Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia

COPPE UFRJ

MODELING, CONTROL AND ELECTROMECHANICAL DESIGN OF A MODULAR LIGHTWEIGHT MANIPULATOR FOR INTERACTION AND INSPECTION TASKS

Marco Fernandes dos Santos Xaud

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadores: Ramon Romankevicius Costa
Antonio Candea Leite

Rio de Janeiro
Março de 2016

MODELING, CONTROL AND ELECTROMECHANICAL DESIGN OF A MODULAR LIGHTWEIGHT MANIPULATOR FOR INTERACTION AND INSPECTION TASKS

Marco Fernandes dos Santos Xaud

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Ramon Romankevicius Costa, D.Sc.

Prof. Antonio Candea Leite, D.Sc.

Prof. Alessandro Jacoud Peixoto, D.Sc.

Prof. Marco Antonio Meggiolaro, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
MARÇO DE 2016

Xaud, Marco Fernandes dos Santos

Modeling, Control and Electromechanical Design of a Modular Lightweight Manipulator for Interaction and Inspection Tasks/Marco Fernandes dos Santos Xaud. – Rio de Janeiro: UFRJ/COPPE, 2016.

XIX, 177 p.: il.; 29, 7cm.

Orientadores: Ramon Romankevicius Costa

Antonio Candea Leite

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2016.

Bibliografia: p. 158 – 168.

1. Manipuladores robóticos seriais. 2. Controle cinemático de manipuladores. 3. Projeto eletromecânico de manipuladores. 4. Controle híbrido de força e posição. 5. Singularidades cinemáticas. I. Costa, Ramon Romankevicius *et al.*. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Dedico essa dissertação à minha querida avó Margarida, minha segunda mãe, meu exemplo de vida. Te amo eternamente!

Agradecimentos

Agradeço primeiramente a minha família, por todo o amor depositado em mim em todos os anos de minha vida.

Em especial, a minha mãe (e melhor amiga) Rita de Cássia, a meu pai Celso, as minhas irmãs Ananda e Clara (apesar de terem pavor de ciências exatas), e a meus avós maternos Margarida e Joaquim, por sempre me incentivarem a lutar por meus objetivos, por acreditarem em mim, torcerem pelo meu sucesso, e por serem o meu maior exemplo de vida. Agradeço a eles por todo o carinho recebido e pelos valores compartilhados ao longo da vida. Tenho certeza que chegar até aqui não teria valor algum se não fosse por vocês. Também não posso deixar de agradecer meus primos (e a minha prima emprestada Alana Thomé) pela amizade e pelo companheirismo de sempre e que são muito importantes em minha vida.

Agradeço ao meu professor e orientador Ramon Romankevicius Costa pela oportunidade ímpar de me aceitar como aluno orientado de mestrado e como engenheiro do LEAD. A aplicação da área acadêmica em projetos de pesquisa e desenvolvimento é algo que acredito, e infelizmente isso é uma realidade ainda distante e pouco estimulada no Brasil. Felizmente, essa parece ser a proposta dos projetos do LEAD, que estimulam seus alunos a trabalhar e desenvolver dissertações e teses relacionadas aos mesmos projetos que desenvolvem.

Agradeço ao meu orientador Antonio Candea Leite pela disponibilidade, confiança, parceria e amizade que possibilitaram o desenvolvimento desta dissertação.

Agradeço aos professores Liu Hsu, Fernando Lizarralde e Eduardo Nunes pelos valiosos ensinamentos que foram de suma importância tanto para esta pesquisa quanto para a minha formação.

Agradeço ao amigo Rodrigo Fonseca Carneiro e ao professor e amigo Alessandro Jacoud Peixoto pelo companheirismo, pela paciência e pelas incontáveis contribuições técnicas que ajudaram muito no desenvolvimento de meus conhecimentos em eletrônica, e, consequentemente, para o desenvolvimento da eletrônica do

DORIS.

Agradeço a meus grandes parceiros que me acompanham desde a graduação: os cavaleiros de Atena Guilherme Carvalho (Zeka, o eterno lesionado) e Renan Freitas (o eterno rabugento). Também agradeço aos meus grandes amigos do LEAD, especialmente Alex Neves (o eterno guru), Ighor Marcovistz, Bruno Campello, Luciana Reys, Gabriel Lira, Raphael Gago, Eduardo Elael, Alana Monteiro, Julia Campana, Gabriel Alcantara, Mariana Rufino, Fernando Coutinho (Tico) e Henrique Faria, por todos esses anos de amizade, companheirismo e clima de trabalho descontraído, seja no laboratório ou até em empregos passados.

Agradeço a minha grande amiga Regina Alves, que me acompanha em todas as jornadas há muito tempo, começando pelo colégio, depois a graduação, e por fim, as inesquecíveis caronas que lhe dei nos primeiros anos de mestrado e que me colocaram no honrado posto de *pessoa mais importante da sua vida*. Sua amizade foi e sempre será muito importante na minha vida.

E agradeço à COPPE pela oportunidade de estudar em uma instituição de pesquisa de excelência e prestigiada em todo o mundo.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

MODELAGEM, CONTROLE E PROJETO ELETROMECÂNICO DE UM MANIPULADOR LEVE E MODULAR PARA TAREFAS DE INTERAÇÃO E INSPEÇÃO

Marco Fernandes dos Santos Xaud

Março/2016

Orientadores: Ramon Romankevicius Costa
Antonio Candea Leite

Programa: Engenharia Elétrica

A robótica vem aumentando cada vez mais seu potencial para solucionar demandas encontradas na indústria, incluindo a melhoria das condições ambientais, de saúde e segurança, bem como o aumento da eficiência e produtividade. Particularmente, diversas empresas de petróleo e gás já usam soluções automatizadas em suas instalações, mas recentemente, o uso de manipuladores vem chamando a atenção para tarefas mais específicas e que requerem uma complexa interação com o ambiente. Apesar dos modelos comerciais existentes, as restrições de projeto e orçamento podem motivar a construção de um manipulador customizado. Esse trabalho apresenta o projeto eletromecânico, modelagem e controle de um manipulador para o sistema DORIS, um robô guiado por trilhos para inspeção de instalações *offshore*. Este manipulador deve posicionar autonomamente um sensor de vibração sobre a superfície externa de equipamentos próximos, interagir com painéis *touchscreen*, e movimentar uma câmera para obter diferentes ângulos da plataforma. Para fins de implementação prática, estratégias de controle cinemático e híbrido de força/posição são combinadas e posteriormente simuladas em ambiente computacional. Depois, são propostas mudanças na estrutura cinemática e leis de controle originais para lidar com o problema de singularidades cinemáticas, e minimizar o consumo de energia, utilizando o método da Inversa Filtrada.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

MODELING, CONTROL AND ELECTROMECHANICAL DESIGN OF A MODULAR LIGHTWEIGHT MANIPULATOR FOR INTERACTION AND INSPECTION TASKS

Marco Fernandes dos Santos Xaud

March/2016

Advisors: Ramon Romankevicius Costa
Antonio Candea Leite

Department: Electrical Engineering

Robots have been increasing their potential to overcome several problems found in industry, including process safety, personnel health, efficiency and productiveness. Many O&G companies already use automated solutions, but recently, robotic manipulators have been drawing more attention, since they can perform specific tasks that require complex environment interaction. In spite of the available commercial models, occasionally, the development of an own and dedicated arm is motivated by project constraints and reduced budget. This work presents the electromechanical design and control of a lightweight manipulator for DORIS system, which is a rail-guided robot in development stage for monitoring and inspection of offshore facilities. DORIS arm is meant to pose autonomously a vibration sensor in contact with platform nearby equipment, interact with *touchscreen* panels and movement a small camera to obtain different sights of the platform. For this, hybrid force/position and kinematic control strategies are combined and simulated in computing environment. Further, modifications in the original kinematic structure and control laws are proposed, aiming to deal with kinematic singularities, and minimize energy consumption using the Filtered Inverse method.

Contents

List of Figures

List of Tables

Acronyms

AUVs	Autonomous Underwater Vehicles.
DH	Denavit-Hartenberg.
DHM	Denavit-Hartenberg Modified.
DHS	Denavit-Hartenberg Standard.
DLS	Damped Least Squares.
DoF	Degrees-of-Freedom.
FI	Filtered Inverse.
FIK	Feedback Inverse Kinematics.
MIMO	Multiple-Input Multiple-Output.
ROVs	Remotely Operated Vehicles.
SISO	Single-Input Single-Output.

Chapter 1

Introduction

Robots have the potential to overcome several demands found in industry, specially Oil & Gas offshore facilities. Among these problems, process safety, personnel health, efficiency and productiveness are the most troubling. Many producing companies already use automated solutions for both *subsea* and *topside* environments. Recently, the utilization of robotic arms is getting great importance in this scenario, since they can perform very specific tasks that require a complex interaction with the environment (From 2010). Many available commercial models can realize a large variety of tasks. Occasionally, more specific tasks, project constraints and reduced budget motivate the design and construction of an ad-hoc customized manipulator. This is a challenging problem, and requires a very reliable design of its structure and robust control of the manipulator motion and interaction with objects. This work presents the modeling, control and electromechanical design of a lightweight manipulator for the DORIS robotic system, which is a rail-guided robot in development stage for monitoring, inspection, supervision and surveillance of the topside of offshore facilities (Freitas et al. 2015). DORIS manipulator has the accounts for posing a vibration sensor in contact with platform nearby machinery and interacting with *touchscreen* panels, either in teleoperated or autonomous operational modes, as well as moving a small camera through different angles of the platform. To achieve this goals, hybrid force/position and orientation control schemes based on the kinematic control approach are combined and simulated in computing environment. Furthermore, modifications in the manipulator original structure and control laws are proposed, aiming to: deal with kinematic singularities and ill-conditioned Jacobian matrices as well as optimize a cost function using the Filtered Inverse method (Vargas 2013).

1.1 Overview of offshore robotics

In spite of sporadic or regional crises, the Oil & Gas and petrochemical industry is always growing throughout the world. According to World-nuclear (2012), the demand will grow in the next decades, and so will do all the operation costs (extraction, pre-processing, refine and logistics), specially in *offshore facilities*[1], since the production is expected to be extended gradually to unexplored and inaccessible sites. The working conditions found in this locations - such as heavy rains, gusty winds, extreme temperatures, explosive atmosphere, corrosive and toxic chemicals and confined space - are adverse and will become even more obstacles to the Oil & Gas companies and for the expansion of this economic activity.

Currently, oil and gas companies are getting ready for this scenario by developing new technologies that enable the production at these sites, and consequently, increase the safety of its processes and employees (Freitas et al. 2015). Therefore, companies aim to transform originally *marginal fields* on *commercial fields.* A marginal field refers to an offshore area belonging to a given company whose investment, at a given moment in history, does not generate enough profit to justify its exploration and production, maybe due to economic crisis or even to the absence of appropriate technology for that activity. However, if the technical or economic conditions change, such a field can become commercial.

In this scenario, process automation and robotics are the technologies whose application is growing faster. The use of robots for carrying out inspection tasks, maintenance, repair and supervision of offshore facilities can dramatically increase the efficiency of operations on platforms, improve the safety and health of involved personnel, upgrade the safety of the process and the installation, and yet lower operating and logistics costs.

In the specific case of Brazil, Petrobras has recently discovered huge oil reserves at the pre-salt layer along the Brazilian shore, which, according to Ferro & Teixeira (2009*a*), are located approximately 300 km away from the coastline and from 5000 to 7000 meters depth. The economic importance of these discoveries and the challenge of access and production at these extremely remote areas motivate the development of an offshore production system with high level of automation.

[1] *Onshore* - installations over the land. *Offshore* - installations abroad the sea.

1.1.1 Offshore robots

Some late forecasts, such as Skourup & Pretlove (2009) and From (2010), point out significative advantages brought by the increase of automation and robotics in future oil production fields. The use of robots can reduce the maintenance cost of several sensors or devices that would be deployed throughout an oil facility. For example, in the case of implementing visual monitoring of a whole platform, it would be preferred the use of a robot which could take one camera to every place rather than the deployment of several cameras in different places, since the same goal would be achieved with maintenance cost of only one camera. In addition, the use of robots can lead to decrease in human repetitive operations, increase in productivity and efficiency, and upgrade in Health, Safety and Environment (HSE) conditions. The use of robots in platforms has the potential to replace humans in those tasks that are repetitive and performed under hazardous, unhealthy and confined circumstances (Shukla & Karki 2013).

However, these harsh conditions also pose great technical challenges that these robots must overcome (Chen et al. 2014), such as: (i) *hazardous atmosphere*, which is unfriendly, since it concentrates hydrocarbon gases and other chemicals that are potentially explosive, even with the advent of just a spark; (ii) *dangerous agents*, such as splashy salty water, corrosive chemicals, smoke and radiation; (iii) *extreme weather*, such as storms, gusty winds, hail, snow, direct sunlight, heat emitted from equipment, extreme temperatures - which can vary from -30°C to 50°C - and extreme relative air humidity (nearby 100%); (iv) *constrained space and complex structural paths*, such as pipes, flanges, valves, stairways, tanks and compressors.

Nowadays, the majority of the Oil & Gas robots are Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs) that are mostly being used for *subsea* operations, which include seabed mapping, drilling, inspection and repair underwater equipment, risers, pipelines (Faber Archila & Becker 2013), moorings and anchors. However, the late researches has been focusing on robotic system applications on platform topsides[2] aiming to perform monitoring, supervision, inspection, maintenance and intervention tasks, such as: (i) monitoring of process indicators (pressure, level, flow and temperature); (ii) supervision of machine health of platform equipment; (iii) detection and diagnosis of gas/fluid leakage, acoustic

[2]In an offshore facility, the *topside* is the part located above the sea level and away the splash zone. Generally, the topside is the part that accommodates all the platform equipment.

anomalies, fire and smoke; (iv) inspection of tanks, ovens, compressors, pumps, vessels and towers; (v) manipulation of valves, switches, buttons and levers; (vi) surveillance of non-authorized personnel and abandoned objects.

Among the most known topside robots, MIMROex (Bengel & Pfeiffer 2007) was developed and tested by the Fraunhofer Institute of Manufacturing Engineering and Automation (IPA), and designed to navigate safely through offshore environments and perform inspection tasks autonomously (Figure 1.1a). SINTEF Topside Robotic System (Kyrkjebø et al. 2009) was developed at the Robotics Laboratory Facility, in Trondheim, Norway, and was designed to be a complex and smart instrumentation system to aid onshore operators in monitoring and controlling of the processes of the platform. Sensabot (NREC/CMU 2012), developed by the Carnegie Mellon University, was designed for teleoperated inspection, and certified for operation in harsh weather and toxic, flammable and explosive environments. These robots move around over the platform floor or equipment using wheels and many Degrees-of-Freedom (DoF)s, which make their control design and autonomy very complex to be implemented.

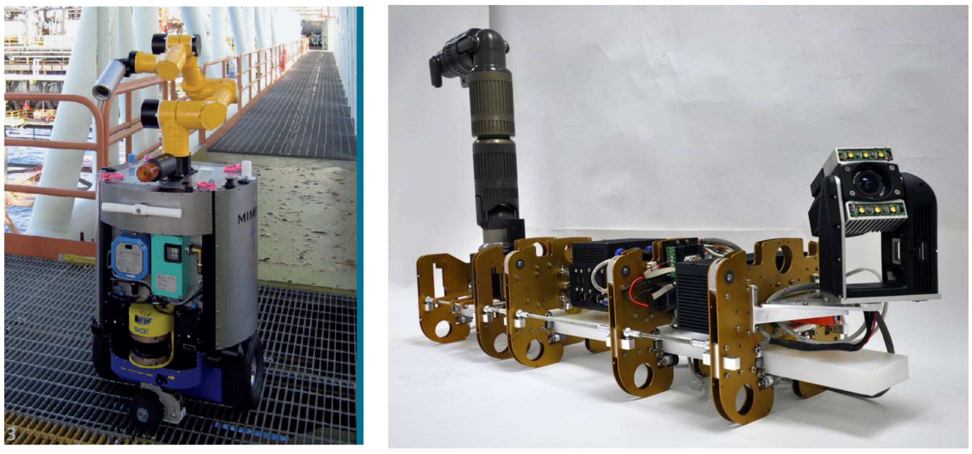


Figure 1.1: Offshore robots: (a) MIMROex from Fraunhofer; (b) ARTIS from DFKI.

Recently, rail-guided robots, such as ARTIS “Autonomous Railguided Tank Inspection System” (GmbH 2011), from DFKI (German Research Center for Artificial Intelligence - Bremen, Germany), have been proposed as a solution to overcome these problems (Figure 1.1b). This concept proposes that the robot moves constrained only along a pre-mounted rail, whose layout should be designed so as to lead the robot until all the desired places it should visit. Since the locomotion occurs only along the rail, the project complexity is merely comprised into the robot itself. Therefore, even though the robot has only one DoF, its reach is customizable and extensible to mostly all the platform sites, depending only on the rail path design. Many researches, such as those from Christensen et al. (2011a,b), have used ARTIS to solve real case study situations, such as ballast water tank inspection. Another research from Borgerink et al. (2014) considers ARTIS with an embedded manipulator for coating tasks and investigates the influence of rail compliance on the end-effector position error due to the ship movement.

1.1.2 DORIS robotic system

Following this trend, a new project called DORIS[3], which is being developed by Federal University of Rio de Janeiro, presents a rail-guided robot, in which independent and functional wagons carry several cameras, sensors and devices to monitor, supervise and inspect different areas and equipment located at the platform topside (Figure 1.2).

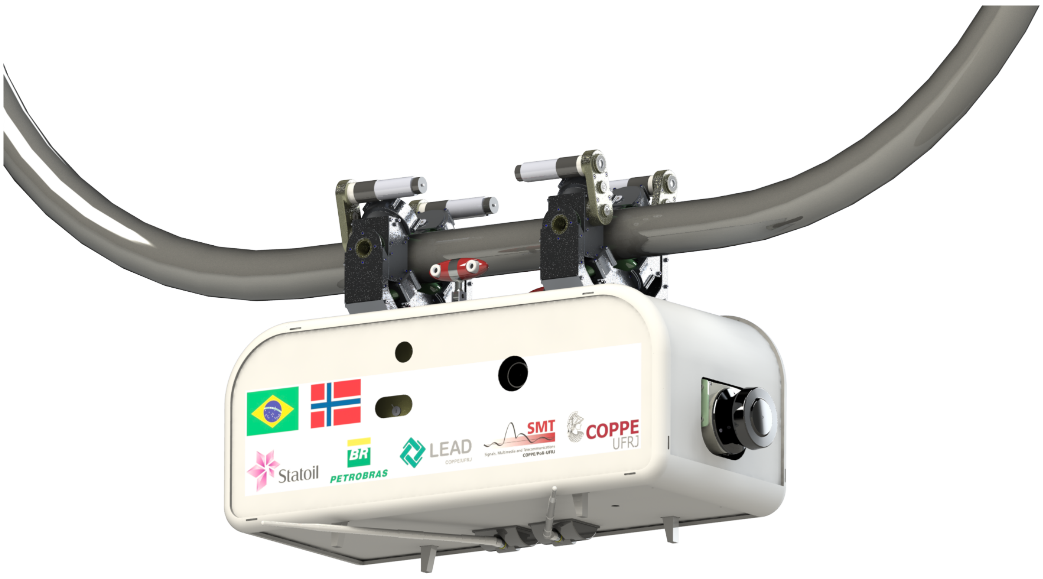


Figure 1.2: 3D model of DORIS robot on rail.

Among this robot functionalities, there are: monitoring of equipment temperature pattern using infrared thermal cameras, surveillance of non-authorized personnel presence or access, detection of audible anomalies using microphone arrays,

[3]DORIS project is supported primarily by Petrobras S.A. and Statoil Brazil Oil & Gas Ltda under contract COPPETEC 0050.0079406.12.9 (ANP-Brazil R&D Program), and in part by the Brazilian research agencies CNPq and FAPERJ.

detection of gas leakages using hydrocarbon sensor, vibration pattern inspection of critical machinery, interaction with *touchscreen* interfaces throughout the platform, posterior or *in loco* processing of collected data, storage and *real-time* transmission of video and audio, automatic alarm emission upon dangerous situations or diagnosed faults, and pre-programming of routine tasks.

Challenges: *touchscreen*, vibration inspection and camera orientation

Some of these tasks described above offer an additional challenge for implementation. The interaction with *touchscreens* requires extension and flexibility of the robot reach. DORIS cameras remains always fixed during the robot operation which makes difficult to get videos from different angles and points of view of the platform.

The vibration inspection of critical equipment is essential to the plant health, since it gives an early prognostic of possible equipment abnormal shakes that may cause an irreversible mechanical brake and, hence, lead to a a catastrophic plant halt (Mathas 2012). In rotating machines, such as air/gas compressors and pumps, this monitoring can assist the identification of mechanical faults, such as unbalance and misalignment (de Lima et al. 2013).

Vibration measurement depends on what it is required for (Emerson 2014), and it can be performed by several techniques that offers a *trade-off* between accuracy and non-intrusiveness (Wilson 1999*a*,*b*,*c*). The vibration sensor selected for DORIS robot is an intrusive and accurate piezoelectric crystal based accelerometer, which requires to be positioned totally perpendicular with respect to the machine surface, and remain touched during the entire measuring process.

1.1.3 DORIS Manipulator

To overcome this challenge, one proposed solution is to attach to DORIS wagon a lightweight robotic manipulator (Galassi et al. 2014). The utilization of manipulators attached at mobile vehicles to complement their activities is an important research subject (From et al. 2014), once the system must deal with conflicts between different mechanisms and objectives. For example, Ferreira & Romano (2007) depicts a semi-passive arm concept to support AUVs' activities. On the other hand, Carneiro et al. (2006) presents the concept of an underwater asset named LUMA for inspection tasks and exploration of sea life, in which an embedded manipulator is being currently investigated to extend the robot range for collecting geological and biological marine samples.

