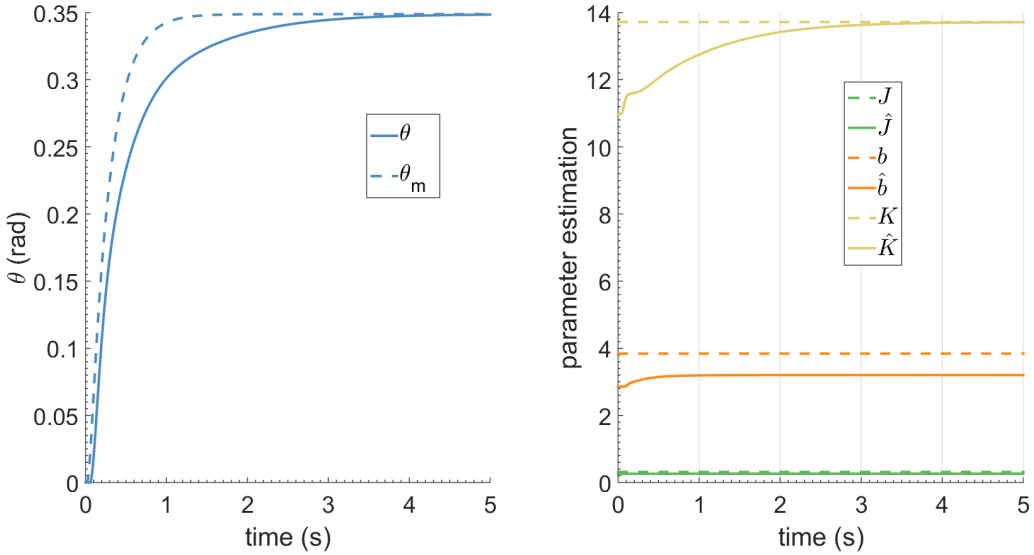


Figure 15 – *Tracking performance and parameter estimation for a step input of* 3 Nm *and parameter uncertainty of 20% and reference model set for transparent behavior.*

Source: *Rafael Sanchez Souza et al, 2018*

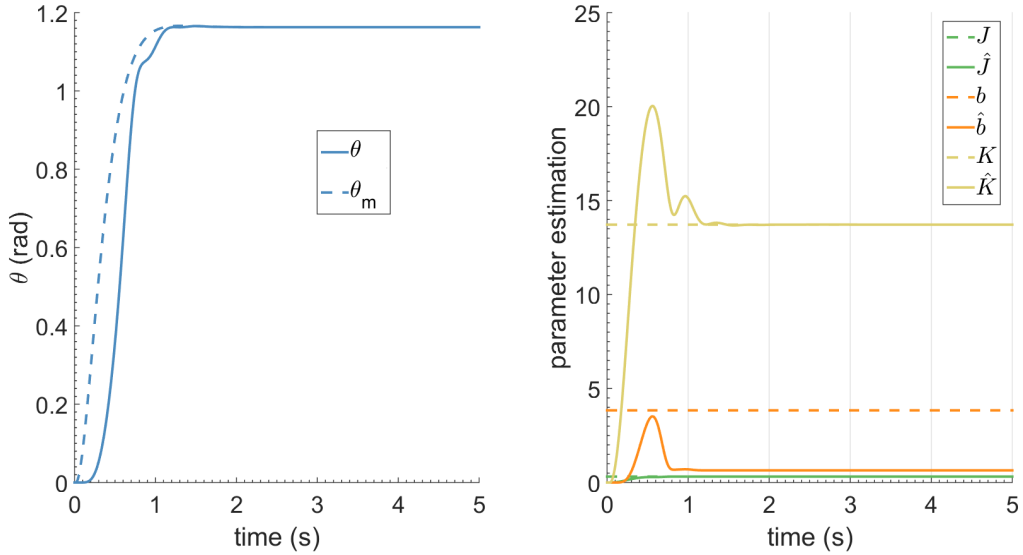


Figure 16 – *Tracking performance and parameter estimation for a step input of* 3 Nm *with zero prior knowledge of the plant and assistive control.*

Source: *Rafael Sanchez Souza et al, 2018*

Parameter Uncertainty and Estimation

As previously mentioned, the adaptive control has the advantage of dealing with unknown parameters and provide an estimation of them. Figure 16 shows the system response to a step function for an initial estimative vector $\hat{\mathbf{a}} = [\ 0 \quad 0 \quad 0\ ]$. Figure 17 shows the system response to a sinusoidal function for the same initial estimative vector. It is possible to observe the performance improving as the parameter estimation converge. A more complex reference signal - sin in contrast to a step - provides a better performance for the estimation.

The controller was also tested with a smoothed square signal. Plant response and parameters estimation can be seen in figure 18.

Controller robustness is verified in the simulations shown with respect to dealing with uncertainties. The parameters J, b and K are varied up to 30% to simulate unpredicted behaviors. The result can be seen in figure 19.

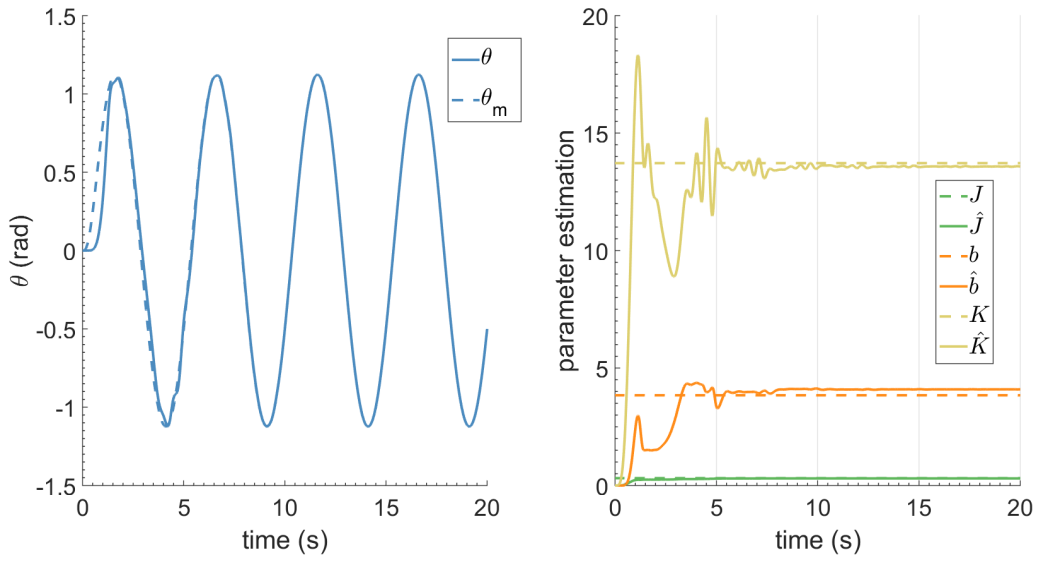


Figure 17 – *Tracking performance and parameter estimation for a sinusoidal input of amplitude 3 Nm with zero prior knowledge of the plant and assistive control.*