In DORIS case, the manipulator is intended to extend the robot workspace range, interact with nearby surfaces, move a lightweight camera through different poses, reduce the stiffness of the touching act, and increase the compliance of the robot structure, which dampens the transmission of vibration towards the robot and, hence, reduces the hazard of mechanical breaking. Thereby, this manipulator should accumulate the main tasks of: (i) move small camera; (ii) pose the vibration sensor correctly over platform nearby equipment and stand still until the process is completed; (iii) interact with nearby *touchscreens*. The small camera attached to the robot end effector may also aid the remote operator to movement the robot arm as desired, which allow the robot both manual and autonomous operation. Finally, it is possible to identify the following aspects concerning this manipulator:

(i) Small camera and vibration sensor (with long enough probe tip) attached to its end-effector;

(ii) Lightweight structure, since DORIS wagons has weight constraints;

(iii) Small payload, since it will only carry a lightweight camera and few small sensors;

(iv) Long range and wide workspace to allow the reaching of all desired nearby equipment;

(v) Interaction control, which may require a force sensor to allow the control of the contact force and compensate accumulative position and orientation errors, hence avoiding collisions between the structures and mechanical damages;

(vi) At least 5 DoFs to make possible the positioning on the 3D space and the control of pitch and yaw (the roll is not needed, since the orientation around the z-axis of the robot end effector is arbitrary);

(vii) Obstacle avoidance;

(viii) Ability to fully retract while idle, hence reducing the risk of collision with nearby platform structures during the robot normal movement along the rail;

(ix) Certification to operate at an offshore facility environment.

1.2 Motivation

Among the available commercial models, we can highlight KUKA lightweight models, as well as some solutions from ABB, Adept, FANUC, Kinova Jaco and Mico, Rethink and COMAU. None of the available options proved to be commercially attractive, since they are not enough lightweight, are not able to fully retract to a sufficient compact position, are not reconfigurable, and their payload are excessively greater than the minimum needed (which also raises the manipulator cost). Particularly, the need for limited space occupation at retracted position is critical for the DORIS working environment, but the already available models do not satisfy this requirement, especially those who most resembles an anthropomorphic arm.

On the other hand, there are commercially available actuators models which bring together unique features which makes them especially attractive for making a custom arm. For example, the German company Harmonic Drive A.G. offers a very compact servomotor model with high reduction gear ratio (up to 100 : 1) that assembles in one piece: a motor capable of providing enough power/torque for our application, embedded incremental encoder and hall sensor, a hollow shaft (through where we can guide power and signal cables, hence realizing a more compact structure), and zero backslash. In addition, there are available on the market compact and easy-to-use controlling drivers, such as those from the Swiss company Maxon Motor.

This motivates the custom-design and manufacture of an ad-hoc manipulator for the DORIS robotic system, so that all the commercial barriers described above can be circumvented. With a customized manipulator, we can: (i) select the links and joints materials to be lightweight as desired; (ii) select the joint motor models according to the robot requirements; (iii) define optimized link lengths according to the requirements of the tasks; (iv) construct a modular kinematic structure that can be reconfigured as needed; (v) improve the manipulator design in the future project developments.

A primary 5-DoF structure have already been proposed for DORIS manipulator, whose links would be constructed of carbon fiber, the joints 3D-printed in titanium, and the kinematic structure features four revolute joints and a prismatic movement along the rail. In addition, the manipulator would feature a small webcam, a piezoelectric vibration sensor, a long probe tip and a load-cell based force sensor. The lightweight structure, as well as the employed servomotors that offers high reduction gear ratio and permit direct joint speed control, should allow a modeling approach that neglects the structure's dynamics. To perform the required tasks, this manipulator should operate in two different modes: (i) manual, which should allow the operator to pose the arm tip wherever desired and calibrate the robot operation; (ii) automatic, which should permit vibration inspection and interaction with *touchscreens* in autonomous way during scheduled tasks. The movement of the small camera is simple since it is teleoperated, however, the collected video should undergo linear transformations to deliver the aligned image to the operator (e.g., in case of the camera being upside down).

In addition, one of the major issues found in robot control is how to deal with joint configurations nearby singularities, in which the robot end-effector is close to the boundaries of its reachable workspace or when two or more axes of motion of robot joints are aligned. In these cases, the mobility of the structure is reduced (Siciliano et al. 2009, Donelan 2010), infinite solutions to inverse kinematics problem may exist and small velocities in the Cartesian space may cause large velocities in the joint space. In those cases, the numerical computation of the inverse Jacobian matrix becomes extremely difficult (and impossible exactly over the singularity), which is why the classic control strategies usually takes the robot away these points. Some recent proposed strategies aim to overcome this problem and can be exploited in the control design for the DORIS manipulator, such as the Damped Least Squares (DLS) method (Nakamura & Hanafusa 1986), Feedback Inverse Kinematics (FIK) method (Pechev 2008), and Filtered Inverse (FI) approach (Vargas 2013).

This ultimate technique proposes the calculation of inverse scalar or matrices dynamically at a rate defined by the matrix conditioning. A consequence of this method is that the manipulator can pass over singularities (except workspace limits) without losing the conditioning of the inverse Jacobian. Further features and highlights of this technique include the easiness of tuning (only one adapting gain) and the possibility of minimizing a cost function related a problem of augmented dimension and additional constraint. Successful results of this method application for joint limits and obstacle avoidance motivate the use of FI (Vargas 2013) for the control of DORIS manipulator. Since it is a mobile robot with limited energy capacity, it is worth to improve its control computing and minimize the use of energy. Efficient solutions with reduced energy consumption have recently been covered by Vergnano et al. (2012), Paes et al. (2014) and Ackerman (2015), which shows off that robots with smooth paths are up to 40% more efficient. In this context, energy cost functions can be investigated and applied to the DORIS manipulator.

Finally, to deliver the manipulator fully operational for DORIS robotic system, the following steps should be accomplished:

(i) Design primary structure (materials, actuators, sensors, kinematic model);

(ii) Simulation of operational modes and control in computing environment;

(iii) Feedback of the simulation results, with the possibility of altering the original structure (with new simulations and conclusions in changing case)

(iv) Mounting of mechanical parts: joints, links, actuators and sensors;

(v) Integration to electronics system (which includes motor/driver tests, supply tests, cable and connector construction and system assembly) and test in protected laboratory environment (with new mounting, tests and conclusions in changing case).

(vi) Integration to DORIS and field tests (i.e., operation at platform environment);

(vii) Feedback of field tests, with the possibility of new changes, simulations and final field tests.

1.3 Objectives

In this work, the objective is to develop the modeling, control and electromechanical design for a lightweight modular robot manipulator in order to perform interaction and inspection tasks in offshore platforms. In particular, this work aims to:

- Investigate, implement and test control strategies to make possible the performance of the following tasks:
  - Pose the vibration sensor perpendicularly to the surface of a nearby machine and remain in full contact with controlled force during the measuring process.
  - Interact with nearby *touchscreens*, which movement should be force-controlled and the fingertip should remain perpendicularly posed with respect to the surface, regardless of the followed path over the surface.
- Define operational modes/procedures for:

    - Movement a lightweight camera through the manipulator workspace to allow the visualization of the environment from different points of views.
    - Fully retract when idle to avoid collision with nearby platform structures.

- Test the filtered inverse technique and the augmented Jacobian (Vargas 2013) to:
    - Enhance the computational performance near singularities.
    - Investigate a cost function to minimize energy consumption.

- Propose modifications of the original manipulator structure aiming to solve possible design flaws (such as singular configurations).

The overall goal is to demonstrate the advantages of designing and developing an own manipulator for very specific interaction tasks instead of purchasing a commercial model, which is not economically attractive in most cases. For simplification reasons, a name was given for this manipulator: TETIS[4].

1.4 Methodology

Firstly, this work introduces a review of basic important concepts regarding the control of serial manipulators, such as kinematic modeling and control, etc. Then, the electromechanical structure of the DORIS manipulator with 5-DoF will be detailed, including its required functionalities, studies regarding the selection of the better sensors and materials, and its kinematic model, which will aid us to identify possible design flaws, and, hence, propose modifications for it. On view of this design, there will be proposed a flowchart to organize the operation modes (teleoperation and automatic) and task phases of the manipulator, which will make possible the investigation of suitable control techniques for each phase. Both *touchscreen* and vibration inspection tasks are organized in five phases:

(i) *Trajectory tracking* towards a goal contact surface, stopping near and perpendicularly oriented with respect to it. This will be achieved via pose control scheme for the robot end effector based on the kinematic control approach

[4]This name refers to two aspects concerning the robot: according to the Greek mythology, *Doris* is daughter of the Titaness *Tethys* and mother of the Nereid *Tethis*. Also, the chemical element *Titanium*, which the manipulator joints are made of, was named for the Titans of the Greek mythology.

(ii) *Approach* towards the surface with the control of the contact force. This will be accomplished with interaction control scheme for the robot end effector based on the position-force hybrid control approach (Leite 2005, 2011). For this step, a force sensor needs to be attached to the tip of the robot end effector.

(iii) *Path tracking over the contact surface*, remaining perpendicular to it[5] until the task is completed. If the surface is known, the end-effector remains posed perpendicularly to the surface if the x, y components of the contact force are null (Leite 2005). The dynamic of the contact forces during the interaction can be modeled by several approaches, such as presented by Murray et al. (1994). In this work, we utilize the *Coulomb friction model.*

(iv) *Surface geometry re-estimation*, case needed to circumvent the interaction problem on contact surfaces with unknown geometry. Originally, we consider a structured environment, being the position and geometry of all contact surfaces perfectly known and invariable. Yet, we consider later the situation in which, after successive robot uses, the error accumulation makes the surface pose to be not accurately known. We can re-estimate the surface normal vector by using the encoders and force sensor measurements and introducing small displacements of the end-effector over the surface (Leite 2005).

(v) *Retraction*, which works similarly to the first phase, except that the goal path is exactly the the opposite and points towards the original retracted pose.

The Robotics Toolbox (Corke 2011) in MATLAB/Simulink environment is used to simulate these steps above and illustrate the manipulator operation.

Still in MATLAB/Simulink environment, we test the Filtered Inverse method (Vargas 2013) to control the trajectory tracking phase, compare its application with the use of classic Jacobian pseudo-inverse in the context of dealing with the proximity of singularities, and investigate a solution to minimize the energy consumption.

1.5 Contribution

The main contributions of this work are:

[5]Being *perpendicular to a surface* at a specific point over it means to be aligned with the normal vector of an imaginary flat plane tangent to the original surface at this point.

(i) A prospective study on manipulator mechanics, aiming to demonstrate the feasibility of the design of lightweight, modular and compact manipulators for mobile robots to develop simple interaction tasks.

(ii) Combination of different control strategies: pose end effector control, hybrid position-force control, estimation of the normal vector of contact surfaces, singularity avoidance using the Filtered Inverse method.

(iii) Utilization of a particular property of the Filtered Inverse method (augmented Jacobian) to investigate the optimization of cost functions, such as mechanical joint limits avoidance and minimum energy consumption, which is a concern for autonomous mobile robots.

1.6 Organization of this work

This work is organized as follows:

- **Chapter 2** - Reviews the essentials of kinematic modeling and control of serial manipulators, such as: forward kinematics (including Denavit-Hartenberg parameters), inverse kinematics, differential kinematics, kinematic control approach, end-effector pose control, interaction control, hybrid control approach, tasks' execution in constrained environments, normal vector estimation for unknown geometry surfaces, and singularity avoidance techniques.
- **Chapter 3** - Details DORIS manipulator electromechanical project (including mechanical structure, description of functionalities, embedded actuators and sensors, study of commercially available parts, and detailing of the integration with DORIS electronics system), kinematic modeling, singularity analysis and suggestions for modifications of the original design.
- **Chapter 4** - Presents how the manipulator will be operated and controlled, which includes the detailing of the manipulator operating modes (teleoperated or autonomous) and the application of control strategies (reviewed in Chapter 2) to the manipulator tasks.
- **Chapter 5** - Presents several simulations in MATLAB/Simulink environment that illustrate the the manipulator operation as described in Chapter 4. The final simulations include the application of the Filtered Inverse and augmented Jacobian techniques.

- **Chapter 6** - Summarizes the final considerations concerning the results obtained from the implementations in Chapter 5. Finally, it discusses future works or researches that shall take advantage of the subjects approached in this thesis.

Chapter 2

Fundamentals of Manipulator Kinematic Modeling and Control

In this chapter, we present the fundamentals and basic concepts of kinematic modeling and control design for serial robot manipulators (Siciliano et al. 2009, Murray et al. 1994, Siciliano & Khatib 2008). The chapter presents:

- The essential concepts for analysis of serial kinematic manipulators.
- Basic fundamentals of the control strategies for kinematic manipulators.
- Algorithms to deal with kinematic singularities.

As shown in Siciliano et al. (2009) for a further reading, robot manipulators can be classified in many categories, according to their structure design, employed material, actuators and tasks. In this work, the robot manipulator to be considered consists of rigid links in an open kinematic chain composed with prismatic and revolute joints. A review of rigid body motion, which is essential for the study of serial rigid-body manipulators, is available in Murray et al. (1994).

2.1 Forward kinematics

The goal of the *forward kinematics* is to determine the position and orientation/attitude of a manipulator end-effector - with respect to an inertial coordinate system - in function of its *joint variables.* As noted above, it will be only considered serial open-chain rigid-link manipulators with only prismatic or revolute joints (see Figure 2.1).

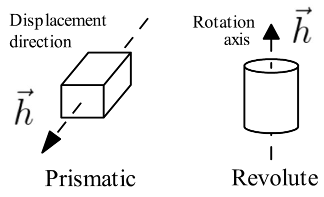


Figure 2.1: Prismatic and revolute joints representations.

2.1.1 Initial definitions

Let us consider a serial kinematic chain of rigid links connected together by joints (Figure 2.2a). The first link of this chain is fixed at the *base*, and an *end-effector* is connected to the last link. Let us arbitrarily assign the frame $\bar{E}_b$ to any point solidary to the base, and, likewise, attach the frame $\bar{E}_e$ to any point solidary to the end-effector. The frame $\bar{E}_b$ is named the *base or world frame*, whose role is to be a reference to the movement of all the manipulator parts. The frame $\bar{E}_e$ is named the *end-effector or tool frame*. Generally, $\bar{E}_e$ origin is placed at the most extreme point of the end-effector, and its components are defined as $\bar{E}_e = \begin{bmatrix}\vec{n} & \vec{s} & \vec{a}\end{bmatrix}$. As mentioned in Siciliano et al. (2009), these components are defined as the end-effector *normal*, *slide* and *approach* vectors, respectively (Figure 2.2b).

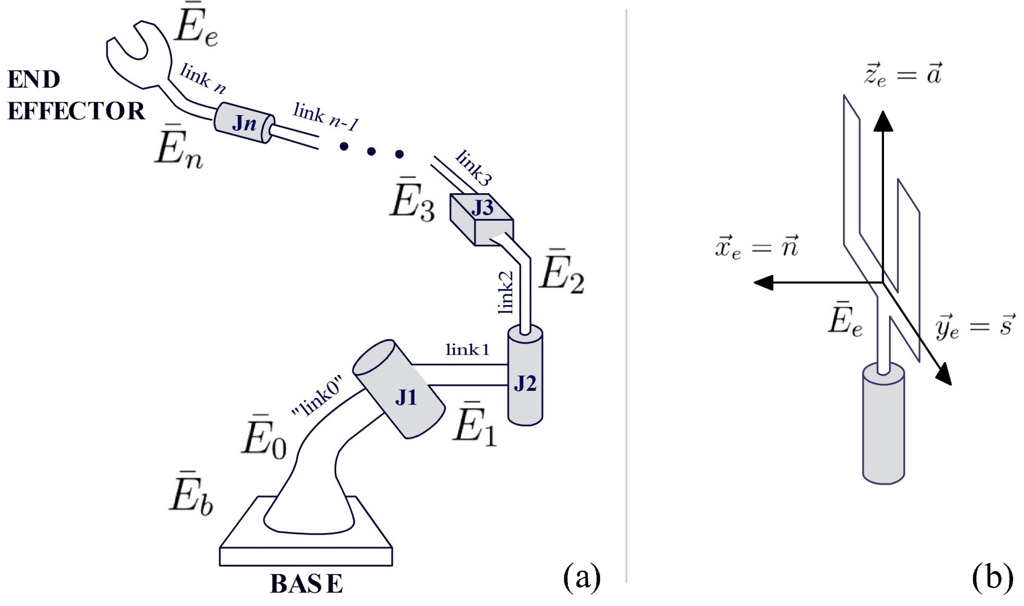


Figure 2.2: Serial kinematic chain: (a) representation; (b) end-effector frame.

For a n-link/n-joint chain (Figure 2.2a), the link-1 is defined as the first link not solidary to the base, and the link-n is defined as the link solidary to the end-effector. The frame $\bar{E}_i$ is solidary to the link i. For modeling issues, it is a good practice to name the imaginary link that connects the base to joint-1 as the “link-0”. The frame $\bar{E}_0$ is solidary to this link, and hence, to the base. Finally, it is worth noticing that every joint-i connects the link $i-1$ to the link i.

2.1.2 End-effector pose with respect to the base

The most practical way to represent the end-effector pose with respect to the base is by means of the homogeneous transformations (Siciliano et al. 2009). Considering that each joint i has one DoF associated with one joint variable q_i, we can assign a homogeneous transformation $T_{i-1,i}(q_i)$ to describe the pose of link i with respect to link $i-1$ in function of q_i. As a joint variable, q can be a displacement (d) for prismatic joints or an angle (θ) for revolute joints. For the complete chain, we want to find the composition:

$$T_{0,n}(q) = T_{01}(q_1)T_{12}(q_2)\cdots T_{n-1,n}(q_n), \tag{2.1}$$

where $q = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix} \in \mathbb{R}^n$ is the *joint variable vector* belonging to the *joint space* $\mathcal{J}$.

Since $\bar{E}_0$ and $\bar{E}_n$ are solidary but not equal to $\bar{E}_b$ and $\bar{E}_e$ respectively, we must define the transformations T_{b0} and T_{ne}, which do not depend on q. Finally, the homogeneous transformation that relates the end-effector with the base poses is given by:

$$T_{be}(q) = T_{b0}T_{0n}(q)T_{ne} = \begin{bmatrix} R_{be}(q) & (\vec{p}_{be})_b(q) \\ 0 & 1 \end{bmatrix}, \tag{2.2}$$

where $(\vec{p}_{be})_b$ is the position vector, which denote the position of the origin of the frame $\bar{E}_e$ with respect to the frame $\bar{E}_b$, and $R_{be} = \begin{bmatrix} (\vec{n}_e)_b & (\vec{s}_e)_b & (\vec{a}_e)_b \end{bmatrix} \in SO(3)$ is the rotation matrix, which denotes the orientation of the $\bar{E}_e$ with respect to the frame $\bar{E}_b$.

Notice that rotation matrices provide a redundant description of the frame orientation since they are characterized by nine elements which are not independent but related by six constraints due to the orthogonality conditions. A minimal (3-DoF) representation of orientation can be obtained by using rotation matrices expressed in terms of sets of Euler angles (e.g., Roll-Pitch-Yaw angles) or *angle-axis* formulation (Siciliano et al. 2009), but these present non-uniqueness for representing some configurations.

The *quaternion* formulation was proposed as a generalization of complex numbers (Hamilton 1844, 1853). In robotics, the *unit quaternions* are used as an elegant and *singularity-free* representation for a body orientation (Chou 1992, Siciliano et al. 2009). Besides being good for computational implementation, it is only defined by 4 parameters and only 1 constraint. A quaternion for representing a rotation of θ around the vector ω is defined as $Q_\theta = (\eta_\theta, \epsilon_\theta)$, where:

$$\eta_\theta = cos(\theta/2), \in \mathbb{R} \quad \epsilon_\theta = sin(\theta/2)\omega, \in \mathbb{R}^{3\times 1}, \tag{2.3}$$

and $\|Q_\theta\| = 1$ is the sole constraint.

2.1.3 Denavit-Hartenberg parameters

There are several approaches to find the matrix T_{be}. One of the most established method in the literature is the *Denavit-Hartenberg (DH) convention.* This approach is a systematic procedure that defines, for each link $i = 1, 2, \cdots, n$, the position and orientation of the frame $\bar{E}_i$ in terms of four different parameters that define an unique homogeneous transformation with respect to the previous link. In the literature, there are two ways to apply this method: (i) *DH Standard*, utilized by Siciliano et al. (2009) and Spong et al. (2006); (ii) *DH Modified*, which is a variation of DH Standard and have been utilized by NASA and described by Craig (2005).

2.2 Operational space

The *operational space* $x \in \mathbb{R}^m$ defines the minimum DoFs to describe a given task. For example, the positioning of the end-effector within the 3D-space (without caring about the orientation) needs only $m = 3$ DoF's. When the task needs to define orientation, it must be described by a minimal representation, such as Euler or Roll-Pitch-Yaw angles, which requires up to 3-DoF. Hence, we can represent the configuration of the end-effector as:

$$x = \begin{bmatrix} p \\ o \end{bmatrix} \in \mathbb{R}^m, m \leq 6, \tag{2.4}$$

where $p \in \mathbb{R}^{m_p}, m_p \leq 3$ is the position, and $o \in \mathbb{R}^{m_o}, m_o \leq 3$ is a representation of the orientation. Thus, we can define the robot *forward kinematics* as a non-linear function k:

$$x = k(q). \tag{2.5}$$

It is worth noticing that the maximum number of DoF needed by a robot is 6. Anthropomorphic manipulators are capable of performing 6-DoF tasks. They have a 3-DoF arm and a decoupled 3-DoF wrist. Planar manipulators only need 2-DoF for describing their position.

2.2.1 Kinematic redundancy

From a more comprehensive point-of-view, a manipulator is called to be *kinematically redundant* when the number of its joints (n), which is equal to its degrees-of-freedom or mobility in most cases, is greater than the needed to describe a given task. In a more strict point-of-view, redundancy can be classified into two categories:

(i) *Intrinsic redundancy*, when the dimension of the operation space is smaller than the dimension of the joint space ($m < n$).

(ii) *Functional redundancy*, when the number r of variables needed to describe a specific task is smaller then the dimension of the joint space ($r < n$)

Generally, it is common to assume that $m = r$, but this is not valid for all cases. For example, let us consider a three-link planar robot, in which $n = 3$ and $m = 3$, i.e., we can achieve, at least, positions in x and y-axes and orientation around z-axis. This robot is not redundant, but it can become functionally redundant if a given task requires only control of x and y positions, i.e., $r = 2$.

2.3 Workspace

The manipulator *workspace* is defined as the region described by the end-effector frame origin when all the robot joints execute all the combined possible motions. In other words, it is the set of all points in $\mathbb{R}^3$ that the end-effector can reach. Generally, the workspace of a manipulator is an index of its performance with respect to its operational space, since through it we can analyze *how far* the arm can extend, and to how many orientations it can be driven when it is nearly fully outstretched.

It is common to split the workspace into to definitions. The *reachable workspace* W_R is the region of all points that the end-effector frame can reach with, at least, one orientation. Yet, the *dexterous workspace* W_D is the region where the end-effector frame can reach with all possible orientations. We can formally define the reachable workspace for a n-ink manipulator as follows:

$$W_R = \{(\vec{p}_{be})_b : q \in \Theta\}, \tag{2.6}$$

where $\Theta = \left\{ q = \begin{bmatrix} q_1 & \cdots & q_i & \cdots & q_n \end{bmatrix}^T : \forall i \in [1, n] \mid q_{im} \leq q_i \leq q_{iM} \right\}$ contains, for all i, all the possible values of q_i between its minimum q_{im} and maximum q_{iM} mechanical limits. Analogously for the dexterous space:

$$W_D = \left\{ (\vec{p}_{be})_b : (\vec{p}_{be})_b \in W_R \land \forall R_{be} \in SO(3), \exists q \mid T_{be}(q) = \begin{bmatrix} R_{be} & (\vec{p}_{be})_b \\ 0 & 1 \end{bmatrix} \right\}, \tag{2.7}$$

where $T_{be}(q)$ is the homogeneous transform of the forward kinematics. It is worth noticing that the dexterous workspace is a subset of the reachable workspace ($W_D \subset W_R$). As an analogy to the human arm, if it is fully extended, the pointer finger tip is reaching a point of the arm reachable workspace. However, it does not belong to the dexterous space, since the only motion that can be done at this point without moving the finger tip from its position is a hand *roll*. In Figure 2.3, we can check the sketches of the reachable workspace of established manipulators.

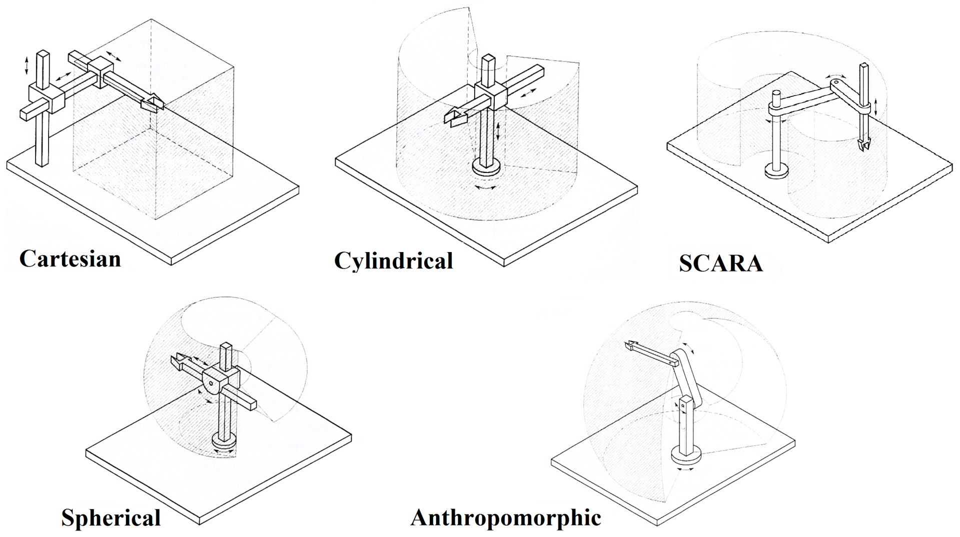


Figure 2.3: Workspace sketches of established manipulators.

2.4 Inverse kinematics

The problem of the *inverse kinematics* relies on the search for finding the inverse function k^{-1} that defines the joint space q with respect to the operational space x, i.e.:

$$q = k^{-1}(x). \tag{2.8}$$

However, unlike the forward kinematics, this is a challenging task, since: (i) it is a non-linear function, i.e., hard to represent in a closed form; (ii) may generate multiple or infinite solutions; (iii) may generate infinite solutions, specially in redundant structures; (iv) may not admit any solution. If we face this challenge departing from the homogeneous transformation $T_{be}(q), q \in \mathbb{R}^n$, we would have 12 equations (3 for position and 9 for orientations) with n variables to solve. Thus, there are several strategies to approach the problem of inverse kinematics, besides trying to solve it using algebraic and geometric intuition.

2.4.1 Kinematic decoupling

This elegant method (Siciliano et al. 2009) can be applied to manipulators in which the end-effector orientation can be decoupled from its position (generally, industrial manipulators with spherical wrist). This occurs if the axes of the three last joints cross themselves at the same point (*decoupling point*). In this case, their movements never change the position of this point.

Once this point is recognized, it is worth solving the inverse kinematic problem until this point (which becomes much more simple since it has 3 less unknown variables to find), and then solve the problem for the last three wrist joints separately.

2.4.2 Paden-Kahan Subproblems

This geometric algorithm, originally presented by Paden (1985) and first built by Kahan (1983), is a systematic procedure that can be applied to solve the inverse kinematics by calculating the solution of a set of subproblems. The problem consists of seeking four possible situations by analyzing the manipulator, and reducing them into appropriate subproblems whose solution are known. Unfortunately, there may exist robots which cannot be solved by using this canonical problems (Murray et al. 1994).

2.4.3 Iterative approach

Given a task in the operational space $x = \begin{bmatrix} p_e & o_e \end{bmatrix}^T, x \in \mathbb{R}^m$, a simple way to determine its inverse kinematics $q = k^{-1}(p_e, o_e)$ is by using an iterative algorithm (Siciliano et al. 2009), as illustrated in the block diagram in Figure 2.4, where $J(q) = \nabla_q k(q)$ is called the manipulator *analytical Jacobian* (as further explained in 2.5.2) . The *non-linear error* $\epsilon = x - k(q)$ that represents the difference between the robot real pose and the computed pose converges asymptotically to zero if $\lambda > 0$ is a positive proportional gain, as demonstrated by the *Lyapunov Method* (Slotine et al. 1991).

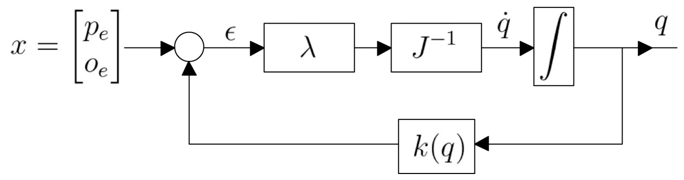


Figure 2.4: Inverse kinematics via iterative algorithm.

Notice that the analytical Jacobian $J(q)$ may not be square neither invertible in some manipulator configurations. It can be replaced by its transposed form $J^T(q)$ (conjugate gradient) or pseudo-inverse $J^{\dagger}(q)$, whose application will be explained further in Section 2.6.2.1.

2.5 Differential kinematics

As seen in Section 2.1, the forward kinematics provides a non-linear problem to solve. On the other hand, as we observe the iterative approach for finding the inverse kinematics (Section 2.4.3), the *Jacobian matrix* turns possible the problem linearizing. It provides the *differential kinematics*, i.e., the linear mapping between the joint space velocities $\dot{q}$ and the linear and angular velocities of an arbitrary point over the manipulator structure, such as its end-effector.

The Jacobian is one of the most important variables for the analysis of a robotic manipulator, assisting in: (i) kinematic control; (ii) identification of singularities; (iii) smooth path planning; (iv) calculation of robot motion dynamics; (v) computation of the transformation of applied forces/torques at the end-effector to joints.

2.5.1 Geometric Jacobian

The Geometric Jacobian maps the joint velocities to the end-effector linear and angular velocities, i.e.:

$$\begin{bmatrix} \dot{p} \\ \omega \end{bmatrix} = J_G(q)\dot{q}, \tag{2.9}$$

where $J_G(q) = \begin{bmatrix} J_p(q) \\ J_o(q) \end{bmatrix} \in \mathbb{R}^{m \times n}$, and $J_p(q)$ is the *position Jacobian* and $J_o(q)$ is the *orientation geometric Jacobian.*

Considering a n-joint serial manipulator, its geometric Jacobian can be calculated in an abstract manner that is independent of coordinate systems and which DH approach is being used, as follows:

$$\begin{bmatrix} \vec{v}_n \\ \vec{\omega}_n \end{bmatrix} = J_n(q)\dot{q} = \begin{bmatrix} J_{p1} & J_{p2} & \cdots & J_{pn} \\ J_{o1} & J_{o2} & \cdots & J_{on} \end{bmatrix} \dot{q}, \tag{2.10}$$

where $\vec{v}_n = \dot{\vec{p}}_n$ is the linear velocity of joint n and ω_n is the angular velocity of joint n. For the i^{th} joint:

$$\begin{bmatrix} J_{pi} \\ J_{oi} \end{bmatrix} = \begin{bmatrix} \vec{h}_i \\ 0 \end{bmatrix} \text{, for a prismatic joint,} \tag{2.11}$$

$$\begin{bmatrix} J_{pi} \\ J_{oi} \end{bmatrix} = \begin{bmatrix} \vec{h}_i \times \vec{p}_{in} \\ \vec{h}_i \end{bmatrix} \text{, for a revolute joint,} \tag{2.12}$$

where $\vec{h}_i$ is the unitary axis of joint i ($\vec{z}_{i-1}$ for Denavit-Hartenberg Standard (DHS) modelings and $\vec{z}_i$ for Denavit-Hartenberg Modified (DHM)), and $\vec{p}_{in}$ is the vector that links $\bar{E}_i$ to $\bar{E}_n$.

Notice that J_{pi} and J_{oi} represent the contributions of joint i to the final linear and angular velocity, respectively. Also notice that this Jacobian can be represented with respect to a coordinate system. For example, if we want to represent it with respect to the base frame, we would have, for a i^{th} revolute joint (using DHM):

$$(J_i)_b = \begin{bmatrix} \widehat{(\vec{z}_i)}_b \left[(\vec{p}_{bn})_b - (\vec{p}_{bi})_b\right] \\ (\vec{z}_i)_b \end{bmatrix}. \tag{2.13}$$

If we want this Jacobian to be represented in another coordinate system $\bar{E}_f$ with $R_{fb} = \bar{E}_f^* \bar{E}_b$, then the following equation is valid:

$$(J_n)_f = \begin{bmatrix} R_{fb} & 0 \\ 0 & R_{fb} \end{bmatrix} (J_n)_b. \tag{2.14}$$

Notice that this Jacobian provides a mapping to the velocities n^{th} link frame. However, it can be obtained in another solidary frame to this link. Generally, it is worth mapping it to the end-effector. This can be done using the following equation:

$$\begin{bmatrix} \vec{v}_e \\ \vec{\omega}_e \end{bmatrix} = \begin{bmatrix} \mathcal{I} & -\vec{p}_{ne}\times \\ 0 & \mathcal{I} \end{bmatrix} \begin{bmatrix} \vec{v}_n \\ \vec{\omega}_n \end{bmatrix}, \tag{2.15}$$

or, with respect to a given frame (e.g., the base),

$$\begin{bmatrix} (\vec{v}_e)_b \\ (\vec{\omega}_e)_b \end{bmatrix} = \begin{bmatrix} I & -\widehat{(\vec{p}_{ne})}_b \\ 0 & I \end{bmatrix} \begin{bmatrix} (\vec{v}_n)_b \\ (\vec{\omega}_n)_b \end{bmatrix}, \tag{2.16}$$

where $\mathcal{I}$ is an *identity operator*. If the Jacobian maps to an intermediary joint $l < n$, the next joints should not be considered in its computation, since their contribution to the motion of joint l motion, i.e.:

$$J_l(q) = \begin{bmatrix} J_{p1} & J_{p2} & \cdots & J_{pl} & 0 & \cdots & 0 \\ J_{o1} & J_{o2} & \cdots & J_{ol} & 0 & \cdots & 0 \end{bmatrix}. \tag{2.17}$$

2.5.2 Analytical Jacobian

Another way to find the differential kinematics is using a minimal representation of the orientation ϕ (e.g., using Euler angles and RPY) and deriving it with respect to time. Therefore, departing from $x = \begin{bmatrix} p^T & \phi^T \end{bmatrix}^T = k(q) \in \mathbb{R}^m$, we have:

$$\dot{x} = \begin{bmatrix} \dot{p} \\ \dot{\phi} \end{bmatrix} \underbrace{\frac{\partial k(q)}{\partial q}}_{J_A(q)} \dot{q}, \tag{2.18}$$

where $J_A(q) \in \mathbb{R}^m$ is the *analytical Jacobian*.

2.5.3 Representation Jacobian

Both geometric and analytical Jacobian maps the joint velocity space to the same linear velocity, since $v = \dot{p}$. On the other hand, the first maps to the angular velocity, whereas the second maps to differential quantities in the operational space, i.e., $\omega \neq \dot{\phi}$. The vector ω has a clear physical meaning, and its components are called *nutation*, *spin* and *precession*, respectively. Yet, $\dot{\phi}$ in not intuitive, but its integral with respect to time yields ϕ, while the integral of ω has no physical interpretation. The relationship between these quantities is given by:

$$\dot{\phi} = J_R(\phi)\omega, \tag{2.19}$$

where $J_R(\phi)$ is called the *representation Jacobian*. Therefore, we can also find a relationship between the geometric and analytical Jacobian. Gathering the relationships in matrix form:

$$\dot{x} = \begin{bmatrix} \dot{p} \\ \dot{\phi} \end{bmatrix} = \underbrace{\begin{bmatrix} I & 0 \\ 0 & J_R(\phi) \end{bmatrix}}_{T_R(x)} \begin{bmatrix} v \\ \omega \end{bmatrix} \rightarrow \dot{x} = \underbrace{T_R(x) J_G(q)}_{J_A(q)} \dot{q}, \tag{2.20}$$

we have:

$$J_A(q) = T_R(x)J_G(q). \tag{2.21}$$

2.5.4 Singularities

Singularities are undesired situations for the control of robotic manipulators. They occur in such joint configurations that lead the Jacobian to lose rank.

The rank of a matrix as the end-effector Jacobian J_e is defined by the number of its linearly independent columns, which is also the dimension of its *column vector space* $\mathcal{R}(J)$. In robotics, $rank(J_e)$ is the number of independent DoFs in $SE(3)$ through which the end-effector can move. If the Jacobian is defined for another point (e.g., the 3^{rd} link), then, $rank(J_3)$ denotes the number of independent DoFs through which the 3^{rd} link can move. Also, a singularity can occur if the dimension of the Jacobian *null space* increases, i.e., if the movement of some joints does not cause any effect to the end-effector motion.

Finding the singularities of a manipulator is of great interest, since:

- The represent robot configurations in which its mobility is reduced, so that we cannot impose arbitrary motions to the end-effector;
- In the singularity neighborhood, small velocities at the end-effector cause large joint velocities. The opposite situation may also occur;
- Infinite solutions to the inverse kinematics may arise exactly over a singularity point.

Also, the singularities can be classified as:

- Boundary singularities: when the arm is fully retracted or outstretched at its mechanical limit, so that the end-effector is placed at the robot reachable workspace boundary. This is not exactly an issue, since it is not common to drive the robot to the limit of its workspace.
- Internal singularities (2 or more axes are aligned): unlike boundaries, this singularity type can occurs anywhere inside the robot workspace, which is why this situation is concerned as a critical problem to path planners and robot controllers.

The computation of singularities using the Jacobian determinant may be a tedious task with no easy solution for complex structures. However, in manipulators with kinematic decoupling (especially anthropomorphic arms, see section 2.4.1), this task can be split into two separate problems:

(i) *Arm singularities*, which result from the motion of the first three of more joints.

(ii) *Wrist singularities*, which result from the motion of the wrist joints.

In structures with kinematic decoupling (for simplicity, let us consider $n = 6$), the geometric Jacobian is given by:

$$J = \begin{bmatrix} J_{11} & 0 \\ J_{21} & J_{22} \end{bmatrix}, \tag{2.22}$$

since q_4, q_5, q_6 do not contribute to the linear motion. Hence, the Jacobian determinant is simply given by:

$$det(J) = det(J_{11})det(J_{22}), \tag{2.23}$$

which makes the task of finding singularities much more simple, since we can analyze separately the problems of $J_{11} = 0$ (arm) and $J_{22} = 0$ (wrist).

2.5.5 Manipulability

Manipulability is an index that measures the *distance* that a given joint configuration is from a Jacobian singularity. When a manipulator is near singular configurations, the end-effector mobility is reduced, which is a critical problem for most applications. Generally, it is worth to plan end-effector paths that maximizes its manipulability.

In non-redundant manipulators - in which the Jacobian is a square matrix, since m (task #DoF) $= n$ (joint space #DoF) - the manipulability is quantized by the matrix eigenvalues. The closer to zero that any eigenvalue is, the nearer a singular configuration the robot finds itself. However, in redundant manipulators, the Jacobian is non-square. *Single Value Decomposition* (SVD) is a elegant toll to circumvent this problem, since it generalizes the concept of eigenvalues and eigenvectors. Hence, being $J \in \mathbb{R}^{m \times n}, n > m$, we have the following decomposition:

$$J = \sum_{i=1}^{m} \sigma_i u_i v_i^T, \tag{2.24}$$

where v_i and u_i are respectively the input and output singular vectors, and σ_i are J singular values and sorted like $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$, where $r = rank(J)$. This means that, when J multiplies v_i, a vector along u_i direction is multiplied by σ_i, i.e.:

$$Jv_i = \sigma_i u_i. \tag{2.25}$$

Hence, it is worth noticing that, for $\sigma_i >> 0$, large movements along u_i can be generated with small input in $v_i = q_i$. On the other hand,, when $\sigma_i \approx 0$, any movement along u_i requires large inputs $v_i = q_i$, which describes a nearly singularity situation (2.5.4). If $\sigma_i = 0$, a mathematical solution for a movement along u_i is unfeasible.

One established way to describe the robot manipulability involving its joint and linear/angular velocities is by using this decomposition. To understand, firstly let us define:

$$\mathcal{Q} = \left\{\dot{q},\ \dot{q}^T\dot{q} = 1\right\}, \tag{2.26}$$

as the set of normalized velocities (geometrically deployed over an unit radius sphere) which can be described by the *joint velocity space*. Using $\dot{q} = J^{\dagger}\nu = J^T(JJ^T)^{-1}\nu$ (as described further in Section 2.6.2.1), this can be mapped to the velocities that the robot can perform in the end-effector *operational velocity space* as follows:

$$\nu^T(JJ^T)^{-1}\nu = 1, \tag{2.27}$$

which defines an ellipsoid of vectors ν in $\mathbb{R}^m$ with orientation and shaped set by its quadratic norm JJ^T. Hence, we conclude that an unit sphere of possibilities for joint velocities generates an ellipsoid of possibilities for the end-effector movement, as illustrated in Figure 2.5 for a two-link planar arm. A simpler and established way to determine the robot manipulability is using the ellipsoid volume, which is proportional to:

$$\omega(q) = \sqrt{det[J(q)J^T(q)]}. \tag{2.28}$$

Figure 2.5 also shows the manipulability measure with respect to the generated ellipsoid for different joint configurations (two-link planar arm).

2.6 Kinematic control

Kinematic control of the end-effector position, orientation or other variables - such as interaction force/torque - is a problem applied to manipulators whose dynamic effects are neglected within the closed loop system, which are also called as *kinematic structures*. This is only possible when the velocities of the robot joints $\dot{q}$ can

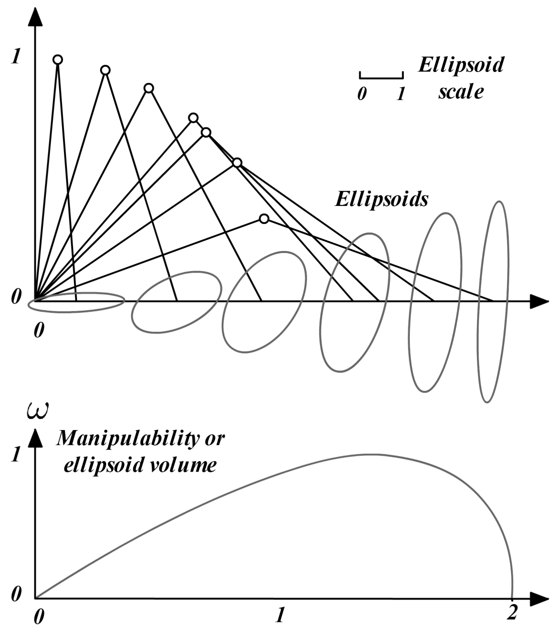


Figure 2.5: Manipulability ellipsoids and volume for two-link planar arm.

be directly controlled. Ideally, this happens when the gain of the joint local velocity control loop is infinitely high. In practice, this can be almost achieved by using high-performance drivers and very high gear transmission ratios, hence provide high torques, and, in most cases, small velocities. Dynamics effects can be also unconsidered when the actuators non-linearities (e.g., motor backslash) are negligible, the joint velocities are sufficiently small, and the structure is sufficiently lightweight (very rare).

2.6.1 Actuator control at joint velocity level

The majority of industrial robots employ actuators that have a *fine* velocity control loop with sufficient high gain K_i. Figure 2.6 represents a velocity control loop of a kinematic servo-actuator acting on a joint i, with reference set-point of u_i and torque τ_i provided to the joint shaft. Hence, we consider that, for $K_i \to \infty$, the joint velocity error $e_i = u_i - \dot{q}_i$ rapidly goes to zero ($e_i \to 0$), and, consequently, $u_i \to \dot{q}_i$. Considering that, for a n-joint manipulator, the above affirmation is valid for every $i = \{1, 2, \cdots, n\}$, we assume that all the manipulator joints $q = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix}^T$ can be directly and instantaneously controlled at speed level, i.e., $\dot{q} \approx u$, being u_i the *control signal* applied to the driver that powers the i^{th} joint. As discussed in Section 2.4.3, the direct availability of $\dot{q}$ allows the implementation of a closed loop iterative algorithm that permits the computation of the robot inverse kinematics.