Source: *Rafael Sanchez Souza et al, 2018*

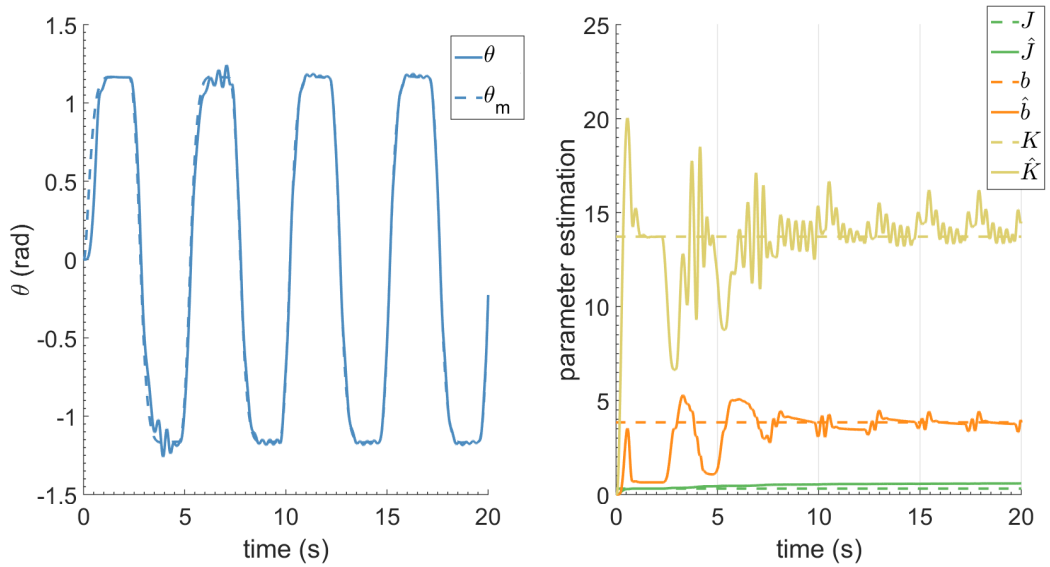


Figure 18 – *Tracking performance and parameter estimation for a smoothed square input of amplitude 3 Nm with zero prior knowledge of the plant and assistive control*

Source: *Rafael Sanchez Souza et al, 2018*

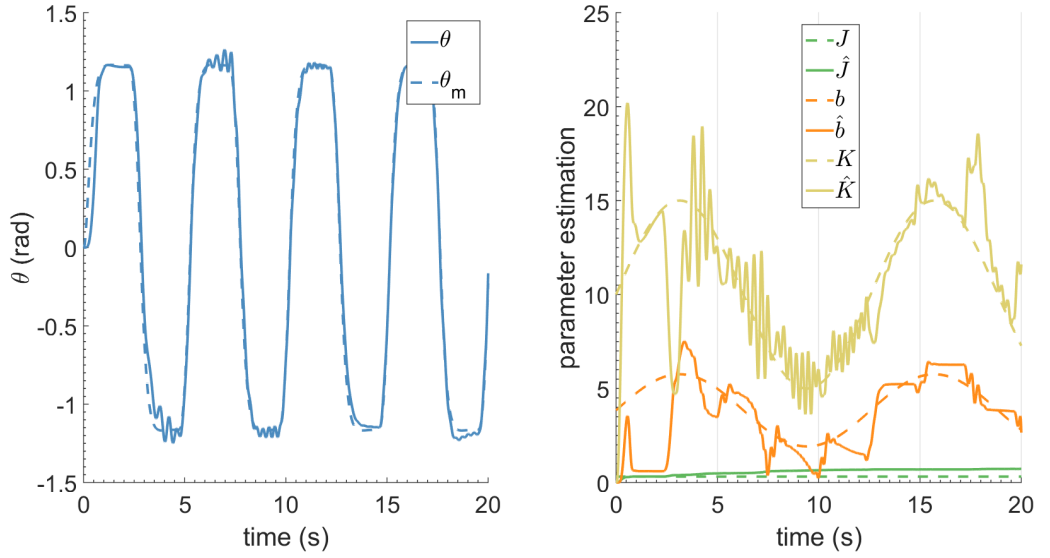


Figure 19 – *Tracking performance and parameter estimation for a smoothed square entrance of amplitude 3 Nm with zero prior knowledge of the plant, assistive control and plant parameters variation.*

Source: *Rafael Sanchez Souza et al, 2018*

B.3 Discussions

The adaptive control is a well-known technique that is suitable for the control of systems with large changes in the parameters, like the human limbs. Therefore, adaptive control of a robot in close interaction with the human (such as an exoskeleton) seems to be an ideal candidate for this application. It can be observed that it allows proper tracking of the reference signal with accurate identification of the parameters, even in the case of variation of these parameters.

The adaptive controller (MRAC) presented here was able impose the desired dynamics and simultaneously estimate the plant parameters. It was found that the response was bounded as in the robustness test for parameters variation up to 30%.

An important aspect was to show that the MRAC controller would be able to identify the parameters and adapt with time. Figure 17 shows the variation of the tracking error for a sinusoidal input and it can be seen the improvement with time. This estimation behaves properly even in the case of an smoothed square wave input (see figure 18) and time-varying parameters, as in figure 19.

However, in the case of a system with multiple degrees of freedom, the implementation of an MRAC controller, as presented here, would require the inversion of $J_p(q)$ (or $J_p(q)D(q)^{-1}$), from equation (24) (COLBAUGH; SERAJI; GLASS, 1990). If the system is redundant (and $J_p(q)$ is rectangular), a method to resolve redundancy must be provided. The usual strategy to resolve redundancy is to find a generalized inverse of $J_p(q)$ based on quadratic optimization procedure.