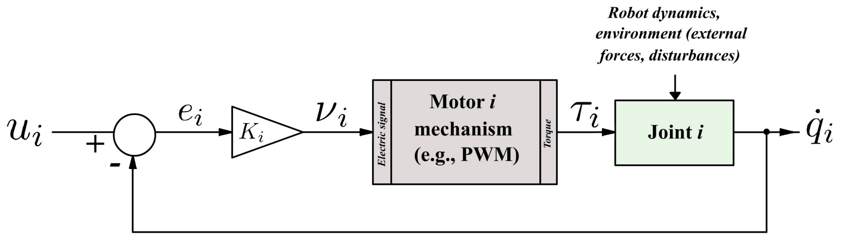

Robot dynamics, environment (external forces, disturbances)
u_i
+
-
e_i
K_i
ν_i
Electric signal
Motor *i* mechanism (e.g., PWM)
Torque
τ_i
Joint *i*
$\dot{q}_i$

Figure 2.6: Block diagram: generic speed control loop of servomotor.

This is the key point to develop the strategy for the kinematic control.

2.6.2 Control loop

Assuming the condition found above $\dot{q} = u$, we can close a loop in order to control the manipulator end-effector pose x to a desired configuration x_d in the operational space. Let us assume our operational space as $x = k(q), x \in \mathbb{R}^m, q \in \mathbb{R}^n$, where k is a function that describes the robot forward kinematics (Section 2.1). The closed loop is shown as a block diagram in Figure 2.7.

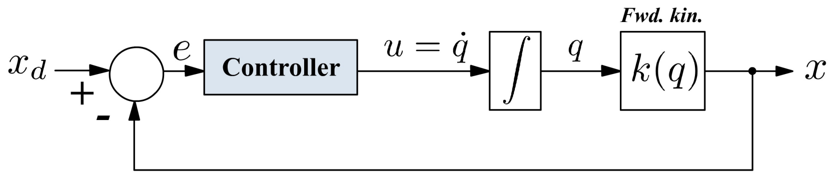


Figure 2.7: Closed kinematic control loop.

The error is defined as $e = x_d - x$. Our goal is to design a control function $u = f(e)$ to guarantee asymptotic convergence of the error signal to zero. For this, notice that the error dynamics can be written as a first order differential equation, since the open loop has only one integrator. Hence, if we define:

$$\dot{e} + Ke = 0, \tag{2.29}$$

and $K = K^T > 0$ is the pose gain matrix. Then, the system (2.29) is asymptotically stable and $e \to 0$.

To define the control law, we can rewrite equation 2.29 as:

$$\underbrace{\dot{x}_d - \dot{x}}_{\dot{e}} + K\underbrace{(x_d - x)}_{e} = \dot{x}_d - \underbrace{J\dot{q}}_{\dot{x}} + Ke = 0, \tag{2.30}$$

where $J = J_A \in \mathbb{R}^{m \times n}$ is the analytical Jacobian, since we are using the forward kinematics that maps the joint space q into the operational space x, and the velocity mapping is performed by J_A through $\dot{x} = [\dot{p}^T \quad \dot{o}^T]^T = J_A\dot{q}$ (see Section 2.5.2). Assuming $\dot{q} = u$ due to the kinematic hypothesis, 2.30 is rewritten as:

$$\dot{x}_d - J\underbrace{u}_{\dot{q}} + Ke = 0, \tag{2.31}$$

and we can finally defined the control law as:

$$u = J^{-1}\underbrace{(\dot{x}_d + Ke)}_{\nu}, \tag{2.32}$$

where $\nu = \begin{bmatrix}\nu_p^T & \nu_o^T\end{bmatrix}^T$, ν_p is the signal associated to the position control, and ν_o is the part related to the orientation control. Commonly, ν is called to be the *Cartesian control signal*, since it acts in the level of the operational space dimension.

2.6.2.1 Jacobian pseudo-inverse

Notice that 2.32 is only valid if J is invertible, i.e., if it is a square non-singular matrix, which only occurs if $r = n$ (non-redundant manipulator) and the robot is out of singular configurations (which makes J to lose rank). In case of redundant manipulators ($r < n$), we can use the Jacobian right pseudo-inverse matrix (Siciliano et al. 2009). Considering a full-rank matrix ($rank(J) = r$), we define the Jacobian right-pseudo-inverse matrix as:

$$J^\dagger = J^T(JJ^T)^{-1} \qquad JJ^\dagger = I, \tag{2.33}$$

where $rank(J^\dagger) = rank(J) = r$ when the robot is out of singularity configurations. Notice that $J^\dagger = J^{-1}$ when $r = n$. The final control loop block diagram is shown in figure:

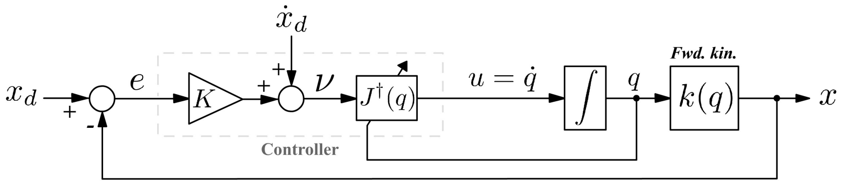


Figure 2.8: Block diagram: kinematic control.

A simpler method is to use the Jacobian transposed in place of $J^\dagger$ (Siciliano et al. 2009), in which the error converges only in regulation tasks ($\dot{x}_d = 0$). In this method, we expect a worse transitory, since the transposed matrix does not cancel the robot Jacobian.

2.6.3 Cartesian control law

The error signal is composed of both position and orientation errors, i.e.:

$$e = \begin{bmatrix} e_p^T & e_o^T \end{bmatrix}^T. \tag{2.34}$$

Assuming that the analytical Jacobian is being used, as demonstrated in 2.32, we can generalize the Cartesian control law as:

$$u = \dot{q} = J_A^{\dagger}\nu, \; J_A = \begin{bmatrix} J_{Ap}^T & J_{Ao}^T \end{bmatrix}^T$$

$$\underbrace{\begin{bmatrix} \nu_p \\ \nu_o \end{bmatrix}}_{\nu} = \underbrace{\begin{bmatrix} \dot{p}_d \\ \dot{o}_d \end{bmatrix}}_{\dot{x}} + \begin{bmatrix} K_p e_p \\ K_o e_o \end{bmatrix}. \tag{2.35}$$

Notice in (2.35) that, in the case of position control, we want to take the end-effector position $\vec{p}_{be}$ to a desired position $\vec{p}_d$. Assuming that the robot will be controlled from its base, let us define $p = (\vec{p}_{be})_b$, $p_d = (\vec{p}_d)_b$ and the position error as $e_p = p_d - p$, which we want to take to zero. The position control is considerably simple, since the end-effector linear velocity $\vec{v}_e$ is equal to $\dot{\vec{p}}$, and the three first rows (position part J_p) of the analytical Jacobian are equal to those of the geometric Jacobian. Hence, for position control, the law defined in (2.35) is always valid.

On the other hand, the definition of a Cartesian law for orientation control is more complex since:

- It depends on a particular representation of the end-effector orientation (e.g., euler angles, roll-pitch-yaw (RPY), unit *quaternions*), each one with its own oneness and complexity for implementation.
- The three last rows (orientation part J_{Ao}) of the analytical Jacobian (which depends on the chosen representation) are different from those of the geometric Jacobian. This happens because the time derivative of the orientation representation $\dot{o}$ is different from the end-effector angular velocity ω, as discussed in Section 2.5.3.

If we define the end-effector orientation by Euler or RPY angles $o = k(q)$, their time-derivative is given by the mapping of the orientation part of the analytical Jacobian J_A over the joint velocities, i.e., $\dot{o} = J_{Ao}\dot{q}$. Hence, the orientation error signal is defined as $e_o = o_d - o$, its dynamics is expressed by $\dot{e}_o = \dot{o}_d - \dot{o}$, and the orientation control law defined in 2.35 is valid.

For other approaches, it is suitable to use the geometric Jacobian in the control law. Since the position part is equal for both Jacobian types ($J_{Gp} = J_{Ap}$), the position Cartesian control law is the same, as follows:

$$\nu_p = \dot{p}_d + Ke_p. \tag{2.36}$$

Yet, the orientation part of the geometric Jacobian J_{Go} maps the joint velocities to the end-effector angular velocity ω. Hence, the orientation Cartesian control law is defined as follows:

$$\nu_o = \omega_d + Ke_\phi, \tag{2.37}$$

and the joint control law is defined as:

$$u = J_G^\dagger \begin{bmatrix} \nu_p^T & \nu_o^T \end{bmatrix}^T. \tag{2.38}$$

2.6.4 Quaternion-based orientation control law

Commonly, the Euler and RPY representations are the most used due to their physically intuitive meaning. However, these representations are not suitable for a closed control loop, because of their computational complexity, representation ambiguity for some configurations, and great non-linearity of the relationship between $\dot{o}$ (angle derivative) and ω (angular velocity), which turns very critical the stability analysis of the system without linearization.

A suitable choice for attitude representation is the *unit quaternion* (Siciliano et al. 2009), which is singularity free and more computational efficient. Let us define as $\tilde{q} = (\eta, \epsilon)$ as the unit quaternion representation of the robot attitude matrix $R = R_{be}$, and $Q_d = (\eta_d, \epsilon_d)$ as the unit quaternion representation of the desired attitude R_d, where η is the scalar part and ϵ is the vectorial part.

Yet, it is worth noticing that the Cartesian law defined in 2.35 cannot be used for quaternions, since it is not valid to define the quaternion error as $e_o = Q_d - Q$ and its dynamics as $\dot{e}_o = \dot{Q}_d - \dot{Q}$. A correct way to approach quaternion errors should be discussed though.

2.6.4.1 Orientation error via rotation matrix

Firstly, it is worth introducing how to define attitude error through rotation matrices. Consider that we want to take the robot current orientation $R = R_{be}(q)$ to a desired attitude $R_{be'} = R_d$. The orientation error R_ϕ can be defined using two different approaches.

In the *body frame approach*, the goal orientation is defined by post-multiplying R_{be} by R_ϕ (Siciliano et al. 2009) to achieve the desired attitude, i.e., $R_{be}R_\phi = R_{be'}$. Hence, R_ϕ is defined as a rotation around the current (or body) frame $\bar{E}_e$. Finally, the error is given by:

$$\underbrace{R}_{R_{be}} R_\phi = \underbrace{R_d}_{R_{be'}}$$

$$R_\phi = R^T R_d, \tag{2.39}$$

where $\bar{E}_e \to \bar{E}_{e'}$ when $R_\phi \to I$.

In the *inertial frame approach*, the goal orientation is defined by pre-multiplying R_{be} by $\bar{R}_\phi$ (Siciliano et al. 2009) to achieve the desired attitude, i.e., $\bar{R}_\phi R_{be} = R_{be'}$. Hence, R_ϕ is defined as a rotation of the robot pose with respect to the base (or inertial) frame $\bar{E}_b$. Finally, the error is given by:

$$\bar{R}_\phi \underbrace{R}_{R_{be}} = \underbrace{R_d}_{R_{be'}}$$

$$\bar{R}_\phi = R_d R^T, \tag{2.40}$$

where $R_\phi \to I$ when $\bar{E}_e \to \bar{E}_{e'}$.

Notice that a relationship between these two approaches can be found by combining 2.39 with 2.40, which yields:

$$R_d = RR_\phi = \bar{R}_\phi R \to \bar{R}_\phi = RR_\phi R^T. \tag{2.41}$$

2.6.4.2 Quaternion error

Using the body approach, the *orientation quaternion error* $R_\phi = R^T R$ is given by the quaternion product (Siciliano et al. 2009):

$$Q_\phi = Q^{-1} * Q_d. \tag{2.42}$$

Notice that $Q_\phi \to (1, 0_{1\times3})$ if and only if R aligns with R_d, which motivates us to define the *orientation error* as the vectorial part of Q_ϕ only, which is given by:

$$e_\phi = \epsilon_\phi = \eta\epsilon_d + \eta_d(-\epsilon) + \widehat{(-\epsilon)}\epsilon_d$$

$$e_\phi = \eta\epsilon_d - \eta_d\epsilon + \hat{\epsilon}_d\epsilon, \tag{2.43}$$

where "$\hat{}$" is the operator of the skew-symmetric matrix (Murray et al. 1994). Likewise, if we use the inertial approach to define the attitude error, than $\bar{R} = R_d R^T$, the unit quaternion error is defined as $\bar{Q}_\phi = Q_d * Q^{-1}$, and the orientation error is defined as:

$$e_\phi = \eta\epsilon_d - \eta_d\epsilon - \hat{\epsilon}_d\epsilon. \tag{2.44}$$

2.6.4.3 Control design and stability analysis

A suitable Cartesian control law for orientation is proposed in (2.37) as follows:

$$\nu_o = \omega_d + K_o e_\phi, \tag{2.45}$$

where $e_\phi = \eta\epsilon_d - \eta_d\epsilon + \hat{\epsilon}_d\epsilon$ is defined as in (2.43) using the body approach and the geometric Jacobian is being used. In the closed loop, we apply (2.45) in (2.38), which brings us to the following system:

$$\omega_d - \omega + K_o e_\phi = 0, \tag{2.46}$$

where $\tilde{\omega} = \omega_d - \omega$. Notice yet that this is a non-linear system, which stability cannot be ensured in the same way it was stated for the linear system generated by the use of the analytical Jacobian (equation 2.29). One way to approach this problem is to introduce the quaternion error propagation formula (Lizarralde & Wen 1996, Yuan 1988), which states a relationship between the angular velocity linear error $\tilde{\omega} =$ and the time-derivative of the quaternion error e_ϕ, as follows:

$$\dot{\eta}_\phi = \underbrace{-\frac{1}{2}\epsilon_\phi^T}_{J_{R\eta,\phi}}\tilde{\omega}, \tag{2.47}$$

$$\dot{e}_\phi = \dot{\epsilon}_\phi = \underbrace{\frac{1}{2}(\eta_\phi I - \hat{\epsilon}_\phi)}_{J_{R\epsilon,\phi}}\tilde{\omega}, \tag{2.48}$$

where $J_{R,\phi} = \begin{bmatrix} J_{R\eta,\phi}^T & J_{R\epsilon,\phi}^T \end{bmatrix}^T$ is called the quaternion representation Jacobian (see Section 2.5.3). Then, the system (2.46) can be rewritten as:

$$J_{R\epsilon}^{-1}\dot{e}_\phi + K e_\phi = 0. \tag{2.49}$$

To evaluate the system stability, the following positive-definite Lyapunov function (Slotine et al. 1991) is proposed:

$$V_o(\eta_\phi, e_\phi) = (\eta_\phi - 1)^2 + \epsilon_\phi^T \epsilon_\phi > 0, \tag{2.50}$$

where η_ϕ is the scalar part of the quaternion error Q_ϕ. Then, differentiating (2.50) with respect to time, we get:

$$\dot{V}_o = 2(\eta_\phi - 1)\dot{\eta}_\phi + 2\epsilon_\phi^T \dot{\epsilon}_\phi. \tag{2.51}$$

Applying (2.48) in the above equation, we get:

$$\dot{V}_o = 2(\eta_\phi - 1)(-\frac{1}{2}\epsilon_\phi^T)\tilde{\omega} + 2\epsilon_\phi^T \frac{1}{2}(\eta_\phi I - \hat{\epsilon}_\phi)\tilde{\omega}$$

$$\dot{V}_o = \cancel{-\eta_\phi \epsilon_\phi^T \tilde{\omega}} + \epsilon_\phi^T \tilde{\omega} + \cancel{\eta_\phi \epsilon_\phi^T} \tilde{\omega} - \epsilon_\phi^T \hat{\epsilon}_\phi \tilde{\omega} \tag{2.52}$$

Since $\tilde{\omega} = -K_o \epsilon_\phi$ from (2.46), $e_\phi = \epsilon_\phi$, and $\hat{a}a = 0, \forall a \in \mathbb{R}^{3\times1}$, if we consider K_o as a positive-definite diagonal matrix, then:

$$\dot{V}_o = \epsilon_\phi^T(-K_o \epsilon_\phi) - \epsilon_\phi^T \hat{\epsilon}_\phi (-K_o \epsilon_\phi) =$$

$$\dot{V}_o = -\epsilon_\phi^T K_o \epsilon_\phi \cancel{- K_o \epsilon_\phi^T \hat{\epsilon}_\phi \epsilon_\phi}$$

$$\dot{V}_o = -\epsilon_\phi^T K_o \epsilon_\phi \leq 0, \tag{2.53}$$

Then, once $V_o(\eta_\phi, e_\phi)$ is continuously differentiable, radially unlimited and positive-definite, and $\dot{V}_o$ is negative semi-definite for all states, we state by the *LaSalle's invariance principle* (Slotine et al. 1991) that all the system trajectories converge to the largest invariant set $\bar{\Omega}$ in:

$$\Omega = \{(\eta_\phi, e_\phi) : \dot{V}_o\} = \{(\eta_\phi, e_\phi) : e_\phi = 0\}. \tag{2.54}$$

By stating that $\tilde{\omega} = -K_o e_\phi$, we have $\tilde{\omega} = 0$ in the invariant set. Then, considering the quaternion constraint for unit-norm $\eta_\phi^2 + e_\phi^2 = 1$, we have:

$$\bar{\Omega} = \{(\eta_\phi, e_\phi) : \eta_\phi = 1,\ e_\phi = 0\}, \tag{2.55}$$

which implies that $(\eta_\phi, e_\phi) = (1, 0_{1\times3})$ is a globally asymptotically stable equilibrium.

2.6.5 Control of redundant manipulators

In case of redundant manipulators (as approached in 2.6.2.1), the extra DoFs can be used to maximize an objective function. To demonstrate, let us consider the control law:

$$u = J^{\dagger}[\dot{x}_d + K(x_d - x)]. \tag{2.56}$$

By observing the expansion of the Jacobian null space $J(I - J^{\dagger}J) = 0$, we can rewrite 2.56 as:

$$u = J^{\dagger}[\dot{x}_d + K(x_d - x)] + \mu(I - J^{\dagger}J), \tag{2.57}$$

which will not affect the error dynamics $\dot{e} + Ke = 0$. Notice that μ is the proposed additional DoF, which can be used to maximize an objective function $\omega(q) > 0$, and it is defined by:

$$\mu = \left(\frac{\partial\omega(q)}{\partial q}\right)^T. \tag{2.58}$$

The most classic examples of this function are:

- Mechanical joint range $q_{im} \leq q_i \leq q_{iM}$: $\omega(q) = -\frac{1}{2n}\sum_{i=1}^{n}\left(\frac{q_i - \bar{q}_i}{q_{iM} - q_{im}}\right)^2, /; \bar{q}_i = mean(q_{im}, q_{iM})$
- Distance to obstacles: $\omega(q) = max\,\|p(q) - O\|$
- Robot manipulability: $\omega(q) = \sqrt{det(JJ^T)}$

2.7 Interaction control

One of the most important manipulator performance indexes is its capacity to interact with the environment (Siciliano et al. 2009). The physical quantity that best represents the state of interaction is the *contact force* $\vec{F_c}$ between the manipulator end-effector and the manipulated surface/object.

Among the tasks that involves physical contact, we can highlight polishing, assembly, deburring, machining, drilling, cutting, button/switching pressing, etc. Interaction control between the manipulator and the environment is critical for the success of these tasks. It works by controlling the contact forces to a desired value, yet avoiding high contact forces, which are mostly unwelcome or inadmissible, since they may over-stress both the manipulator and the environment touched object.

2.7.1 Interaction control problem

During interaction, the environment naturally restricts the path geometries that the end-effector can describe, which is called *constrained motion*.

A simple way to solve the interaction problem would be to apply an *accurately planned motion control* of position and orientation, but this would depend on a strictly detailed model of both manipulator (kinematic and dynamic) and environment (geometry and mechanical properties). Although the robot model can be defined with sufficient accuracy, a good detailing of the environment is mostly hard to be obtained. It can also lead to uncertainties of the exact localization of the robot at the environment, which may interfere in the performance of the motion control.

In addition, if the environment is not well structured, path planning errors may cause the appearance of unplanned contact forces that lead the end-effector to deviate itself from the desired trajectory. An accurate control system would react to overcome this situation, which is considered as a system disturbance. However, this can lead to a situation in which the contact forces are high enough to stress the joint actuators or the mechanical parts to their saturation limit, causing hence a mechanical breakage. This is even more serious in environments with high stiffness.

To solve the interaction problem, we cannot only apply a motion control that does not manage the stiffness of the interaction situation. It is necessary to address a control strategy that guarantees a compliant behaviour during the contact. Hence, for tasks that require interaction, it is essential to control not only the motion (position and orientation), but also the force applied by the end-effector.

2.7.2 Interaction control methods

The methods for controlling interaction can be divided into two categories: those that control force *indirectly*, and those that control force *directly*.

In the first method, the force control strategies are realized via motion control without the explicit closure of a force feedback loop, within which we can highlight the classic *stiffness/compliance control* (Salisbury 1980) and the *impedance control* (Hogan 1985). In those techniques, the interaction forces are not measured neither controlled directly, but influenced by the stiffness/compliance of the environment and the manipulator, and then, controlled indirectly by the motion control of the end-effector pose. Some more recent studies, such as Homayounzade & Keshmiri

(2014), proposes an adaptive algorithm to compensate parametric uncertainties of the robot model for tasks at unknown environments, where the constraint forces are estimated only by the joints' positions and velocities.

However, if the given task requires a *more accurate* force regulation and/or path tracking, it is necessary to provide a control strategy that acts *directly* at a force feedback loop (hence, needing a force sensor to provide this feedback). Within such strategies, we can highlight the classic *hybrid control of force/position* (Raibert & Craig 1981) and the *parallel control* (Anderson & Spong 1988). Other more comprehensive methods, such as Leite (2005), Leite et al. (2009), combines force control with robot *visual servoing*, which permit the robot to localize and inspect objects in unstructured environments. Leite (2011) uses adaptive control to solve the problem of of uncertain parameters of the manipulator kinematics/dynamics, environment and camera. Some more recent studies, such as Bechlioulis et al. (2010), approaches *neuro-adaptive networks* for design of force/position control to solve former unresolved issues in contact maintenance during tasks, overshoot, better speed of response and higher accuracy level.

The cooperation of contact force information with virtual viewing is also of great importance for tasks performed by teleoperated robots, since the remote operator is better guided with force feedbacks at the controller/joysticks to accomplish the required goals with more safety and accuracy. Examples of applications related to this subject: robots for cargo transportation (Ribeiro 2013), maintenance in nuclear plants (Meggiolaro et al. 1999), and medical surgery.

2.7.3 Contact modeling

At this moment, it is important to understand the involved contact forces between the end-effector and the environment during the interaction. Murray et al. (1994) describes different approaches for modeling this situation, in which the end-effector is a fingertip tool that only grasps an object at a single point with fixed location. This means that we neglect that this finger can roll or slide along the surface. We also ignore the kinematic and dynamics of the finger motion, only focusing on the forces transmitted between the contact.

2.7.3.1 Frictionless interaction

The first approach depicts a *frictionless contact point*. To explain, let us first consider that the tool touches a surface γ at a contact point without friction, applying to it a force $\vec{f}_{ce}$. In this case, the surface reacts only with a normal contact force $\vec{f}_{cs} = \vec{f}_{ns}$. In practice, perfectly frictionless cases are never found, yet, the model above can be suitable for describing contact situations in which the friction between the tool and the surface is negligible or unknown. However, this situation requires that the end-effector only applies a force perfectly perpendicular to the surface, otherwise, $\left\|\vec{f}_{ce}\right\| \neq \left\|\vec{f}_{cs}\right\|$ and the tool slides indefinitely over the surface due to the remaining resulting force.

2.7.3.2 Static friction

For a more generalized situation, we need a model provides adequate description of both normal and friction forces, such as the *Coulomb friction*. This model establishes that a friction force arises tangentially over the surface and its magnitude is proportional to the normal force applied by the tool. To clarify, let us consider the situation illustrated by Figure 2.9, in which the tool applies a force $\vec{f}_{ce}$ over a locally flat surface γ, with *coefficient of static friction* $\mu_s < 1$ between them. Let us define $\vec{f}_{ce} = \vec{f}_{te} + \vec{f}_{ne}$, where $\vec{f}_{ne}$ is a normal force with respect to the surface, and $\vec{f}_{te}$ is the tangent force. To maintain the dynamic balance, the surface should react with a contact force $\vec{f}_{cs} = \vec{f}_{ns} + \vec{f}_{ts}$, in opposite direction but equal in magnitude.

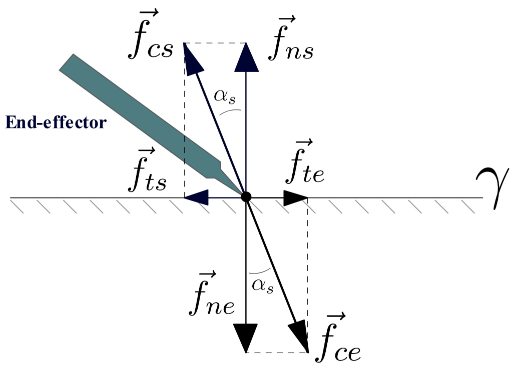


Figure 2.9: Coulomb friction interaction model.

The Coulomb Law states that this dynamic balance can only be ensured if:

$$\left\|\vec{f}_{te}\right\| \leq \underbrace{\mu_s \left\|\vec{f}_{te}\right\|}_{fr_{max}}, \tag{2.59}$$

i.e., the tool will remains in contact at the same point without sliding if the tangential force that it applies on the surface does not exceed in magnitude the *maximum static friction* fr_{max} supported by between them. It is clear to state that equation 2.59 can be geometrically represented as a region (called *cone of friction*) within which the direction of the force applied by the tool must remain. To calculate the angle α_s that defines this cone, let us consider the situation in Figure 2.10, in which the tool is about to slide ($\left\|\vec{f}_{te}\right\| = fr_{max}$). Then:

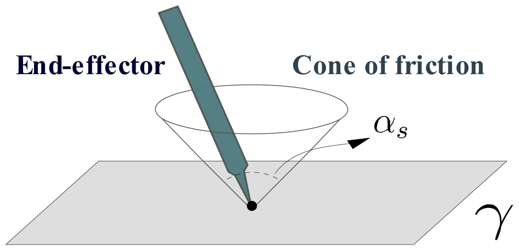


Figure 2.10: Cone of friction.

$$\underbrace{\cancel{\left\|\vec{f}_{ce}\right\|} sin(\alpha_s)}_{\left\|\vec{f}_{te}\right\|} = \mu_s \underbrace{\cancel{\left\|\vec{f}_{cs}\right\|} cos(\alpha_s)}_{\left\|\vec{f}_{ns}\right\|}$$

$$\alpha_s = tan^{-1}(\mu_s). \tag{2.60}$$

A more realistic contact model allows not only the involved torques, but also the applied torques during the interaction. This model, called as *soft-finger contact*, is suitable for very specific applications that are not being approached in this work.

2.7.3.3 Kinetic friction

Once the tangential applied force exceeds the static limit and the tool starts sliding, a smaller friction force is verified, and its magnitude is given by:

$$fr_k = \mu_k \left\|\vec{f}_{te}\right\|, \mu_k < \mu_s, \tag{2.61}$$

where μ_k is the *coefficient of kinetic friction*. If a given task requires the tool to move along a path over a surface and the friction forces must be considered, we must consider both static and dynamic frictions in the contact model. Since the kinetic friction is smaller, once the tool starts to move, it can be controlled with a contact force of smaller magnitude.

2.7.4 Constrained motion

The constrained motion of a manipulator tool over a rigid surface (considered ideally smooth, for simplicity) is represented by the position p of the contact point between them, and the constraint surface is described by:

$$\Psi(p) = 0. \tag{2.62}$$

If the surface is an unlimited flat plane, for example, then $\Psi(p) = ap_x + bp_y + cp_z + d = 0,\ \{a, b, c, d\} \in \mathbb{R}$. Thus, if we consider that the end-effector is controlled to be always in contact with the surface of interest, then the end-effector position must always satisfy 2.62. Therefore, the constrained motion of the end-effector is given by:

$$\mathcal{V}(p)^T \dot{p} = 0 \qquad \dot{\mathcal{V}}(p)^T \dot{p} + \mathcal{V}(p)^T \ddot{p} = 0, \tag{2.63}$$

where $\mathcal{V}(p) = \left(\frac{\partial \Psi}{\partial p}\right)$ is a vector with the same direction of the normal vector in p. On the other hand, the torque applied to each joint is given by $\tau = J^T F$, where $F = \begin{bmatrix} f_e & \tau_e \end{bmatrix}^T$ represents the forces/torques exerted over the tool in the task space (Siciliano et al. 2009). Since we are assuming a frictionless situation, then $F = \begin{bmatrix} f & 0 \end{bmatrix}^T$ and the contact force and $\mathcal{V}(p)$ have the same direction. Thus, $f = \alpha \mathcal{V}$, where $\alpha \in \mathbb{R}^*$ is a constant that defines f magnitude. Then, we can rewrite equation 2.63 as follows:

$$f^T \dot{p} = 0. \tag{2.64}$$

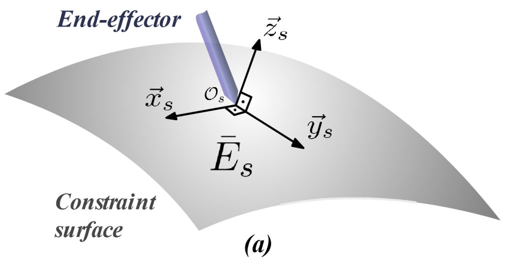
Notice that it is worth using 2.64 instead of 2.63 because f is a contact force that can be measured, while the normal vector $\mathcal{V}$ is generally unknown. Yet, if there is friction between the contact surface and the tool tip, a better model should be proposed to found $\mathcal{V}(p)$, as will be presented in Section 2.7.9.

2.7.5 Artificial and natural constraints

To design a closed loop system that controls the end-effector motion and contact force over a constrained geometry Ψ, it is worth first understand the concept of *artificial and natural constraints.*

The *natural constraints* are the position, velocity and force limitations naturally imposed by the environment geometry. Hence, the robot can control only the variables that are not subjected to natural restrictions. These variables are the DoFs over which the designer can establish *artificial constraints*, in other words, the system control objectives (such as reference trajectories or points).

Notice that the set of natural and artificial constraints are complementary, since they are composed of all the involved task variables (position, velocity and force in all directions). It is worth illustrating this concept with the situation of the interaction between the manipulator end-effector and a generic surface (Figure 2.11a). Also, it is valid to introduce the constraint or surface coordinate system $\bar{E}_s$, whose origin $\mathcal{O}_s \in \Phi$. Also, we arbitrarily define $\vec{z}_s$ as a normal vector with respect to a flat plane γ tangent in $\mathcal{O}_s$ with respect to Ψ and pointing "inside" the surface, and $\vec{x}_s, \vec{y}_s \in \gamma$.



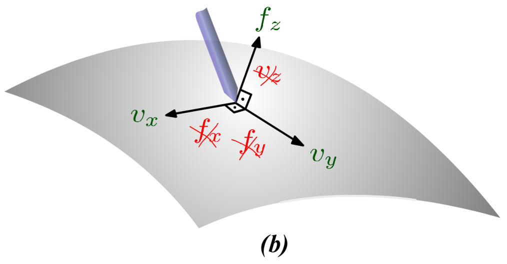


Figure 2.11: Artificial and natural constraints: (a) constraint coordinate system $\bar{E}_s$; (b) velocity and force constraints in $\bar{E}_s$.

As we can notice in Figure 2.11b, it is impossible to develop an arbitrary velocity along $\vec{z}_s$ or angular velocities around $\vec{x}_s$ and $\vec{y}_s$, due to the natural constraints imposed by this surface environment. Likewise, it is impossible to develop an arbitrary force along $\vec{x}_s$ and $\vec{y}_s$, and an arbitrary torque around $\vec{z}_s$. However, a control system can be established to impose artificial constraints to the DoFs that are not naturally restricted. Therefore, the manipulator can arbitrarily control linear velocities and position in $\vec{x}_s$ and $\vec{y}_s$, and control the end-effector force along $\vec{z}_s$. Analogously, it can control applied torques around $\vec{x}_s$ and $\vec{y}_s$ and angular velocity around $\vec{z}_s$. These complementary constraints are summarized in Table 2.1.

Table 2.1: Artificial and natural constraints

Variable	Along or around $\vec{x}_s$ and $\vec{y}_s$	Along or around $\vec{z}_s$
Velocity (v)/ position (p)	OK (artificial)	NO (natural)
Force (f)	NO (natural)	OK (artificial)
Angular vel. (ω)/ angle (θ)	NO (natural)	OK (artificial)
Torque (τ)	OK (artificial)	NO (natural)

2.7.6 Force control problem

Let us consider the problem of a kinematic control of force for a manipulator already in interaction with a generic surface, in which we assume that the contact force applied to the end-effector can be measured by a coupled force sensor. Our goal is to lead the contact force f to a desired value f_d. We can simply define the force error as:

$$e_f = f_d - f. \tag{2.65}$$

We can map, without losing of generality, the relationship between the measured force f at the end-effector and the joint torques by $\tau = J^T(q)f$ (Siciliano et al. 2009), being J the geometric Jacobian. As a kinematic control is being considered (Section 2.6), then, the arm motion is given by $\tau = \dot{q}$. Then, a Cartesian law of force can be mapped to a control signal to the manipulator joints:

$$\dot{q} = J^T \nu_f. \tag{2.66}$$

By defining a Cartesian law with proportional and feed-forward strategies ($\nu_f = \dot{f}_d + K_p e_f, K_p = K_p^T > 0$) and considering the manipulator differential kinematics $\dot{x} = J\dot{q}$, we have:

$$\dot{x} = JJ^T(\dot{f}_d + K_p e_f). \tag{2.67}$$

The measured force dynamics can be modeled as a spring with elastic constant K_m by the Hooke law as $f = K_m x, K_m = K_m^T > 0$. From 2.67, this leads us to the following error dynamics:

$$\dot{e}_f = \dot{f}_d - K_m \underbrace{JJ^T(\dot{f}_d + K_p e_f)}_{\dot{x}} - (I - K_m JJ^T)\dot{f}_d - K_m JJ^T K_p e_f$$

$$e_f + JJ^T K_m K_p e_f = (I - JJ^T K_m)\dot{f}_d. \tag{2.68}$$

Clearly, the error dynamics is a non-linear system, whose asymptotic stability can be stated by the *Lyapunov* method (Slotine et al. 1991). A Lyapunov function $V(e_f)$ can be chosen as:

$$2V_f(e_f) = e_f^T K_p e_f, \tag{2.69}$$

being $V_f > 0, K_p > 0$. Differentiating 2.69 with respect to time and considering 2.68, we have:

$$\dot{V}_f = \dot{e}_f^T e_f + e_f^T \dot{e}_f = e_f^T(I - K_m JJ^T)K_p \dot{f}_d - K_m JJ^T K_p e_f. \tag{2.70}$$

In the case of force regulation ($\dot{f}_d = 0$) and assuming that J is a full-rank matrix, then $\dot{V}_f = -K_m J J^T K_p e_f$. This lead us to state $V_f > 0, \dot{V}_f < 0$ conditions, which implies that $e_f \to 0$, i.e., the system is asymptotically stable. For tracking a time-varying force function $f(t)$, $\dot{f} \neq 0$ and the chosen Lyapunov function cannot be used to state the system asymptotic stability.

2.7.7 Hybrid control of force and position: overview

The description of an interaction task concerning the *artificial and natural constraints* (Section 2.7.5) requires an strategy that uses the artificial constraints to specify the system goals and control only the variables that are not subjected to the natural constraints. This causes the control action not to affect the variables that are naturally restricted by the environment, which would lead to a conflict between the control and the interaction constraints and, hence, to an undesired system behavior.

Since theses constraints involve both force and position variables, such a control system is called to be *hybrid* (Raibert & Craig 1981). It proposes an strategy that combines position/velocity with force/torque, which considers that both motion and force are within two complementary work-subspaces (Mason 1981). Thereby, the force and position controls can be split into two different loops that do not affect each other. The element that performs this separation and defines which variables should be controlled by either force or position is the *selection matrix* S. Through this approach, the hybrid controller uses the matrix S to split the force and the position control loops that acts over error signals computed at the constraint surface coordinate system $\bar{E}_s$. Hence, the control laws can be designed independently to ensure that the goals for the position and force artificial constraints are achieved.

The hybrid control law is defined as:

$$\nu_h = \nu_{hf} + \nu_{hp}, \tag{2.71}$$

where ν_{fh} and ν_{ph} are the decoupled control signals acting in the force and position subspaces, respectively. It is worth remembering to transform both control signals to the coordinate system that it will be controlled.

To exemplify, let us suppose that a given task requires position control in x, y axes and force control in z-axis (perpendicular to the contact surface). We define the *force selection matrix* as:

$$S_f = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.72}$$

which will cancel the force control efforts in the complementary DoFs. The *position selection matrix* is complementary to S_f, and it is defined as:

$$S_p = (I - S_f) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{2.73}$$

Let us suppose that the given manipulator is being controlled from its base using its geometric Jacobian $(J_e)_b$ with respect to the base. Then, equation (2.71) must be represented as:

$$(\nu_h)_b = (\nu_{hf})_b + (\nu_{hp})_b. \tag{2.74}$$

Generally, the position error of the end-effector is defined with respect to the base, i.e., $p = \vec{p}_{be} \to (e_p)_b = (p_d)_b - p_b$, whereas the force error is defined with respect to the end-effector frame $(e_f)_b = f_d - (f_c)_e$, because the contact force $(f_c)_e$ is often measured at a force sensor coupled to the manipulator tool. In hybrid controllers, it is preferable a faster response in position control and smaller offset in force control. Thus, the position control signal is defined with both proportional (P) and feed-forward strategies as:

$$(\nu_p)_b = (\dot{p}_d)_b + K_{pp}(e_p)_b. \tag{2.75}$$

On the other hand, the force control signal is defined with a PI strategy as:

$$(\nu_f)_e = K_{pf}(e_f)_e + K_{if} \int_{0^-}^{\infty} (e_f)_e, \tag{2.76}$$

where we shall notice that the feed-forward signal is neglected since we are are assuming only force regulation ($\dot{f}_d = 0$, see Section 2.7.6). Now, the control signals are split into two loops apart. Since the selection matrix decouples only at the constraint frame $\bar{E}_s$ (located at the contact surface), the control signals must be firstly represented in $\bar{E}_s$, then operated by S, and, finally, represented at the base frame, as follows:

$$(\nu_{hf})_b = R_{bs} S_f R_{se} (\nu_f)_e \tag{2.77}$$

$$(\nu_{hp})_b = R_{bs} S_p R_{sb} (\nu_p)_b. \tag{2.78}$$

Finally, a block diagram of the hybrid control of force and position is shown in Figure 2.12.

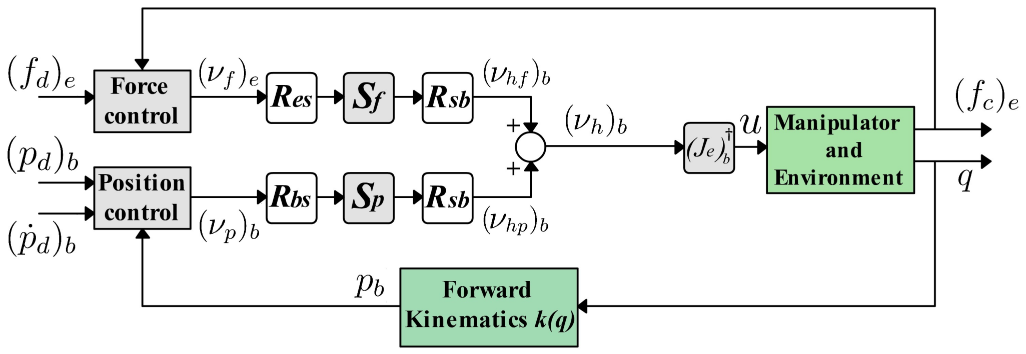


Figure 2.12: Block diagram: hybrid control (controlling from base).

2.7.8 Hybrid control considering known surface

This problem considers the control of force/position (hybrid) and orientation of a manipulator over a known smooth surface (with constraint coordinate frame $\bar{E}_s$) without losing contact during the entire task (Figure 2.13). Hence, our goal is to regulate forces only in $\vec{z}_s$ direction and track a path only in $\vec{x}_s, \vec{y}_s$ directions (i.e., a projected path over the surface). In addition, this means that the end-effector shall be posed within the cone of friction so as not to slide. Generally, we want to keep it always perpendicular to the constraint surface (i.e., $\vec{z}_e \rightarrow \vec{z}_s$), so that the friction forces $f_{x,y}$ over the surface are cancelled and the normal force f_z is the only non-null component.

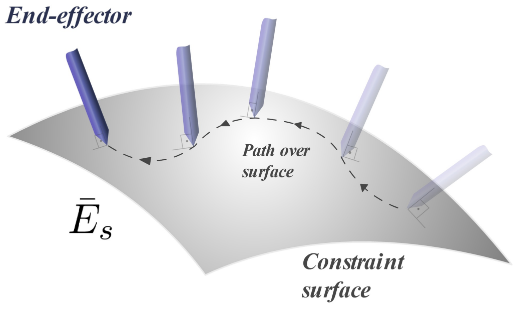


Figure 2.13: Hybrid control over known surface - following a path without losing contact.

Considering that the manipulator is being controlled using its geometric Jacobian with respect to the tool frame $(J_e)_e$, the signals of hybrid control of force/position are given by:

$$(\nu_h)_e = \underbrace{R_{es}S_fR_{se}(\nu_f)_e}_{(\nu_{hf})_e} + \underbrace{R_{es}S_pR_{sb}(\nu_p)_b}_{(\nu_{hp})_e}, \tag{2.79}$$

where $(\nu_f)_e$ and $(\nu_p)_b$ are defined as in equations 2.76 and 2.75, respectively.

Now, for the orientation kinematic control problem, we assume that the end-effector is always in contact with the surface and the robot shall lead its current orientation $R = R_{be}$ to a desired orientation R_d. As seen in Section 2.6.4.1, we can define the matrix orientation error using both inertial or body approaches. In this case, since we are using the Jacobian reference to the end-effector (body) frame, it is suitable using the body approach, which gives us the following orientation error matrix:

$$R_\phi = R^T R_d. \tag{2.80}$$

Considering $q_\phi = (\eta_\phi, \epsilon_\phi)$ as the unit quaternion (Siciliano et al. 2009) associated with R_ϕ, the following control law can be adopted:

$$(v_o)_e = K_{po}\epsilon_\phi + \omega_d, \tag{2.81}$$

where $K_{po} = K_{po}^T > 0$ is a positive-definite matrix, and $\omega_d = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$, since the end-effector does not vary during the entire contact.

Then, assembling both hybrid and orientation control signals in $v_e = \begin{bmatrix} (\nu_h)_e & (\nu_o)_e \end{bmatrix}$, the kinematic control is performed by the following sentence:

$$\dot{q} = u = (J_e)_e^\dagger \nu_e, \tag{2.82}$$

where, in this case, the Jacobian pseudo-inverse $J^\dagger$ is being used. The complete block diagram of the hybrid control of force/position and orientation over known surfaces is found in Figure 2.14.