APPENDIX C – Symbolic Manipulation Code

The following code has been written in Wolfram Mathematica, meant to generate symbolic expressions of the ODEs that govern system motion, the A, B Q, R and N matrices to be used in Matlab.

```
(*Symbolicmanipulationcode.Exportstheexpressionsusedtocreatethefunctions RicattiA.m, RicattiB.m
Clear["Global*"]
(*Rotation*)
T01 = {{Cos[theta1[t]], −Sin[theta1[t]], 0, 0}, {Sin[theta1[t]], Cos[theta1[t]], 0, 0}, {0, 0, 1, 0}, {0, 0, 0, 1
T12 = {{Cos[theta2[t]], −Sin[theta2[t]], 0, L1}, {Sin[theta2[t]], Cos[theta2[t]], 0, 0}, {0, 0, 1, 0}, {0, 0, 0,
T23 = {{Cos[theta3[t]], −Sin[theta3[t]], 0, L2}, {Sin[theta3[t]], Cos[theta3[t]], 0, 0}, {0, 0, 1, 0}, {0, 0, 0,
T34 = {{Cos[theta4[t]], −Sin[theta4[t]], 0, L3}, {Sin[theta4[t]], Cos[theta4[t]], 0, 0}, {0, 0, 1, 0}, {0, 0, 0,
q = {theta1[t], theta2[t], theta3[t], theta4[t]};
qp = D[q, t];
(*centrer of mass*)
G10 = {l1 ∗ Cos[theta1[t]], l1 ∗ Sin[theta1[t]], 0};
G20 = {L1 ∗ Cos[theta1[t]] + l2 ∗ Cos[theta2[t]], L1 ∗ Sin[theta1[t]] + l2 ∗ Sin[theta2[t]], 0};
G30 = {L1 ∗ Cos[theta1[t]] + L2 ∗ Cos[theta2[t]] + l3 ∗ Cos[theta3[t]], L1 ∗ Sin[theta1[t]] + L2 ∗ Sin[th
G40 = {L1 ∗ Cos[theta1[t]] + L2 ∗ Cos[theta2[t]] + L3 ∗ Cos[theta3[t]] + l4 ∗ Cos[theta4[t]], L1 ∗ Sin[tl
(*end-effector*)
Ge0 = {L1 ∗ Cos[theta1[t]] + L2 ∗ Cos[theta2[t]] + L3 ∗ Cos[theta3[t]] + L4 ∗ Cos[theta4[t]], L1 ∗ Sin[t
(*Jacobian end effector*)
Jacob = D[Ge0, {{theta1[t], theta2[t], theta3[t], theta4[t]}}];
Jacobtranspose = Transpose[Jacob];
(*forces at end effector*)
Force = {Fx, Fy, 0};
Fphi = 0;
(*velocidades*)
Vg10 = Simplify[D[G10, t]];
Vg20 = Simplify[D[G20, t]];
Vg30 = Simplify[D[G30, t]];
Vg40 = Simplify[D[G40, t]];
```

```
(*Velocidades angulares*)
w1 = {0, 0, D[theta1[t], t]};
w2 = {0, 0, D[theta2[t] + theta1[t], t]};
w3 = {0, 0, D[theta3[t] + theta1[t] + theta2[t], t]};
w4 = {0, 0, D[theta4[t] + theta3[t] + theta1[t] + theta2[t], t]};
J1 = DiagonalMatrix[{Jx1, Jy1, Jz1}];
J2 = DiagonalMatrix[{Jx2, Jy2, Jz2}];
J3 = DiagonalMatrix[{Jx3, Jy3, Jz3}];
J4 = DiagonalMatrix[{Jx4, Jy4, Jz4}];
dq1 = D[theta1[t], t];
dq2 = D[theta2[t], t];
dq3 = D[theta3[t], t];
dq4 = D[theta4[t], t];
T = Simplify[(m1 * Vg10.Vg10 + w1.J1.w1 + m2 * Vg20.Vg20 + w2.J2.w2 + m3 * Vg30.Vg30 + w3.J
V = Simplify[-m1 * gravity.G10 - m2 * gravity.G20 - m3 * gravity.G30 - m4 * gravity.G40];
R = c1 * dq1^2/2 + c2 * (dq2 - dq1)^2/2 + c3 * (dq3 - dq2)^2/2 + c4 * (dq4 - dq3)^2/2;
DTDq1p = Simplify[D[D[T, D[theta1[t], t]], t]];
DTDq2p = Simplify[D[D[T, D[theta2[t], t]], t]];
DTDq3p = Simplify[D[D[T, D[theta3[t], t]], t]];
DTDq4p = Simplify[D[D[T, D[theta4[t], t]], t]];
DVDq1 = Simplify[D[V, theta1[t]]];
DVDq2 = Simplify[D[V, theta2[t]]];
DVDq3 = Simplify[D[V, theta3[t]]];
DVDq4 = Simplify[D[V, theta4[t]]];
DTDq1 = Simplify[D[T, theta1[t]]];
DTDq2 = Simplify[D[T, theta2[t]]];
DTDq3 = Simplify[D[T, theta3[t]]];
DTDq4 = Simplify[D[T, theta4[t]]];
DRDq1p = Simplify[D[R, D[theta1[t], t]]];
DRDq2p = Simplify[D[R, D[theta2[t], t]]];
DRDq3p = Simplify[D[R, D[theta3[t], t]]];
DRDq4p = Simplify[D[R, D[theta4[t], t]]];
```

```
Eq1 = Simplify[DTDq1p - DTDq1 + DVDq1 + DRDq1p] - T1 + T2 - (Jacobtranspose.Force)[[1]];
Eq2 = Simplify[DTDq2p - DTDq2 + DVDq2 + DRDq2p] - T2 + T3 - (Jacobtranspose.Force)[[2]];
Eq3 = Simplify[DTDq3p - DTDq3 + DVDq3 + DRDq3p] - T3 + T4 - (Jacobtranspose.Force)[[3]];
Eq4 = Simplify[DTDq4p - DTDq4 + DVDq4 + DRDq4p] - T4 - (Jacobtranspose.Force)[[4]];

ddq1 = D[D[theta1[t], t], t];
ddq2 = D[D[theta2[t], t], t];
ddq3 = D[D[theta3[t], t], t];
ddq4 = D[D[theta4[t], t], t];
(*theta1*)
a11 = Simplify[Coefficient[Eq1, ddq1]];
a12 = Simplify[Coefficient[Eq2, ddq1]];
a13 = Simplify[Coefficient[Eq3, ddq1]];
a14 = Simplify[Coefficient[Eq4, ddq1]];
(*theta2*)
a21 = Simplify[Coefficient[Eq1, ddq2]];
a22 = Simplify[Coefficient[Eq2, ddq2]];
a23 = Simplify[Coefficient[Eq3, ddq2]];
a24 = Simplify[Coefficient[Eq4, ddq2]];