2.7.9 Hybrid control over uncertain surface

This problem considers the control of force/position (hybrid) and orientation of a manipulator over surfaces whose geometry are unfamiliar or its parameters are quite different from what is known. In such case, the constraint coordinate system $\bar{E}_s$ is either uncertain of may vary slightly with respect to a initial known information about the surface. Thus, it is suitable to present strategies for the estimation of the

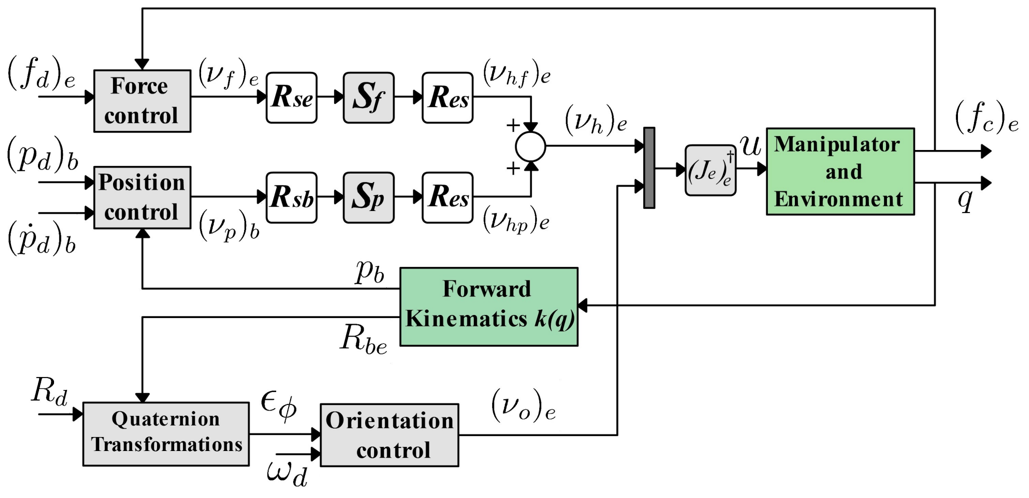


Figure 2.14: Block diagram: hybrid control + orientation control over known surfaces (controlling from end-effector).

geometric constraint parameters and for the *online* orientation control based on the current estimation.

In such case, we shall estimate the orientation of the surface geometry with respect to the base $\tilde{R}_{bs}$. In case of mobile robots, R_{bs} may be unknown also due to cumulative errors of the forward kinematics or to robot localization inaccuracy.

Several strategies approaches the estimation of the surface coordinate frame, each one with its advantages and restrictions. In this work, we present two different methods to find this estimation, which consider a priori that the end-effector pose is not to far from the actual surface and it points always toward it, i.e., $\vec{z}_e \cdot \vec{z}_s > 0$.

2.7.9.1 Normal vector re-estimation via infinitesimal displacement

This strategy, approached by Leite et al. (2009), is suitable for online estimation of the surface normal vector. Basically, upon in contact with the surface, the end-effector attempts to perform a sufficient small displacement $\vec{\Delta}$ over the surface. This displacement can be performed deliberately or naturally during a path tracking in contact with the surface. Since $\bar{E}_s$ is unknown, the end-effector actually performs a displacement $\tilde{\vec{\Delta}} = \tilde{M} - \tilde{T}$ (Figure 2.15), which can be measured via forward kinematics with respect to the base frame. From this displacement, we can define an unit vector $\vec{t} = \frac{\tilde{\vec{\Delta}}}{\left\|\tilde{\vec{\Delta}}\right\|}$ that certainly tangent the surface.

Assuming the Coulomb friction as the adopted contact model (Section 2.7.3) and the contact force always applied within the cone of friction (Section 2.7.3.2), during

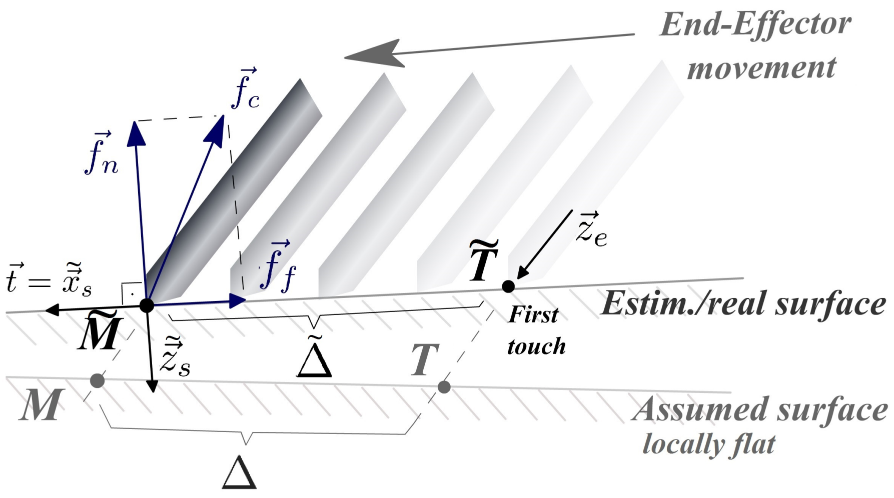


Figure 2.15: Displacement over surface and contact force arising.

the tool displacement, we state that the contact force $\vec{f}_c$ can be decomposed into two components (Figure 2.15). The first is the friction force $\vec{f}_f$, which is a projection of $\vec{f}_c$ over the tangent vector $\vec{t}$, i.e.:

$$\vec{f}_f = \frac{\vec{t}\,\vec{t}\cdot}{\vec{t}\cdot\vec{t}}\vec{f}_c \tag{2.83}$$

The second is the normal contact force $\vec{f}_n$, which is given by:

$$\vec{f}_n = \vec{f}_c - \vec{f}_f = \left(\mathcal{I} - \frac{\vec{t}\,\vec{t}\cdot}{\vec{t}\cdot\vec{t}}\right)\vec{f}_c \tag{2.84}$$

It is worth noticing that the surface normal vector (unit) that we are looking for is collinear with the normal force and points to the other direction, hence:

$$\tilde{\vec{z}}_s = -\frac{\vec{f}_n}{\left\|\vec{f}_n\right\|} \tag{2.85}$$

Finally, it is suitable to define $\tilde{\vec{x}}_s = \vec{t}$ and $\tilde{\vec{y}}_s = -\tilde{\vec{x}}_s \times \tilde{\vec{z}}_s$ using the right-hand rule. Since all the vectors above can be computed with respect to the base frame, the estimation $\tilde{R}_{bs}$ is given by:

$$\tilde{R}_{bs} = \begin{bmatrix} \widetilde{(\vec{x}_s)}_b & \widetilde{(\vec{y}_s)}_b & \widetilde{(\vec{z}_s)}_b \end{bmatrix}. \tag{2.86}$$

This approach is suitable for flat surfaces and, the smaller the displacement Δ is realized, more satisfactory results will be provided for curved surfaces. If we consider an *infinitesimal* displacement $\Delta \to d\Delta$, this method is valid for all smooth surfaces, since $M \to T$ and the surface is considered to be locally flat always. Notice that this approach fails if there is small or no friction between the surface and the tool tip, since it would not generate a tangent force, which is the tangent vectorial component that is needed for the normal vector estimation.

This method is suitable for tasks that require constant displacements (*path tracking*) of the tool along the surface, since a new $\tilde{R}_{bs}$ estimation is provided at every instant upon an small displacement appearance.

2.7.9.2 Flat plane definition by three contact points

This strategy considers the theorem that three non-collinear points defines an unique flat plane. Using this approach, the end-effector shall attempt to be positioned over three arbitrary non-collinear points A, B, C over the contact surface. In practice, when the end-effector touches the surface - assuming that this contact occurs within the cone of Coulomb friction (Section 2.7.3.2) - it actually touches the points $\tilde{A}, \tilde{B}, \tilde{C}$ (Figure 2.16), whose positions with respect to a known coordinate system (e.g., $\bar{E}_b$) can be measured via forward kinematics. Two vectors can be defined though: $\vec{v} = \tilde{B} - \tilde{A}, \vec{w} = \tilde{C} - \tilde{A}$. A normal vector between them is found via the cross product by:

$$\vec{n} = \vec{v} \times \vec{w}. \tag{2.87}$$

Since we are assuming that the end-effector is already pointing to the surface, the

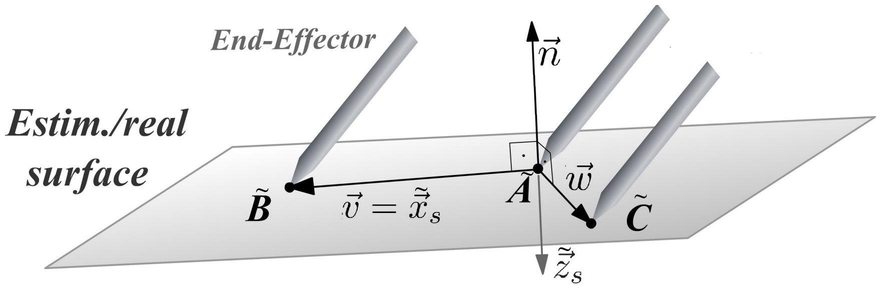


Figure 2.16: Definition of a flat plane by three non-collinear points.

elementary vector $\vec{z}_s$ is defined according to the direction of the normal vector $\vec{n}$ (Figure 2.16). Hence:

$$\tilde{\vec{z}}_s = \begin{cases} \frac{\vec{n}}{\|\vec{n}\|}, & \text{if } \vec{n} \cdot \vec{z}_e > 0 \\ -\frac{\vec{n}}{\|\vec{n}\|}, & \text{if } \vec{n} \cdot \vec{z}_e < 0 \end{cases} \tag{2.88}$$

A tangent vector is already defined by either $\vec{v}$ or $\vec{w}$. Arbitrarily, the elementary vector $\vec{x}_s$ is defined as:

$$\tilde{\vec{x}}_s = \frac{\vec{v}}{\|\vec{v}\|}, \tag{2.89}$$

and the frame is completed using the right-hand rule, which yields $\widetilde{\vec{y}}_s = -\widetilde{\vec{x}}_s \times \widetilde{\vec{z}}_s$. To find $\tilde{R}_{bs}$, it is worth representing the vectors with respect to the base frame. Hence, considering $v = \tilde{B}_b - \tilde{A}_b$ and $w = \tilde{C}_b - \tilde{A}_b$, the surface geometry estimation is given by $\tilde{R}_{bs} = \left[\widetilde{(\vec{x}_s)}_b \quad \widetilde{(\vec{y}_s)}_b \quad \widetilde{(\vec{z}_s)}_b\right]$, where:

$$\begin{cases} \widetilde{(\vec{x}_s)}_b = \frac{v}{\sqrt{v^T v}} \\ \widetilde{(\vec{z}_s)}_b = sign\left[(\hat{v}w)^T (\vec{z}_e)_b\right] \frac{\hat{v}w}{\sqrt{(\hat{v}w)^T(\hat{v}w)}} \\ \widetilde{(\vec{y}_s)}_b = -\widehat{\widetilde{(\vec{x}_s)}_b}(\vec{z}_e)_b \end{cases} \tag{2.90}$$

This method is suitable to estimate the geometry of only flat surfaces, hence, it cannot be applied to a curved geometry, unless the distances between the three points are sufficient small. In flat surface applications, it can generate a very accurate estimation of $\tilde{R}_{bs}$, which gets better the greater are the distances between the contact points. In addition, this method does not utilize any force measurements in the estimation process, which turns out an advantage, since force measurements are generally noisier and less accurate than the joint position/velocity values provided from the actuator encoders. The force measurement is utilized only as a flag to indicate that the end-effector is in contact with the surface.

A limitation of this approach is that one cannot perform online *estimation* during a *path track over the surface*, since these two steps cannot be performed in parallel. Also notice that a frictionless situation would change the manner that this method is applied, but not make it impossible to be performed. In this case, the tool sliding will occur, but it is possible to save the position of three different non-collinear points during the contact period.

2.7.9.3 Comparison between the proposed estimation techniques

Table 2.2 summarizes the main advantages and disadvantages of both estimation techniques approached in this work.

2.7.9.4 Block diagram

With the estimation of R_{bs}, we consequently estimate R_{es}. Figure 2.17 shows a generic block diagram of hybrid and orientation control with estimation of the geometry of the constraint surface.

Table 2.2: Comparison between surface estimation techniques

Technique	Infinitesimal displacement	Three points
Uses information from:	Joint positions q Fwd. kinematics $T_{be}(q)$ Measured contact force $(\vec{F}_c)_e$	Joint positions q Fwd. kinematics $T_{be}(q)$
Surface geometry:	Any smooth (for infinitesimal or sufficiently small displacements	Flat plane
Friction:	Coulomb friction	Coulomb friction or no friction
Tool restriction:	Within cone of friction	Within cone of friction
Accuracy:	Good (but depends on the force sensor noise)	Good
Online estimation:	Yes	No

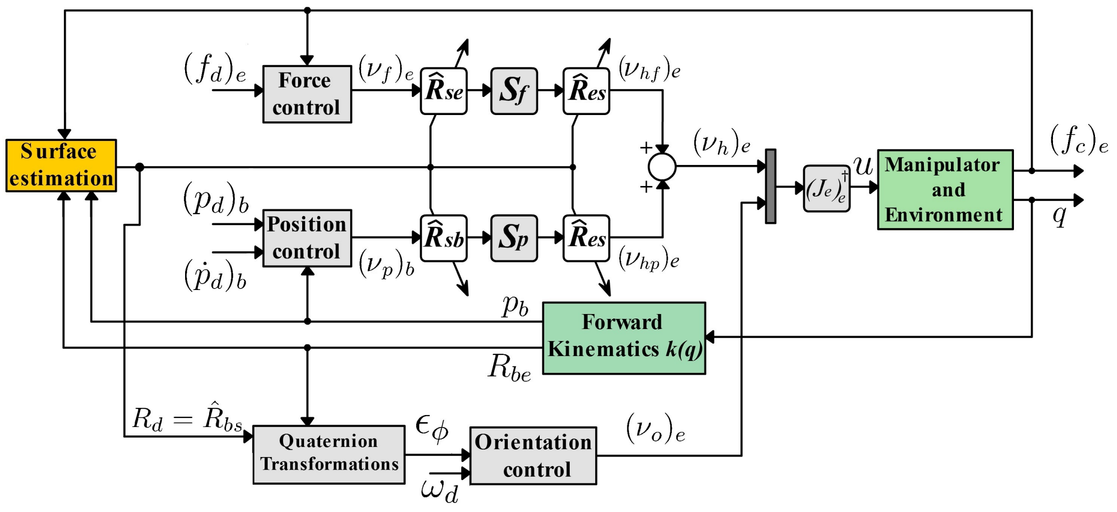


Figure 2.17: Block diagram: hybrid control + orientation control over unknown surfaces (controlling from end-effector).

2.8 Filtered inverse method (FI)

As discussed in this Chapter, finding the robot *inverse kinematics* via iterative approach and *kinematic control* are two sightly similar tasks that depends on the online calculation of the Jacobian inverse J^{-1} (or pseudo-inverse $J^\dagger$) matrix. As discussed in 2.5.4, the computation of the Jacobian inverse becomes a difficult task at singularity configurations and even near them. In these situations, undesirable great velocities can be generated at joint level.

In classic control though, singularities are completely avoided, as well as their neighborhood. This reduces the possibilities for the paths that the robot can follow, and the computation of Jacobian inverse leads to a close-to-singular or ill-conditioned matrix, causing hence inaccurate results. Some established solutions already approach the problem of J^{-1} computation, such as the *DLS* and the *FIK*.

The DLS method (Nakamura & Hanafusa 1986, Siciliano et al. 2009) proposes that J^{-1} or $J^{\dagger}$ is replaced by:

$$J^{*} = J^{T}(JJ^{T} + \delta I)^{-1}, \tag{2.91}$$

where $\delta \geq 0$ is a damping/regulation factor (Tikhonov & Arsenin 1977) that makes the inverting process better numerically conditioned. It is clear to see that this factor introduces a trade-off (Mayorga et al. 1993) between the accurate computation of $J^{\dagger}$ (with $\delta \approx 0$) and the feasibility to find $J^{\dagger}$ (with $\delta >> 0$). The adaptation of δ is an important research subject until today, as approached by Wampler et al. (1986) and Chiaverini et al. (1994). One method, proposed by Nakamura & Hanafusa (1986), is to couple up δ with the robot manipulability (e.g., $\omega = \sqrt{det(JJ^{T})}$, see 2.5.5) by:

$$\delta = \begin{cases} 0, \omega \geq \omega_0, \delta_o \left(1 - \frac{\omega}{\omega_0}\right), \omega < \omega_0, \end{cases} \tag{2.92}$$

where δ_0 is a scaling constant in singularity configurations, and ω_0 defines a neighborhood limit near a singularity, which can be defined according to the manipulator mechanics. This method creates a joint speed damping around the neighborhood of a singularity, which causes the robot to deviate its reference trajectory when near these situations. Thus, δ can be understood as a weighting factor that privilege either accurate solutions with low robustness near singularities (for small δ values) or low tracking accuracy with feasible and robust solutions (for high δ values).

On the other hand, the FIK method (Pechev 2008) is one of the most recent proposed approaches to deal with matrix inversion near singularities. This strategy introduces a feedback loop that minimizes the error between the demanded and the actual velocity $e_{\nu} = \nu - J\dot{q}$, hence leading to the system $\dot{q} = K(s)(JK(s) + T)^{-1}\nu$. There is an suitable adaptive law for $K(s)$ that leads the error e_{ν} to an arbitrarily small value, and, hence, $K(s)(JK(s) + T)^{-1} \rightarrow J^{-1}$, which makes the Jacobian matrix inversion not necessary. This method is considerably robust near singularities and computationally efficient, if compared to classic methods based on the pseudo-inverse application.

There are other recent methods that attempt to solve the inverse kinematics problem using more complex strategies, such as generic algorithms (dos Santos Scofano et al. 2005), and dual-quaternion/Davies method (de Oliveira et al. 2014). Vargas (2013) proposes an elegant approach to estimate $J^\dagger$ online and dynamically in a system closely similar to a first-order filter. Among its main features, we highlight: (i) possibility of the robot to quickly pass near (or even over) kinematic singularities; (ii) possibility to weigh and prioritize different control goals/cost functions; (iii) consideration of additional constraints.

2.8.1 Main concepts: SISO case

To present the method, it is worth to first to understand the problem of a simple scalar first-order Single-Input Single-Output (SISO) system:

$$\dot{y} = \zeta(t)u, \tag{2.93}$$

where $u \in \mathcal{L}_\infty$ is a limited system input or control signal, y is the system output, and $\zeta(t)$ is a non-linear time-varying scalar function. Assume that the control goal is to track a reference trajectory $y_d(t)$ in order to guarantee the asymptotic convergence of the tracking error $e := y_d - y$ to zero. A control law $u(t)$ which linearizes the control system in (2.93) and ensures the achievement of the control goal is given by:

$$u = \zeta^{-1}(t)\nu, \quad \nu = \dot{y}_d + Ke, \tag{2.94}$$

where $K > 0$. Notice that this control law uses an instantaneously computed inverse $1/\zeta$. Now, consider using a time-varying function $\theta(t)$ dynamically updated so that:

$$\zeta\theta \to 1. \tag{2.95}$$

Thus, the following error signal is introduced to ensure a suitable stable dynamics for θ, as follows:

$$S = \zeta\theta - 1. \tag{2.96}$$

To analyze the error stability, let us consider the Lyapunov function (Slotine et al. 1991) as $V(S) = S^2/2$. Its time-derivative is given by:

$$\dot{V} = S\dot{S} = S(\dot{\zeta}\theta + \zeta\dot{\theta}) \tag{2.97}$$

By analysing 2.97, a suitable choice for the *parametric update law* is given by:

$$\dot{\theta} = \gamma S\zeta, \tag{2.98}$$

where $\gamma > 0$. Hence, we get:

$$\dot{V} = S\dot{\zeta}\theta - \gamma S^2\zeta^2. \tag{2.99}$$

If we consider $\zeta(t)$ as a constant function $\dot{\zeta} = 0$, then the stability analysis becomes simpler, and we state that $\dot{V} \leq 0$. In this case, the parametric update law in 2.98 is written as:

$$\dot{\theta} = -\gamma(\zeta\theta - 1)\zeta = -\gamma\zeta^2\theta + \gamma\zeta, \tag{2.100}$$

where ζ is a constant. Rewriting this system in Laplace domain (with all the initial conditions zeroed), then:

$$s\theta = -\gamma\zeta^2\theta + \frac{\gamma\zeta}{s}. \tag{2.101}$$

For $\zeta \neq 0$, then, θ is given by:

$$\theta = \frac{1}{s}\frac{\gamma\zeta}{s + \gamma\zeta^2} = \frac{1}{s}\frac{\frac{1}{\zeta}}{\frac{s}{\gamma\zeta^2} + 1} = \frac{1}{s}\frac{1}{1 + \tau s}\frac{1}{\zeta}, \tag{2.102}$$

where the time constant is defined as $\tau = 1/\gamma\zeta^2$. The response in time domain is given by:

$$\theta(t) = \frac{1}{\zeta} - \frac{\tau}{\zeta}e^{-\frac{1}{\tau}t}, \quad t > 0, \tag{2.103}$$

which is a limited function. Hence, we state that, if ζ is a constant, $\theta \in \mathcal{L}_\infty$.

Now, let us investigate a more *intuitive interpretation* for this problem. To do so, let us rewrite the parametric update law in 2.98:

$$\dot{\theta} = -\gamma\zeta^2\theta + \gamma\zeta(t), \tag{2.104}$$

where ζ is *interpreted* as a constant (unit step) in the second term ($\zeta(t) = \zeta\mathrm{u(t)}$) and as a generic limited function in the third term ($\zeta(t)$). Rewriting this system in Laplace domain (with all the initial conditions zeroed), then:

$$s\theta = -\gamma\zeta^2\theta + \gamma\zeta. \tag{2.105}$$

For $\zeta \neq 0$, then, θ is given by:

$$\theta = \frac{\gamma\zeta}{s + \gamma\zeta^2} = \frac{\frac{1}{\zeta}}{\frac{s}{\gamma\zeta^2} + 1} = \frac{1}{\tau s + 1}\frac{1}{\zeta}. \tag{2.106}$$

Notice though that θ can be *interpreted* as the output of a linear filter with time constant τ, in which θ exponentially tends to the filter input is $1/\zeta$. Hence, the signal θ is called to be as the *filtered inverse* of ζ. Notice hence that, the smaller the values of ζ and γ are, the slower the filtering process is. According to Vargas (2013), this means that: (i) the filter updating rate can be controlled through a suitable choice of γ; (ii) when ζ is near zero, high values of $\theta \approx 1/\zeta$ are generated, but the filter is stable and slower (since τ is high).

Vargas (2013) also performs the system analysis for the case of a time-varying limited function ($\dot{\zeta}(t) \neq 0$), in which we state the following properties:

(i) $\zeta \equiv 0 \Rightarrow \theta(t) = \theta(0)$ (i.e., θ is not assigned to an infinity value if ζ is a constant initially set as zero.

(ii) ζ constant $\Rightarrow \theta \in \mathcal{L}_\infty$, as seen in equation 2.103.