(*theta3*)
a31 = Simplify[Coefficient[Eq1, ddq3]];
a32 = Simplify[Coefficient[Eq2, ddq3]];
a33 = Simplify[Coefficient[Eq3, ddq3]];
a34 = Simplify[Coefficient[Eq4, ddq3]];
(*y4*)
a41 = Simplify[Coefficient[Eq1, ddq4]];
a42 = Simplify[Coefficient[Eq2, ddq4]];
a43 = Simplify[Coefficient[Eq3, ddq4]];
a44 = Simplify[Coefficient[Eq4, ddq4]];
(*resto*)
b1 = -Simplify[Eq1/.{ddq1 → 0, ddq2 → 0, ddq3 → 0, ddq4 → 0}];
b2 = -Simplify[Eq2/.{ddq1 → 0, ddq2 → 0, ddq3 → 0, ddq4 → 0}];
```

```
b3 = -Simplify[Eq3/.{ddq1 -> 0, ddq2 -> 0, ddq3 -> 0, ddq4 -> 0}];
b4 = -Simplify[Eq4/.{ddq1 -> 0, ddq2 -> 0, ddq3 -> 0, ddq4 -> 0}];
MatrixA = {{a11, a12, a13, a14}, {a21, a22, a23, a24}, {a31, a32, a33, a34}, {a41, a42, a43, a44}};
MatrixB = {b1, b2, b3, b4};
fx1 = {dq1, dq2, dq3, dq4};
fx2 = Inverse[MatrixA].MatrixB;
(*SDRE*)
(*SDRE*)
BnI = {Coefficient[MatrixB[[1]], {T1, T2, T3, T4}, 1], Coefficient[MatrixB[[2]], {T1, T2, T3, T4}, 1], Co
AnI = Simplify[MatrixB - BnI.{T1, T2, T3, T4}];
(*InMatlab, atan2isy, x; notx, y; likemathematica*)
theta1eq = ArcTan[Fy, Fx];
theta2eq = ArcTan[Fy, Fx];
theta3eq = ArcTan[Fy, Fx];
theta4eq = ArcTan[Fy, Fx];
(*ATTENTION, SINCFUNCTIONINMATLABISNORMALIZED, SOWEHAVETODIVIDEHEREB
sinctheta1 = Coefficient[AnI[[1]], Sin[theta1[t]]] * Sinc[theta1[t]/Pi];
sinctheta2 = Coefficient[AnI[[2]], Sin[theta2[t]]] * Sinc[theta2[t]/Pi];
sinctheta3 = Coefficient[AnI[[3]], Sin[theta3[t]]] * Sinc[theta3[t]/Pi];
sinctheta4 = Coefficient[AnI[[4]], Sin[theta4[t]]] * Sinc[theta4[t]/Pi];
(*biasterms, tobeaddedlateron*)
beta1 = Coefficient[AnI[[1]], Cos[theta1[t]]] * Cos[theta1[t]]/beta[t];
beta2 = Coefficient[AnI[[2]], Cos[theta2[t]]] * Cos[theta2[t]]/beta[t];
beta3 = Coefficient[AnI[[3]], Cos[theta3[t]]] * Cos[theta3[t]]/beta[t];
beta4 = Coefficient[AnI[[4]], Cos[theta4[t]]] * Cos[theta4[t]]/beta[t];
betacolumn = Inverse[MatrixA].{{beta1}, {beta2}, {beta3}, {beta4}};
AnI55 = Coefficient[AnI[[1]], dq1, 2] * dq1 + Coefficient[AnI[[1]], dq1, 1];
AnI56 = Coefficient[AnI[[1]], dq2, 2] * dq2 + Coefficient[AnI[[1]], dq2, 1];
AnI57 = Coefficient[AnI[[1]], dq3, 2] * dq3 + Coefficient[AnI[[1]], dq3, 1];
AnI58 = Coefficient[AnI[[1]], dq4, 2] * dq4 + Coefficient[AnI[[1]], dq4, 1];
AnI65 = Coefficient[AnI[[2]], dq1, 2] * dq1 + Coefficient[AnI[[2]], dq1, 1];
AnI66 = Coefficient[AnI[[2]], dq2, 2] * dq2 + Coefficient[AnI[[2]], dq2, 1];
```

```
AnI67 = Coefficient[AnI[[2]], dq3, 2] * dq3 + Coefficient[AnI[[2]], dq3, 1];
AnI68 = Coefficient[AnI[[2]], dq4, 2] * dq4 + Coefficient[AnI[[2]], dq4, 1];
AnI75 = Coefficient[AnI[[3]], dq1, 2] * dq1 + Coefficient[AnI[[3]], dq1, 1];
AnI76 = Coefficient[AnI[[3]], dq2, 2] * dq2 + Coefficient[AnI[[3]], dq2, 1];
AnI77 = Coefficient[AnI[[3]], dq3, 2] * dq3 + Coefficient[AnI[[3]], dq3, 1];
AnI78 = Coefficient[AnI[[3]], dq4, 2] * dq4 + Coefficient[AnI[[3]], dq4, 1];
AnI85 = Coefficient[AnI[[4]], dq1, 2] * dq1 + Coefficient[AnI[[4]], dq1, 1];
AnI86 = Coefficient[AnI[[4]], dq2, 2] * dq2 + Coefficient[AnI[[4]], dq2, 1];
AnI87 = Coefficient[AnI[[4]], dq3, 2] * dq3 + Coefficient[AnI[[4]], dq3, 1];
AnI88 = Coefficient[AnI[[4]], dq4, 2] * dq4 + Coefficient[AnI[[4]], dq4, 1];
AnIspeed = {{AnI55, AnI56, AnI57, AnI58}, {AnI65, AnI66, AnI67, AnI68}, {AnI75, AnI76, AnI77, A
AnIrectangular = Join[DiagonalMatrix[{sinctheta1, sinctheta2, sinctheta3, sinctheta4}], AnIspeed, 2];
RicattiArectangular = Inverse[MatrixA].AnIrectangular;
RicattiArectangularId = Join[DiagonalMatrix[{0, 0, 0, 0}], DiagonalMatrix[{1, 1, 1, 1}], 2];
UberRicattiA = Join[RicattiArectangularId, RicattiArectangular];
UberRicattiB = Join[DiagonalMatrix[{0, 0, 0, 0}], Inverse[MatrixA].BnI];
RicattiVectorx = {theta1[t], theta2[t], theta3[t], theta4[t], dq1, dq2, dq3, dq4};
RicattiVectoru = {T1, T2, T3, T4};
finalfx = Join[fx1, fx2];
Ricattieq = UberRicattiA.RicattiVectorx + UberRicattiB.RicattiVectoru;
(*Virtual System*)
qv1 = thetav1[t];
qv2 = thetav2[t];
Gv10 = {lv1 * Cos[thetav1[t]], lv1 * Sin[thetav1[t]], 0};
Gv20 = {Lv1 * Cos[thetav1[t]] + lv2 * Cos[thetav2[t]], Lv1 * Sin[thetav1[t]] + lv2 * Sin[thetav2[t]], 0};
Gve0 = {Lv1 * Cos[thetav1[t]] + Lv2 * Cos[thetav2[t]], Lv1 * Sin[thetav1[t]] + Lv2 * Sin[thetav2[t]], th
Vgv10 = D[Gv10, t];
Vgv20 = D[Gv20, t];
VgvE0 = D[Gve0, t];
wv1 = {0, 0, D[thetav1[t], t]};
wv2 = {0, 0, D[thetav2[t] + thetav1[t], t]};
Jv1 = DiagonalMatrix[{Jxv1, Jyv1, Jzv1}];
Jv2 = DiagonalMatrix[{Jxv2, Jyv2, Jzv2}];
```

```
dqv1 = D[thetav1[t], t];
dqv2 = D[thetav2[t], t];
Tv = Simplify[(mv1 * Vgv10.Vgv10 + wv1.Jv1.wv1 + mv2 * Vgv20.Vgv20 + wv2.Jv2.wv2)/2];
Vv = 0;
Rv = 0;
Jacobv = D[Gve0, {{thetav1[t], thetav2[t]}}];
Jacobtransposev = Transpose[Jacobv];
Kv = {{k11v, k12v}, {k21v, k22v}};
Dv = {{d11v, d12v}, {d21v, d22v}};

DTDqv1p = Simplify[D[D[Tv, D[thetav1[t], t]], t]];
DTDqv2p = Simplify[D[D[Tv, D[thetav2[t], t]], t]];
DVDqv1 = Simplify[D[Vv, thetav1[t]]];
DVDqv2 = Simplify[D[Vv, thetav2[t]]];
DTDqv1 = Simplify[D[Tv, thetav1[t]]];
DTDqv2 = Simplify[D[Tv, thetav2[t]]];
DRDqv1p = Simplify[D[Rv, D[thetav1[t], t]]];
DRDqv2p = Simplify[D[Rv, D[thetav2[t], t]]];

Eqv1 = Simplify[DTDqv1p - DTDqv1 + DVDqv1 + DRDqv1p] - (Jacobtransposev.Force)[[1]] + Kv
Eqv2 = Simplify[DTDqv2p - DTDqv2 + DVDqv2 + DRDqv2p] - (Jacobtransposev.Force)[[2]] + Kv
ddqv1 = D[D[thetav1[t], t], t];
ddqv2 = D[D[thetav2[t], t], t];
av11 = Simplify[Coefficient[Eqv1, ddqv1, 1]];
av12 = Simplify[Coefficient[Eqv2, ddqv1, 1]];
av21 = Simplify[Coefficient[Eqv1, ddqv2, 1]];
av22 = Simplify[Coefficient[Eqv2, ddqv2, 1]];
bv1 = -Simplify[Eqv1/.{ddqv1 → 0, ddqv2 → 0}];
bv2 = -Simplify[Eqv2/.{ddqv1 → 0, ddqv2 → 0}];
MatrixAv = {{av11, av12}, {av21, av22}};
MatrixBv = {bv1, bv2};
fxv1 = {dqv1, dqv2};
fxv2 = Inverse[MatrixAv].MatrixBv + {ddthetav1i, +ddthetav2i};
```

```
AvnI = Simplify[MatrixBv];
AvnI33 = Coefficient[AvnI[[1]], dqv1, 2] * dqv1 + Coefficient[AvnI[[1]], dqv1, 1];
AvnI34 = Coefficient[AvnI[[1]], dqv2, 2] * dqv2 + Coefficient[AvnI[[1]], dqv2, 1];
AvnI43 = Coefficient[AvnI[[2]], dqv1, 2] * dqv1 + Coefficient[AvnI[[2]], dqv1, 1];
AvnI44 = Coefficient[AvnI[[2]], dqv2, 2] * dqv2 + Coefficient[AvnI[[2]], dqv2, 1];

AvnI31 = Coefficient[AvnI[[1]], qv1, 1];
AvnI32 = Coefficient[AvnI[[1]], qv2, 1];
AvnI41 = Coefficient[AvnI[[2]], qv1, 1];
AvnI42 = Coefficient[AvnI[[2]], qv2, 1];

sincthetav1 = Coefficient[AvnI[[1]], Sin[qv1], 1] * Sinc[qv1/Pi];
sincthetav2 = Coefficient[AvnI[[2]], Sin[qv2], 1] * Sinc[qv2/Pi];


betav1 = (Coefficient[AvnI[[1]], Cos[qv1], 1]Cos[qv1] + Coefficient[AvnI[[1]], thetav1i, 1] * thetav1i + C
betav2 = (Coefficient[AvnI[[2]], Cos[qv2], 1]Cos[qv2] + Coefficient[AvnI[[2]], thetav1i, 1] * thetav1i + C
betavmatrix = {{betav1}, {betav2}};

AvnIposition = {{AvnI31 + sincthetav1, AvnI32}, {AvnI41, AvnI42 + sincthetav2}};

AvnIspeed = {{AvnI33, AvnI34}, {AvnI43, AvnI44}};
AvnIrectangular = Join[Join[AvnIposition, AvnIspeed, 2], betavmatrix, 2];
TrueAvrectangular = Inverse[MatrixAv].AvnIrectangular + {{0, 0, 0, 0, ddthetav1i/beta[t]}, {0, 0, 0, 0
TrueAvrectangularId = Join[DiagonalMatrix[{0, 0}], DiagonalMatrix[{1, 1}], {{0}, {0}}, 2];
RicattiAv = Join[TrueAvrectangularId, TrueAvrectangular];
z = {qv1, qv2};
(*Errorfunction = (Jacob.fx1 - fz1)^2*)
Errorfunction = FullSimplify[(Jacob.fx1 - Jacobv.fz1).(Jacob.fx1 - Jacobv.fz1)];
(*simplecalculusspeedweightmatrixidentity[xz] * [J^TJ, -J^T; -J, I][x; z]; halfweight = [03x4J03x3
zm4x4 = ConstantArray[0, {4, 4}];
zm3x3 = ConstantArray[0, {3, 3}];
zm3x2 = ConstantArray[0, {3, 2}];
```

```
zm4x3 = ConstantArray[0, {4, 3}];
zm6x4 = ConstantArray[0, {6, 4}];
zm3x1 = ConstantArray[0, {3, 1}];
zm3x4 = ConstantArray[0, {3, 4}];
I3x3 = IdentityMatrix[3];
halfweight = Join[zm3x4, Jacob, zm3x2, -Jacobv, zm3x1, 2];
fullweight = alpha1 * Transpose[halfweight].halfweight;
errorstatederivative = D[Ge0 - Gve0, t];
ep1 = errorstatederivative[[1]];