(iii) $\zeta \in \mathcal{L}_\infty \Rightarrow \dot{\theta} \in \mathcal{L}_\infty$.

(iv) $\theta(t)$ does not have finite escape time.

Hence, the Filtered Inverse (*FI*) method main concept is to compute dynamically the inverse of a function ζ though an adaptable filter, which is easily tuned only by a single gain γ. Notice though that the necessary condition for the application of this technique is that ζ cannot converge to zero at a rate slower than an exponential function.

2.8.2 Main concepts: MIMO case

The application of FI can be extended to Multiple-Input Multiple-Output (MIMO) systems described by matrices (Vargas 2013). To clarify, consider the following first-order MIMO system:

$$\dot{y} = Z(t)u, \tag{2.107}$$

where $u \in \mathbb{R}^n$ is a limited system input or control signal, $y \in \mathbb{R}^m$ is the system output, and $Z(t) \in \mathbb{R}^{m \times n}$ is a non-linear time-varying matrix function, being $m \leq n$. Assume that the control goal is to track a reference trajectory $y_d(t)$ in order to guarantee the asymptotic convergence of the tracking error $e := y_d - y$ to zero. A control law $u(t)$ which linearizes the control system in (2.107) and ensures the achievement of the control goal is given by:

$$u = Z^\dagger(t)\nu, \quad \nu = \dot{y}_d + Ke, \tag{2.108}$$

where $K = K^T > 0$ is a positive-definite matrix, and $Z^\dagger(t) \in \mathbb{R}^{n \times m}$ is the left pseudo-inverse of $Z(t)$. Analogously to the scalar case, we want to estimate the pseudo-inverse by a matrix Θ dynamically updated. Yet, notice that, in the matrix case, the error signal dynamic must be suitably defined considering the non-linear *right-error*:

$$S_r = Z\Theta - I \quad \in \mathbb{R}^{m\times m}, \tag{2.109}$$

and the *left-error*:

$$S_l = \Theta Z - I \quad \in \mathbb{R}^{n\times n}. \tag{2.110}$$

The matrix parametric update laws are defined by Vargas (2013) as:

$$\dot{\Theta} = -\Gamma Z^T S_r, \text{ for the right error} \tag{2.111}$$

$$\dot{\Theta} = -\Gamma S_l K^T, \text{ for the left error.} \tag{2.112}$$

In non-redundant ($m = n$) and full-rank cases, the update laws described in 2.111 and 2.112 are capable of solve the inverse problem individually, since the left and right pseudo-inverse matrices of Z are the same. This is not applied when $m \neq n$ (full-row-rank cases and full-column-rank cases), since the left and right pseudo-inverses matrices differ from each other. In addition, the error signals $S_r \in \mathbb{R}^{m\times m}$ and $S_l \in \mathbb{R}^{n\times n}$) are of different dimensions, which implies in different properties for each other. Then, a composite parametric updating law that avails simultaneously the two error matrices is proposed as follows:

$$\dot{\Theta} = -\Gamma(Z^T S_r + S_l Z^T). \tag{2.113}$$

The proof of the algorithm stability and convergence is better detailed by Vargas (2013).

2.8.3 Application to manipulator control

As seen in Sections 2.4.3 and 2.6.2.1, the linear mapping between the joint velocities and the end-effector velocities ($\dot{x} = J\dot{q}$) is given by the Jacobian matrix, which can be applied to found the robot inverse kinematics via iterative approach or to perform the kinematic control. Without loss of generality, the choice of:

$$\dot{q} = J^{-1}\nu, \tag{2.114}$$

being $\nu = \dot{x}_d + Ke$ and $K = K^T > 0$, ensures that $e \to 0$. For redundant manipulators $m < n$, the Jacobian right pseudo-inverse $J^\dagger$ is used instead for a least-norm or optimal solution. However, the Cartesian signal ν should not take the arm to singular configurations, where J loses rank and J^{-1} $(m = n)$ or $(JJ^T)^{-1}$ $(m < n)$ are not defined. Even near singularities, the Jacobian matrix is bad conditioned with small pivots (and large inverses), hence leading to undesired high velocities at joint level. This problem is sought to be solved by the use of FI, which proposes that the Jacobian inverse/pseudo-inverse is computed dynamically in Θ. Thereby, the inverse kinematics or control law is now given by:

$$\dot{q} = \Theta\nu, \tag{2.115}$$

where $\Theta \in \mathbb{R}^{n\times m}$ is the *Jacobian filtered inverse/pseudo-inverse*, and it is online updated via:

$$\dot{\Theta} = -\Gamma(J^T S_r + S_l J^T) = -\gamma(J^T J\Theta + \Theta JJ^T - 2J^T), \tag{2.116}$$

where $\Gamma = \gamma I > 0$ is the update gain, S_r and S_l are the error matrices defined in (2.111) and (2.112), respectively. Table 2.3 summarizes the algorithm equations in both continuous domain and in its discrete matching.

Continuous	Discrete (period T)
$e(t) = x_d(t) - x(t)$	$e[n] = x_d[n] - x[n]$
$\nu(t) = \dot{x}_d(t) + Ke(t)$	$\nu[n] = \dot{x}_d[n] + Ke[n]$
$S_r(t) = J(t)\Theta(t) - I^{m\times m}$	$S_r[n] = J[n]\Theta[n] - I^{m\times m}$
$S_l(t) = \Theta(t)J(t) - I^{n\times n}$	$S_l[n] = \Theta[n]J[n] - I^{n\times n}$
$\Theta(t) = \Theta(0) + \int_0^t \dot{\Theta}dt$	$\Theta[n+1] = \Theta[n] + \Delta_\Theta[n]T$
$\dot{\Theta}(t) = -\gamma\left\{J^T(t)S_r(t) + S_l(t)J^T(t)\right\}$	$\Delta_\Theta[n] = -\gamma\left\{J^T[n]S_r[n] + S_l[n]J^T[n]\right\}$
$q(t) = q(0) + \int_0^t \dot{q}dt$	$q[n+1] = q[n] + \Delta_q[n]T$
$\dot{q}(t) = \Theta(t)\nu(t)$	$\Delta_q[n] = \Theta[n]\nu[n]$

Table 2.3: FIA equations in continuous and discrete domains

2.8.4 Modified control law

It is worth noticing that $1/\zeta$ (ζ inverse) is an *odd function*, but θ (filtered inverse) *is not.* To restore this property, Vargas (2013) proposes a modification in the way θ is computed, as follows:

$$\theta_M = \theta\zeta, \tag{2.117}$$

and, thus, the main property of odd functions is restored, i.e., $sign(\theta_M) = sign(\zeta)$. Analogously, for the matrix case, the following modification is proposed:

$$\Theta_M = \Theta\Theta^T Z^T, \tag{2.118}$$

where Z is the MIMO system matrix, and $\Theta\Theta^T$ is a semi-positive-definite and symmetric matrix. To illustrate this method, a simulation was performed aiming to find the filtered inverse Θ_M of a time-varying rotation matrix $R(t) = e^{\hat{\omega}\theta(t)}, \omega = 0.5774 \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$, $t \in [0, 4]$ seconds. Figure 2.18 shows the quaternion vectorial part ϵ_Q of the orientation error $Q = R\Theta_m$ with respect to time. Notice that the error quickly converges to zero. For the manipulator inverse kinematics and control

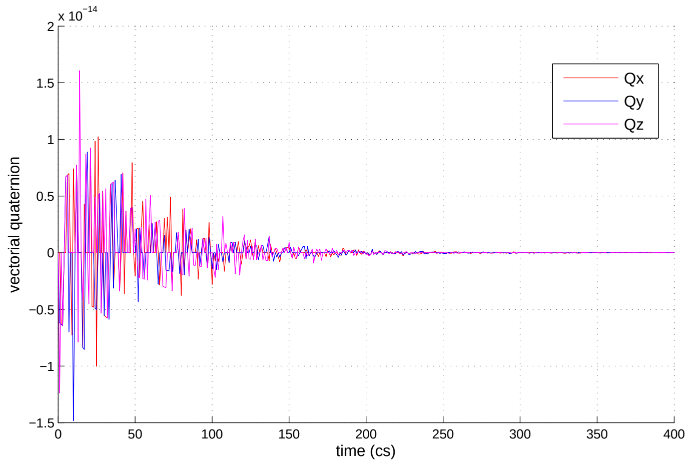


Figure 2.18: Quaternion error of $R\Theta_M$ (vectorial part).

of a manipulator problem, the control signal is given by:

$$\dot{q}_M = \underbrace{\Theta\Theta^T J^T}_{\Theta_M} \nu, \tag{2.119}$$

and the relationship between the robot velocity $\dot{x}$ and the ν is given by:

$$\dot{x}_M = J\,\dot{q}_M = \underbrace{J\Theta\Theta^T J^T}_{M} \nu, \tag{2.120}$$

where $M = M^T \geq 0$.

2.8.5 Augmented Jacobian

As discussed by Vargas (2013), FI provides a solution for inverse kinematics and control even over or near unfeasible and unreachable poses, which permits the robot to be controlled straight over singular configurations. This method can be also applied to satisfy an *additional system constraint* given by the scalar objective function

$f(q) = 0$, in which $q \in \mathbb{R}^n$ is the joint space. Then, the control goals are extended and described by:

$$\begin{bmatrix} x \\ f \end{bmatrix} \rightarrow \begin{bmatrix} x_d \\ 0 \end{bmatrix}. \tag{2.121}$$

By differentiating the previous equation with respect to time, we get:

$$\begin{bmatrix} \dot{x} \\ \dot{f} \end{bmatrix} \rightarrow \underbrace{\begin{bmatrix} J(q) \\ J_f(q) \end{bmatrix}}_{J_+(q)} \dot{q}, \tag{2.122}$$

where $J_f(q) = \partial f / \partial q \in \mathbb{R}^{1 \times n}$, and $J_+(q) \in \mathbb{R}^{(m+1) \times n}$ is formalized as the *augmented Jacobian*. Thus, the control signal defined in 2.119 is now given by:

$$\dot{q}_+ = \underbrace{\Theta_+ \Theta_+^T J_+^T}_{\Theta_{M+}} \nu_+, \tag{2.123}$$

where:

$$\nu_+ = \begin{bmatrix} \nu \\ \nu_f \end{bmatrix} = \begin{bmatrix} \dot{x}_d + Ke \\ -k_f f \end{bmatrix}, \tag{2.124}$$

where Θ_+ is J_+ filtered inverse, $e = \dot{x}_d - x$, $K = K^T > 0$ and $k_f > 0$ is a positive scalar gain. Notice that our goal is to lead this system to $\dot{e} + Ke = 0$ and $\dot{f} + k_f \cdot f = 0$. According to Vargas (2013), it is worth setting $k_f = 1$ and including the system weighing in the computation of the function $f(q)$, since this would lead to a Jacobian J_f with small or null elements in cases in which the constraint is of low priority or does not exist.

The block diagram with the complete manipulator control using the filtered inverse, modified control law and augmented Jacobian is found in Figure 2.19.

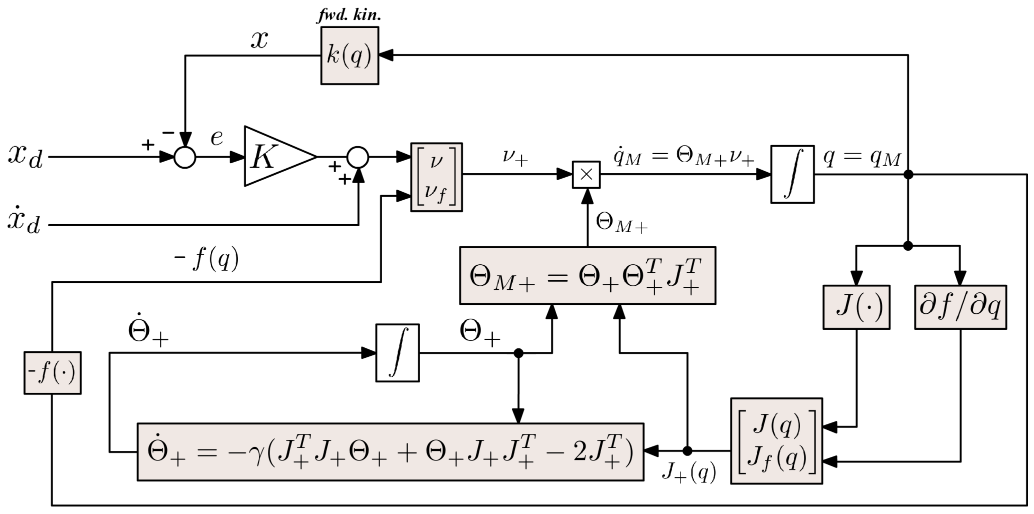
2.8.6 Objective function f

Vargas (2013) proposes two examples of objective functions to be applied in robotics.

2.8.6.1 Joint limits

One of the most common control objectives is to keep the joint angles within its mechanical limits (or even another established limits). We define $[\bar{q}_i - \delta_i, \bar{q}_i + \delta_i], \delta_i > 0$ as the interval in which the i^{th} joint should remain. The objective function is defined by the sum of the normalized joint deviations $\xi_i = \left(\frac{q_i - \bar{q}_i}{\delta_i}\right)$, as follows:

$$f(q_1, q_2, \cdots, q_n) = \sum_{i=1}^{n} \alpha_i \, \xi_i^{2k_i}. \tag{2.125}$$



fwd. kin.
$k(q)$
x
x_d
$\dot{x}_d$
e
K
+
−
+
+
$\begin{bmatrix}\nu\\ \nu_f\end{bmatrix}$
ν_+
$\dot{q}_M = \Theta_{M+}\nu_+$
$q = q_M$
Θ_{M+}
$\Theta_{M+} = \Theta_+\Theta_+^T J_+^T$
$-f(q)$
$-f(\cdot)$
$\dot{\Theta}_+$
Θ_+
$J(\cdot)$
$\partial f/\partial q$
$\begin{bmatrix}J(q)\\ J_f(q)\end{bmatrix}$
$J_+(q)$
$\dot{\Theta}_+ = -\gamma(J_+^T J_+\Theta_+ + \Theta_+ J_+ J_+^T - 2J_+^T)$

Figure 2.19: FIA block diagram, with modified control law and augmented Jacobian.

2.8.6.2 Obstacle avoidance

Another common control objective is to maintain the robot structure away from known obstacles. One proposed objective function is a Gaussian function defined by the distances between arbitrary points p_i (located at the manipulator structure) and the center of a known obstacle μ. The obstacle center is the center of the Gaussian function, whose contour lines are defined by the $M_f = M_f^T = R_f D_f^{-1} R_f^T > 0$. Notice that R_f a matrix rotation that defines the ellipsoid axes, and D_f has positive elements at its diagonal, whose square roots are proportional to the stretching of the contour line in the respective axis direction. Thus, the objective function is given by:

$$f(p_1, p_2, \cdots, p_c) = \sum_{i=1}^{c} \alpha_i \, e^{-(p_i-\mu)^T M_f (p_i-\mu)}. \tag{2.126}$$

Notice that $f(\cdot)$ is a function of q, since the coordinates of the points p_i are given by a function $g(q)$ of the manipulator joints.

Chapter 3

Manipulator Design: Functionalities, Electromechanics and Modeling

This chapter presents the design of DORIS manipulator, including:

- An overview of DORIS robot, which is our case study.
- Project requirements for DORIS manipulator.
- Dissertation on the manipulator required functionalities.
- A study of the existing robotic arms in the market and literature, so as to investigate relationships between the robot link dimensions, supported payload, arm range and joint velocities.
- Details of the electromechanical design, including actuators, control drivers, sensors, employed materials and integration to DORIS electronics system, as well as their selection criteria.
- Kinematic modeling, with the assistance of the mathematical concepts and tools reviewed in Chapter 2.
- Suggestion of modifications of the original structure design, taking into account the analysis of the kinematic modeling.

3.1 DORIS robot

3.1.1 DORIS overview

As presented in Section 1.1.2, DORIS (Figure 3.1) consists of a set of independent mobile modules (or wagons) guided by a rail (Carvalho et al. 2013, Galassi et al. 2014). The purpose of the robot is to carry several sensing systems to perform monitoring, inspection, supervision and surveillance of equipment and personnel of offshore facility *topsides*. For this, a rail, specially developed by DORIS Mechanics team, is installed throughout the topside, passing near the areas and machines of interest. Thus, the rail designed layout is what defines the robot path and, therefore, which areas and equipment shall be monitored. Since the rail is designed to be as simple as possible (concerning material, construction procedures and cost), most of the robot complexity are limited to the mobile modules.

DORIS can operate autonomously or remotely controlled from a remote base, which should be installed at a suitable site on the facility topside. The main tasks that can be realized by DORIS are: collect images, video and audio of the environment, monitor hydrocarbon gas level, monitor vibration of rotating machinery, *real-time in loco* processing and transmission of collected data, and alarm trigging upon detected anomalies - such as abandoned objects, temperature/vibration/gas level outside normal levels, or non-authorized personnel - which imposes the operator to take decisions concerning the occurrence.

3.1.2 Traction system

The robot moves along the rail using a traction system specially designed for this application, which is composed of:

- A mechanical *gimbal* specially designed for the module motion in curves and vertical sections
- Four EC brushless motors with incremental encoder and ratio planetary gearhead (from *Maxon Motor*®, P/N 470625)
- Four power drivers with internal position, speed, acceleration and current control loop (see Section 3.2.5)

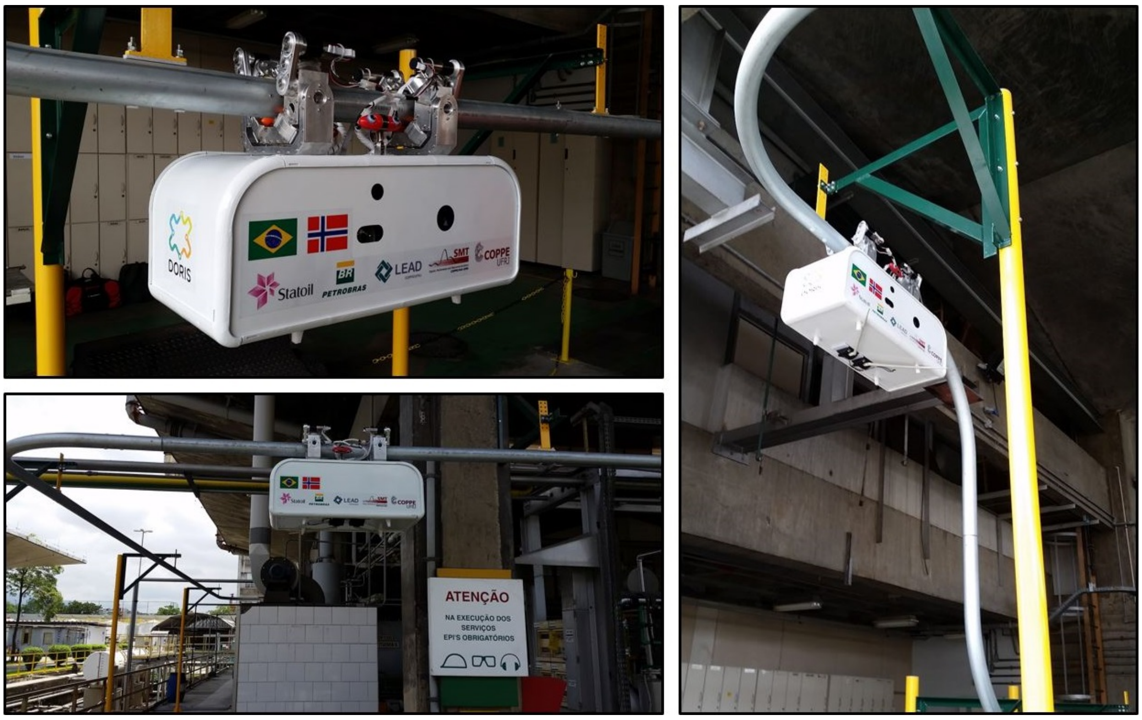


Figure 3.1: Pictures of DORIS operating in a test at an utility plant at Petrobras-CENPES.

3.1.3 Devices

Besides the traction system, each module is independent and can carry whatever devices the user selects for specific functionalities, which are:

Fixed devices:

- High performance computers
- Power system composed of four 33.6 VDC battery packs
- *Vehicle Support System* for battery management, power distribution, supervision and protection of the robot electronics itself and internal voltages/currents
- Wi-Fi *Access Point*, 2.4GHz radio (for emergency), and Ethernet Switch
- IMU (*Inertial Measurement Unit*) and high performance laser scanner to aid navigation and localization

Optional devices:

- Extra computers (to distribute control, task scheduling, localization, navigation and signal processing into two separate computers)

- High definition, infrared thermal and fisheye cameras, USB small stereoscopic webcams (check Section 3.3.2), microphones and lamps, hydrocarbon sensor
- Manipulator TETIS, with the following components fixed at its end-effector:
  - Vibration sensor with a coupled probe tip (check Section 3.3.1)
  - Force sensor (check Section 3.3.3)
  - USB stereoscopic webcam (check Section 3.3.2)
- DAQ - *Data acquisition board* (check Section 3.3.4)

3.1.4 Electric system

DORIS embedded electric system was designed to provide the hardware support for: (i) communication of all its peripheral devices (including the computer); (ii) power distribution to all devices, including the manipulator; (iii) self-monitoring and protection; (iv) communication of control devices (for both traction and manipulator system). It is organized into: (i) *primary electronics*, which comprises the overall communication networks; (ii) *secondary electronics*, represented by DORIS power system and the Vehicle Support System (VSS).

DORIS communications are organized into the following networks (Figure 3.2a):

- *Local Gigabit Ethernet Network*: responsible for priority and high bandwidth communications between DORIS main peripherals. It integrates DORIS computers, Ethernet cameras (thermal, fixed and fisheye), Wi-Fi access points, routers, VSS, and other Ethernet devices to be added in a star-network managed by a Gigabit Ethernet Switch (OSI-Layer 2).
- *Actuation system*: responsible for the actuation of DORIS motors. It is composed by the traction subsystem (see Section 3.1.2) and by the manipulator subsystem (see Section 3.3.4).
- *Controller Area Network* (CAN): responsible for real-time control of the actuation system via a robust network with strict error control managed by DORIS central computers (Corrigan 2008). In the case of DORIS robot, the CAN integrates the actuation system drivers, the PCs, and the manipulator force sensor (see Section 3.3.3) in a bus-network.