ep2 = errorstatederivative[[2]];
ep3 = errorstatederivative[[3]];
ep1v = Coefficient[ep1, {Join[q, qp, z, zp, {beta[t]}]}];
ep2v = Coefficient[ep2, {Join[q, qp, z, zp, {beta[t]}]}];
ep3v = Coefficient[ep3, {Join[q, qp, z, zp, {beta[t]}]}];
Aerror = Join[ep1v, ep2v, ep3v, 1];
Berror = ConstantArray[0, {3, 4}];
Qerror = DiagonalMatrix[{alpha0, alpha0, alpha0}];
zm3x16 = ConstantArray[0, {3, 16}];
zm16x3 = ConstantArray[0, {16, 3}];
(*Ricatti Join*)
zm8x6 = ConstantArray[0, {8, 6}];
zm8x4 = ConstantArray[0, {8, 4}];
zm5x8 = ConstantArray[0, {5, 8}];
zm2x1 = ConstantArray[0, {2, 1}];
zm2x2 = ConstantArray[0, {2, 2}];
zm6x8 = ConstantArray[0, {6, 8}];
zm8x3 = ConstantArray[0, {8, 3}];
zm3x7 = ConstantArray[0, {3, 7}];
zm5x4 = ConstantArray[0, {5, 4}];
zm4x1 = ConstantArray[0, {4, 1}];
betacolumnphysys = Join[zm4x1, betacolumn, 1];
I2x2 = IdentityMatrix[2];
fz12x5 = Join[zm2x2, I2x2, zm2x1, 2];
```

```
fz23x5 = fzend3x5;
fz5x5 = Join[fz12x5, fz23x5, 1];
HalfRicattiAx = Join[Join[UberRicattiA, zm8x4, 2], betacolumnphysys, 2];
HalfRicattiAz = Join[zm5x8, fz5x5, 2];
FullRicattiA = Join[HalfRicattiAx, HalfRicattiAz, 1];
HalfRicattiBx = UberRicattiB;
HalfRicattiBz = zm5x4;
FullRicattiB = Join[HalfRicattiBx, HalfRicattiBz, 1];
(*Error Join*)
zm16x3 = ConstantArray[0, {16, 3}];
EndRicattiA = Join[Join[FullRicattiA, Aerror, 1], zm16x3, 2];
EndRicattiB = Join[FullRicattiB, Berror, 1];
(*Join*)
zm13x3 = ConstantArray[0, {13, 3}];
zm3x13 = ConstantArray[0, {3, 13}];
QRicatti = Join[Join[fullweight, zm3x13, 1], Join[zm13x3, Qerror, 1], 2];
FinalEquilibrium = {theta1eq, theta2eq, theta3eq, theta4eq, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
(*Impedance weight matrices*)
E0 = Join[halfweight, zm3x3, 2];
dE0dt = D[E0, t];
Re2 = alpha2 * Transpose[EndRicattiB].Transpose[E0].E0.EndRicattiB;
Ne2 = Transpose[dE0dt].E0.EndRicattiB;
Ru = DiagonalMatrix[{alphau, alphau, alphau, alphau}];
Rend = Ru + Re2;
EndRicattiN = Ne2;
(*Final differential equation*)
FinalDiff = Join[finalfx, finalfz, {ep1, ep2, ep3}];
(*Parameters*)
L1 = 0.25;
L2 = 0.25;
L3 = 0.34;
L4 = 0.08;
Lv1 = 0.31;
```

```
Lv2 = 0.42;
c1 = 0.4;
c2 = 0.4;
c3 = 0.4;
c4 = 0.4;
m1 = 0.8 * L1;
m2 = 0.8 * L2;
m3 = 0.8 * L3;
m4 = 0.8 * L4;
mv1 = 1.93;
mv2 = 1.52 + 0.52;
Jz1 = m1 * L1^2/4;
Jz2 = m2 * L2^2/4;
Jz3 = m3 * L3^2/4;
Jz4 = m4 * L4^2/4;
Jzv1 = 0.0141;
Jzv2 = 0.0210;
lv1 = 0.165;
lv2 = 0.21;
l1 = L1/2;
l2 = L2/2;
l3 = L3/2;
l4 = L4/2;
(*Error matrices replacement*)
CosSinreplacer = {Cos[theta1[t] - theta2[t]] → cs12, (Cos[theta1[t] - theta3[t]]) → cs13, (Cos[theta1
replacer = {theta1[t] → u[1], theta2[t] → u[2], theta3[t] → u[3], theta4[t] → u[4], dq1 → u[5], dq2 → u
replacerpurefz = {qv1 → u[1], qv2 → u[2], dqv1 → u[3], dqv2 → u[4]};
de0dt = dE0dt/.CosSinreplacer/.replacer;
e0 = E0/.CosSinreplacer/.replacer;
EndRicattiQ = QRicatti;
(*Export Equations to Matlab*)

MatlabFx = FinalDiff/.CosSinreplacer/.replacer;
```

```
MatlabFzvirtual = Join[finalfznobeta]/.CosSinreplacer/.replacer;
MatlabRicattiA = EndRicattiA/.CosSinreplacer/.replacer;
MatlabRicattiB = EndRicattiB/.CosSinreplacer/.replacer;
MatlabEquilibrium = FinalEquilibrium/.replacer;
MatlabQ = EndRicattiQ/.CosSinreplacer/.replacer;
MatlabRicattiR = Rend/.CosSinreplacer/.replacer;
MatlabRicattiN = EndRicattiN/.CosSinreplacer/.replacer;
write1 = OpenWrite["MatlabRicattiA.txt"];
WriteMatlab[MatlabRicattiA, write1];
Close[write1];
write2 = OpenWrite["MatlabRicattiB.txt"];
WriteMatlab[MatlabRicattiB, write2];
Close[write2];
write5 = OpenWrite["fx.txt"];
WriteMatlab[MatlabFx, write5];
Close[write5];
write6 = OpenWrite["fzonly.txt"];
WriteMatlab[MatlabFzvirtual, write5];
Close[write6];
writee0 = OpenWrite["e0.txt"];
WriteMatlab[e0, writee0];
Close[writee0];
writede0dt = OpenWrite["de0dt.txt"];
WriteMatlab[de0dt, writede0dt];
Close[writede0dt];
writeQe1 = OpenWrite["Qe1.txt"];
WriteMatlab[MatlabQ, writeQe1];
Close[writeQe1];
```

APPENDIX D – Numerical Simulation Codes

D.1 Ordinary Differential Equation and Input Computation Code

The following code, written in Matlab, represents the ordinary differential equation of the system, and computes the control inputs.

```
% ODE and SDRE function

function dydt = Ffunction( t,y )


global zx0 zy0 zphi0 l1 l2 l3 l4 m1 m2 m3 m4 L1 L2 L3 L4 ...
    Kx Ky Kphi cx cy cphi mx my mphi Jx1 Jx2 Jx3 Jx4 Jy1 Jy2 Jy3
        Jy4 ...
    Jz1 Jz2 Jz3 Jz4 g c1 c2 c3 c4 alpha Fx Fy k11v k22v k12v
       k21v ...
    Jzv1 Jzv2 mv1 mv2 lv1 lv2 Lv1 Lv2 thetav1i thetav2i
       dthetav1i dthetav2i ddthetav1i ddthetav2i d11v d22v d12v
       d21v ddtheta1vi ddtheta2vi n uhistory tant ...
    ulocal ubefore K k utime globaltime T1 T2 T3 T4 alpha2 A
       disturbon forceon


alpha2=0*exp(-100*t);
externalforces=forceon*forceext(t);
Fx=externalforces(1);
Fy=externalforces(2);


refer=ctj(0,0,t);


thetav1i=refer(1);
```

```
23 thetav2i=refer(2);
24
25 dthetav1i=refer(3);
26 dthetav2i=refer(4);
27
28 ddthetav1i=refer(5);
29 ddthetav2i=refer(6);
30
31 tautot=[Fy*Lv1*cos(y(9)) - Fx*Lv1*sin(y(9));
32  Fy*Lv2*cos(y(10)) - Fx*Lv2*sin(y(10))];
33
34 taue=tautot(2);
35 taus=tautot(1);
36
37 k11v =10.8 + 3.18*abs(taus);
38 k12v =2.83 + 2.15*abs(taue);
39 k21v = 2.51 + 2.34*abs(taue);
40 k22v = 8.67 + 6.18*abs(taue);
41 cv1v2=cos(y(9)-y(10));
42
43
44
45 MK=[104.812+13.584*cv1v2+0.86*abs(taue) + 10.32*cv1v2*abs(taue)
      + 30.528*abs(taus), 13.6024+ 51.84*cv1v2 + 7.052*abs(taue) +
      1.272*abs(taus) + 15.264*cv1v2*abs(taus);
46      27.564+ 41.616*cv1v2 + 24.936*abs(taue) + 29.664*cv1v2*abs(
          taue), 29.4416+ 12.048*cv1v2 + 21.2064*abs(taue) +
          11.232*cv1v2*abs(taue)];
47 Dv=0.26*MK^(1/2);
48
49 d11v=Dv(1,1);
50 d12v=Dv(1,2);
```

```
51 d21v=Dv(2,1);
52 d22v=Dv(2,2);
53
54
55
56 % dthetav1i=0*thetas(3);
57 % dthetav2i=0*thetas(4);
58 %
59 % ddthetav1i=0*thetas(5);
60 % ddthetav2i=0*thetas(6);
61 %% ddthetav1i=ddtheta(1);
62 % ddthetav2i=ddtheta(2);
63 t
64
65
66 A=RicattiA(y);
67 B=RicattiB(y);
68
69
70
71
72 freq=0.01; %update frequency
73
74
75 if t>=k*freq;
76
77
78 Q=RicattiQ(y);
79 N=RicattiN(y);
80 R=RicattiR(y);
81 % Q=diag([0;0;0;0;0;0;0;0;0;0;0;0;0;alpha;0.001;alpha]);
82
```

```
83
84 K=lqr(A,B,Q,R,N);
85
86 u=-K*(y);
87 u(5:end)=0; %nullifying fake controls
88 k=k+1;
89 else
90
91       u=-K*(y);
92       u(5:end)=0; %nullifying fake controls
93
94 end
95
96
97
98
99 % if t>5 && t<10
100 % disturb=distubon*disturbances(t);
101 % end
102
103 T1=u(1)+disturb(1);
104 T2=u(2)+disturb(2);
105 T3=u(3)+disturb(3);
106 T4=u(4)+disturb(4);
107
108 u(1:4)=[T1;T2;T3;T4];
109
110
111 utime(:,n)=u;
112 globaltime(n)=t;
113 dydt=A*(y)+B*u;
114
```

```
115
116 n=n+1;
117 end
```

D.2 Numerical Integrator Code

The following code, written in Matlab, numerically integrates the system equations and plot the relevant variables as a function of time.

```
1  % Calls function to solve the problem
2
3  clear all
4  clc
5  close all
6  global zx0 zy0 zphi0 l1 l2 l3 l4 m1 m2 m3 m4 L1 L2 L3 L4 ...
7      Kx Ky Kphi cx cy cphi mx my mphi Jx1 Jx2 Jx3 Jx4 Jy1 Jy2 Jy3
           Jy4 ...
8      Jz1 Jz2 Jz3 Jz4 g c1 c2 c3 c4 alpha Fx Fy k11v k22v k12v
          k21v ...
9      Jzv1 Jzv2 mv1 mv2 lv1 lv2 Lv1 Lv2 thetav1i thetav2i
          dthetav1i dthetav2i ddthetav1i ddthetav2i d11v d22v d12v
          d21v ddtheta1vi ddtheta2vi n uhistory tant k ubefore
          ulocal K utime globaltime disturbon forceon
10 parameters
11 ubefore=[0;0;0;0;0;0;0;0;0;0];
12 ulocal=[0;0;0;0;0;0;0;0;0;0];
13 tant=-1;
14 k=0;
15
16
17
18
19 n=1;
```

```
20 uhistory(n,:)=zeros(1,7);
21
22
23
24 dummy=ctj(0,0,0);
25
26
27 vang1i=dummy(1);
28 vang2i=dummy(2);
29
30
31
32 %% simulation I- correct initial position
33
34 % ang1i=-2.6469;
35 % ang2i=-4.4218;
36 % ang3i=0.6796;
37 % ang4i=2.5962;
38 %
39 % disturbon=0;
40 % forceon=1;
41
42
43 %% simulation II- wrong initial position disturb on
44
45 % ang1i=-2.6469;
46 % ang2i=-4.4218;
47 % ang3i=0.6796;
48 % ang4i=2.5962;
49
50 % disturbon=1;
51 % forceon=1;
```

```
52 % ang1i=-pi;
53 % ang2i=-pi;
54 % ang3i=0;
55 % ang4i=pi;
56 %% simulation III- wrong initial position disturb on
57
58 % ang1i=-2.6469;
59 % ang2i=-4.4218;
60 % ang3i=0.6796;
61 % ang4i=2.5962;
62
63 % disturbon=1;
64 % forceon=1;
65 % ang1i=pi;%pi/4;
66 % ang2i=pi;%pi/4;
67 % ang3i=0;%pi-pi/4;
68 % ang4i=pi;%pi-pi/4;
69
70 %%
71 %
72 %% simulation IV- right initial position disturb off
       repeatability
73
74  ang1i=-2.6469;
75  ang2i=-4.4218;
76  ang3i=0.6796;
77  ang4i=2.5962;
78
79  disturbon=0;
80  forceon=0;
81
82 %% Call the function
```

```
83  dvang1i=0;
84  dvang2i=0;
85
86
87  angi(1,:)=[ang1i;ang2i;ang3i;ang4i];
88  vangi(1,:)=[vang1i;vang2i];
89
90  erroinit=rq(angi)-rv(vangi);
91
92  y0=[ang1i;ang2i;ang3i;ang4i;0;0;0;0;vang1i;vang2i;dvang1i;
        dvang2i;5;erroinit(1);erroinit(2);erroinit(3)];
93
94
95
96  cont=1;
97  y(1,:)=y0;
98  deltat=0.0001;
99  t=0;
100 tn=0;
101 endtime=11;
102
103 %% ruge kutta 4th
104
105 while t(cont)<endtime
106
107
108     h=deltat-deltat/2*exp(-tn);
109     tn=t(cont);
110     yn=y(cont,:);
111
112     dy=Ffunctionschedule(t(cont),y(cont,:)');
113     k1=dy';
```

```
114         k2=Ffunctionschedule(tn+h/2,(yn+h*k1/2)')';
115         k3=Ffunctionschedule(tn+h/2,(yn+h*k2/2)')';
116         k4=Ffunctionschedule(tn+h,(yn+h*k3)')';
117         dydt(cont,:)=dy';
118
119         y(cont+1,:)=yn+h/6*(k1+2*k2+2*k3+k4);
120         t(cont+1)=tn+h;
121         cont=cont+1;
122
123 end
124
125
126 %%
127
128
129
130 armmotion=rv(y(:,9:10));
131 robotmotion=rq(y(:,1:4));
132
133 zideal=y(:,9:10);
134
135 zpideal=y(:,11:12);
136
137 save('simulation.mat')
138
139 figure(1)
140 plot(t,zideal)
141 title('angle arm')
142 legend('1','2','3','4','5','6')
143
144 figure(2)
145 plot(t,zpideal)
```

```
146 title('velocity arm')
147 legend('1','2','3','4','5','6')
148
149 figure(3)
150 plot(t,y(:,1:4))
151 title('Joint Coordinates of 4DOF manipulator: SDRE Case')
152 legend('Joint 1','Joint 2','Joint 3','Joint 4','5','6')
153 xlabel('t(s)')
154 ylabel('Joint coordinates(rad)')
155
156 figure(4)
157 plot(t,y(:,5:8))
158 title('angular speed robot')
159 legend('1','2','3','4','5','6')
160
161
162 figure(5)
163 plot(t,y(:,13))
164 title('beta')
165 legend('1','2','3','4','5','6')
166
167 figure(7)
168 plot(t,y(:,14:16))
169 title('error position')
170 legend('1','2','3','4','5','6')
171
172 figure(8)
173 plot(t,robotmotion)
174 hold on
175 plot(t,armmotion)
176 title('Position on plane: 4DOF Manipulator End-effector X "Human
        Wrist/Hand" - SDRE Case')
```

```
177 legend('x-4DOF Manipulator (m)','y-4DOF Manipulator (m)','
    Orientation-4DOF Manipulator (rad)','x-2DOF Human Arm (m)','y
    -2DOF Human Arm (m)','Orientation-2DOF Human Wrist/Hand (rad)
    ')
178 xlabel('t(s)')
179
180
181 figure(9)
182 plot(globaltime,utime(1:4,:)')
183 title('Torques applied at joints- SDRE Case')
184 legend('Joint 1','Joint 2','Joint 3','Joint 4')
185 xlabel('t(s)')
186 ylabel('Control Input (Nm)')
187
188
189 figure(10)
190 plot(t,robotmotion-armmotion)
191 title('true error')
```