- *Data acquisition system* (DAQ): responsible for digitizing of analog sensor signals. In DORIS case, it is composed by the hydrocarbon sensor, the vibration sensor (see Section 3.3.1, and an USB data acquisition board.
- *Computer network*: composed by two PCIe/104 modules with embedded Intel® Core i7 and solid-state drive (SSD), which is a non-moving part solution for protection against vibration and mechanical impact, and heat sink. The main computer is dedicated for the overall communications and the robot mission control (Freitas 2016). The second computer is dedicated for *in loco* processing of heavy data amount, such as video, image and audio. The USB devices are directly connected to the computers, including the data acquisition board, three Minoru stereo cameras with embedded microphone (being one use for the manipulator teleoperation, see Section 3.3.2), and the inertial unit (IMU).
- *Wireless technologies*: responsible for the communication between DORIS robot and the remote operation base located at the facility topside. The Wi-Fi communication is employed as primary link for the robot mission control (Freitas 2016) and for the video/audio/sensor data transmission to the remote base, whereas the 2.4GHz Radio Frequency communication is employed for emergency commands and remote robot wake/shutdown.

DORIS power system is divided in two buses: (i) *Motor Power Bus*, for supplying of the power drivers that compose the actuation system; (ii) *Electronics Power Bus*, for supplying of the remaining electronic devices and components. The strategy of galvanically isolate both power buses aims to: (i) avoid transmission of noisy interference from the motors to the remaining electronics, which could affect the computer normal operation, as observed in previous laboratory tests; (ii) better manage the internal power distribution, since the actuation system consumption rate is expected to be different from the remaining electronics.

The Vehicle Support System (VSS) consists a set of four dedicated printed circuit boards (PCBs, see Figure 3.2b), which are described below:

- PCB1 - *Battery Management System* (BMS): physical connection of the four battery packs, communication and interpretation of battery telemetry (such as state of charge, voltage, current and temperature), command of PCB3, radio communication, robot startup/shutdown, and on/off buttons.

- PCB2 - *Supervisory System*: monitoring of tension level and supply currents of all the robot devices, monitoring of the internal temperature and humidity, automatic protection against overcurrent, power saving by turning off devices individually via solid-state relays, communication between VSS and the Ethernet network.

- PCB3 - *Power Bus Switching System*: contains the high-power solid-state relays that are commanded by the BMS board, which are used to set each battery to a desired power bus.

- PCB4 - *DC-DC converters*: contains three DC-DC converters that provide the required tension levels (5, 12 and 24VDC) for DORIS devices.

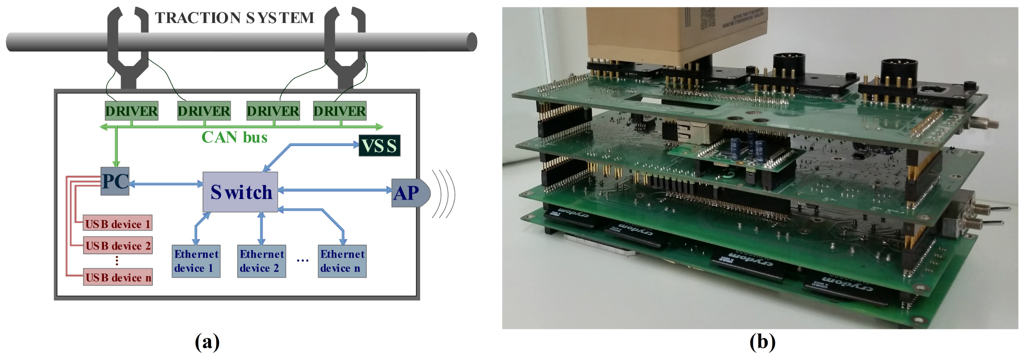


Figure 3.2: DORIS electric system architecture: (a) communications; (b) Vehicle Support System.

3.2 Manipulator electromechanical design

The first procedure for the design of a manipulator is deeply understanding its required tasks, the project constraints and possible extra control objectives. As mentioned in Section 1.1.3, DORIS manipulator (TETIS) has the account of extending the robot sensing range for tasks that require interaction with the environment.

3.2.1 Initial design constraints

Before TETIS project was started, the following design limitations had to be taken into account:

- It should have a lightweight structure, which should not exceed $5kg$, as defined by the Mechanical project of DORIS system.

- Its links and joints should be modular to permit the expansion/modification of the manipulator structure, case needed for improvements or changes of the goal tasks.
- Although lightweight, TETIS should support a maximum of $300g$ *payload*[1]. It should carry, at least, the vibration sensor ($51.03g$ plus $30g$ of the probe tip), the force sensor ($23g$) and the camera (estimated in $100g$) weights added. The combined weight of these coupled devices should not exceed $200g$. A gap weight was inserted over this amount, if we consider secondary elements - such as cables, connectors, screws and general uncertainties - or, in future, more/other coupled devices.
- It should perform vibration inspection, interaction with *touchscreen* surfaces and free camera drive. For this work, we consider that the surfaces over which the manipulator will interact with are always flat planes, since: (i) *touch-screens* are flat panels; (ii) many industrial rotating machines are protected by parallelepiped housing; (iii) if we perform sufficient small movements over surface, we can consider it to be locally flat.
- The rail layout should be designed so that the *touchscreens* and machines of interest are inside the manipulator dexterous workspace (see Section 2.3).
- It should remain at retracted pose while idle so as not to disturb DORIS normal movement, which could possibly collide with nearby equipment or alter the robot center of gravity. According to mechanical recommendations, the retracted position should be such that the arm is below DORIS module.
- While TETIS is being operated, the DORIS travelling along the rail is not allowed. Yet, small DORIS movements should be permitted, since this motion is understood as a prismatic joint that adds an extra DoF to the system. Thus, it is worth highlighting that the rail sections must be always straight near equipment of interest, otherwise, a prismatic movement is not possible (Figure 3.3).
- TETIS should not move while DORIS is travelling along the rail, so as to avoid collisions and save energy. If the joints are not equipped with self-lock mechanism, it is worth mounting a mechanical hook at DORIS wagon so that the manipulator is released over it when idle.

[1]Weight that the manipulator can load, discounting its own weight

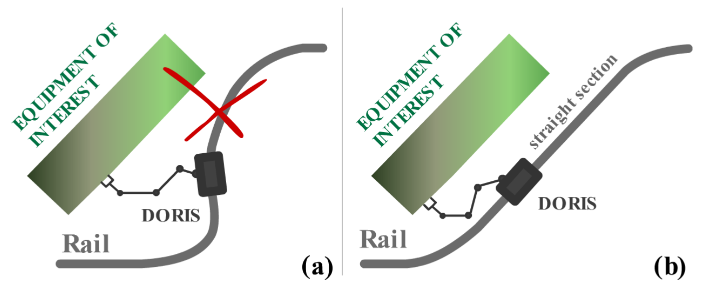


Figure 3.3: Prismatic movement near equipment of interest: (a) not allowed layout; (b) allowed layout (straight rail near equipment).

- The system must ensure its own integrity and the integrity of the equipment to be touched. Hence, the contact with nearby surfaces should be smooth, and the manipulator structure cannot collide with any nearby equipment or platform structure under any circumstances.
- Robot and environment *structuring*[2]:
  - Repeated DORIS operations is of great concern, since it could lead to accumulated error of the robot pose along the rail. A localization system based in particle filter and environment mapping by laser scanning (Carvalho 2016) is being developed and tested. In this work, we consider that this localization system guarantees a reliable repeatability such that the robot pose along the rail is perfectly known, as well as the geometry of nearby surfaces and obstacles to avoid.
  - The platform environment and the manipulator structure by itself are also complex and need to be modeled as perfect as possible. Likewise, successive DORIS operations may accumulate parametric uncertainties, and hence, generate position and orientation errors, which are critical for the robustness of the control system. This is a critical problem, since trajectory planning are usually performed in a structured environment, and hence, the system uncertainties need to be compensated to ensure the proper task fulfillment. As examples of solutions, Leite et al. (2012) propose a remote calibration method for path replanning of subsea robots, and Meggiolaro et al. (2000) suggests the utilization of force measure-

[2]In this work, we define a robot or an environment as *structured* if the spatial position of all its points of interest are known at all time instants ($t \geq 0$). In some cases, future positions are not known, but they can be satisfactory foreseen. In this case, the environment is also considered to be *structured*.

ments to aid the calibration of the own manipulator parameters. In this work, both platform and TETIS structure are considered to be known and invariable with respect to time.

3.2.2 Modeling of mechanical structure

Given the mechanical requirements described in the previous section, a preliminary structure was proposed for TETIS (Figure 3.4). By analysing it, we can highlight:

- It is composed of 5-DoFs, which are exactly the needed for TETIS tasks (3 in position and 2 in orientation).
- The presence of a prismatic DoF dispenses one revolute joint, which makes the manipulator net weight and size smaller.
- The 2nd joint and link ensure a 360° turn around the vertical direction and a safe distance between the manipulator 3rd link and DORIS module bottom.
- The 3rd and 4th joints are disposed in an *elbow configuration* ($\vec{z}_3 = \vec{z}_4$), which permits, at the same time, a satisfactory range extension and reduced occupied space at full retraction pose.

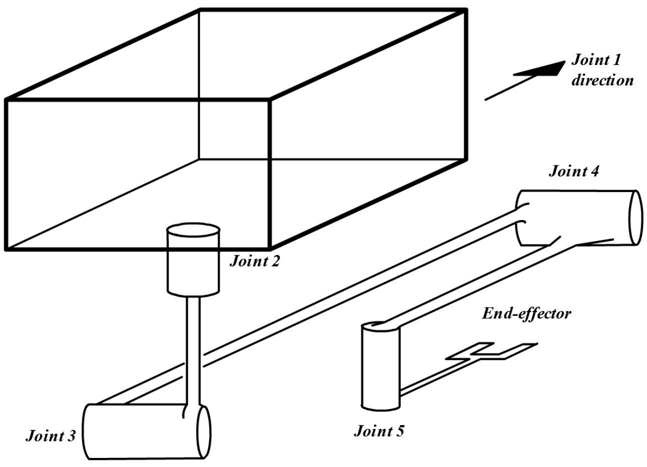


Figure 3.4: TETIS preliminary structure.

Figures 3.5, 3.6, 3.7a and 3.7b present TETIS from different points-of-view through 3D models designed in SolidWorks® software.

Figures 3.8a and 3.8b show the manipulator - still in construction phase - at fully retracted pose in different perspectives. Figure 3.9 shows TETIS with the camera and the force sensor assembled being integrated to DORIS electronics system.

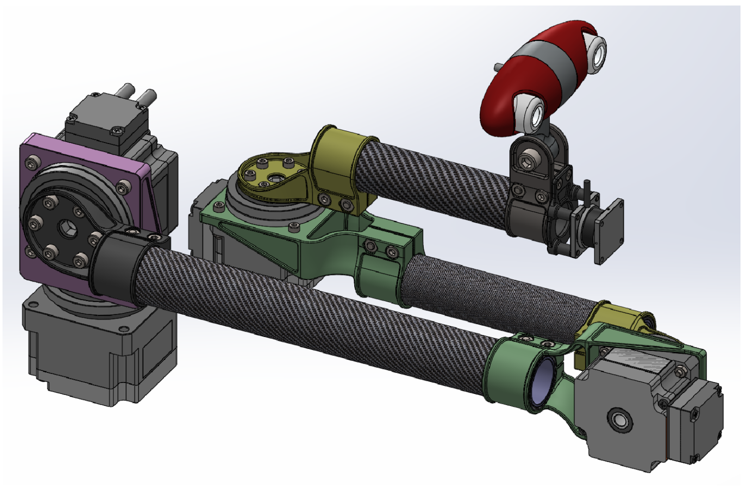


Figure 3.5: TETIS overview.

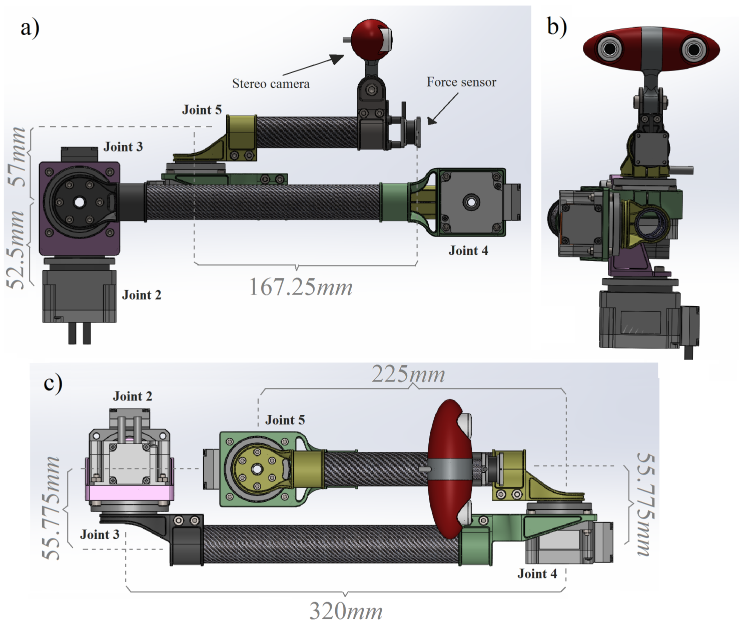
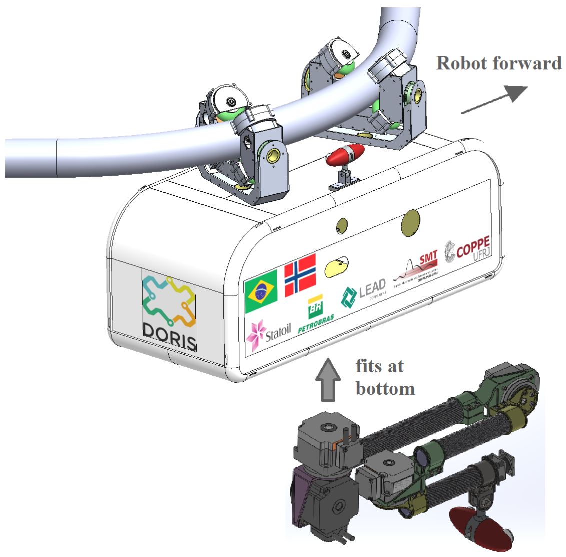


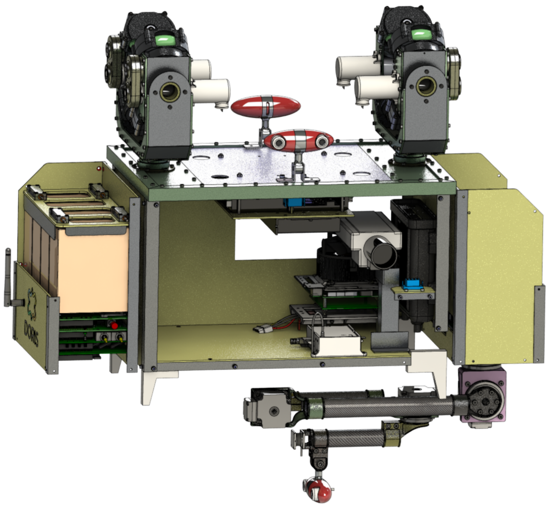
Figure 3.6: TETIS dimensions from different points-of-view: (a) from DORIS left side; (b) from DORIS front; (c) from DORIS bottom.

3.2.3 Study of commercially available manipulators

A market study was performed to investigate relationships between the main features of established commercially available manipulators. A total of 56 models were searched (Table 3.1, among those we can highlight KUKA, ABB, Adept, Comau, Fanuc, Kinova and Rethink Robotics manufacturers. The following features were analyzed: DoF, weight, rated/maximum payload for full stretched arm, approximated length of full stretched arm, volume of working envelope, pose repeatability, joint maximum speeds with rated payload, operation temperature, protection rate,

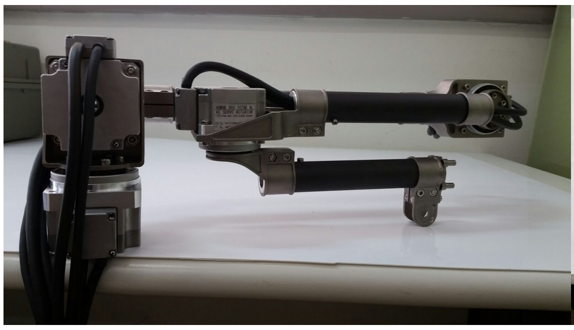


(a) TETIS at fully retracted pose

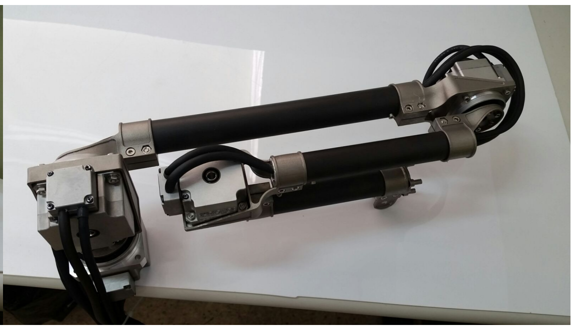


(b) TETIS and DORIS version without housing

Figure 3.7: TETIS mounting at DORIS.



(a) Lateral view



(b) Perspective

Figure 3.8: Manipulator in construction phase - Retracted pose.

and link lengths.

3.2.3.1 Weight vs. payload

One important matter to understand about manipulators is the relationship between weight and supported payload. The manipulator net *weight* is considered to be the combined mass of its links, joints, actuators, sensors and cabling. The manipulator supported *payload* is the maximum weight it can support at its end-effector when fully stretched, not including its own weight. This means that the manipulator joints closed to the base must provide sufficient torque to support the robot weight and the maximum payload. In some manipulators *datasheets*, we can also found their *rated payload*, which is an average payload that it can support during a normal

Figure 3.9: Weights, maximum payloads and rated payloads.

operation. Generally, the robot payload should not exceed its rated payload for a long time.

Figure 3.10 shows the weight, maximum payload and rated payload of several commercial manipulators. Notice that, in all cases, the payloads are much smaller than the robot weight. Now, let us define W_m as the ratio between the robot weight

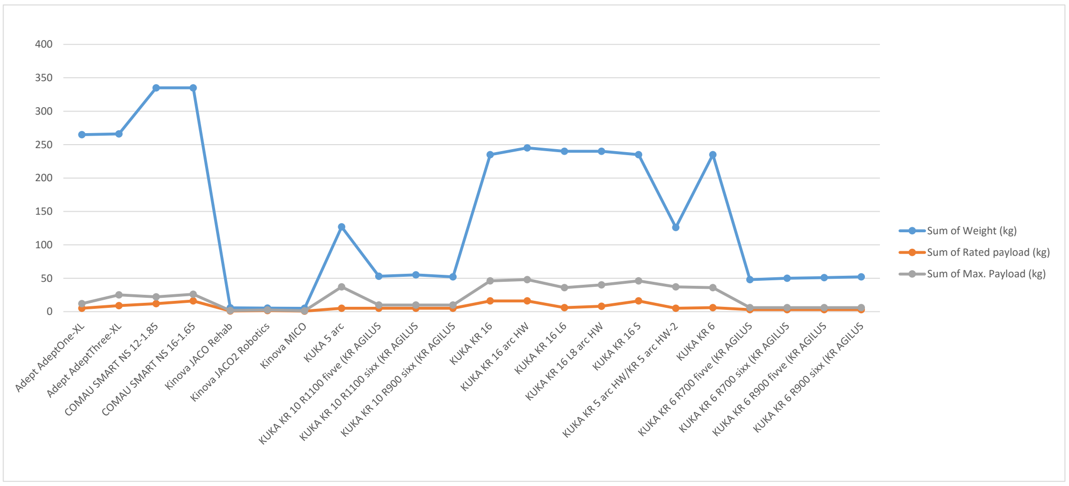


Figure 3.10: Weights, maximum payloads and rated payloads.

Table 3.1: Market search - Commercial manipulators

Manuf.	Model	DoF	Weight (kg)	Payload (kg)	Range (mm)	Repeat. (mm)	Op. temp. ($^\circ C$)	Prot.
ABB	IRB 4400	6	1040	60	3040	0.19	5 to 45	IP54
Adept	AdeptOne-XL	4	265	5	2414	0.04	5 to 50	IP20
Adept	AdeptThree-XL	4	266	9	2592	0.04	5 to 50	IP20
COMAU	SMART NS 12-1.85	6	335	12	2521	0.05	0 to 45	IP67
COMAU	SMART NS 16-1.65	6	335	16	2315	0.05	0 to 45	IP67
FANUC	ARC Mate 100iC	6	130	10	1957	0.08	0 to 45	N/A
FANUC	ARC Mate 100iC/12	6	130	12	1841	0.08	0 to 45	N/A
FANUC	ARC Mate 100iC/12S	6	130	12	1516	0.05	0 to 45	N/A
FANUC	ARC Mate 100iC/6L	6	135	6	1957	0.1	0 to 45	N/A
FANUC	ARC Mate 100iC/7L	6	135	7	2055	0.08	0 to 45	N/A
FANUC	ARC Mate 120iC	6	250	20	2304	0.08	0 to 45	N/A
FANUC	ARC Mate 120iC/12L	6	250	12	2503	0.08	0 to 45	N/A
FANUC	LR Mate 200iD	6	25	7	N/A	0.2	0 to 45	IP69K
FANUC	LR Mate 200iD/4S	6	20	4	N/A	0.2	0 to 45	IP69K
FANUC	LR Mate 200iD/4SH	5	19	4	N/A	0.2	0 to 45	IP69K
FANUC	LR Mate 200iD/7H	5	24	7	N/A	0.2	0 to 45	IP69K
FANUC	LR Mate 200iD/7L	6	27	7	N/A	0.3	0 to 45	IP69K
FANUC	M-10iA/10M	6	130	10	1844	0.08	0 to 45	N/A
FANUC	M-10iA/10MS	6	130	10	1522	0.08	0 to 45	N/A
FANUC	M-10iA/12	6	130	12	1841	0.08	0 to 45	N/A
FANUC	M-10iA/12S	6	130	12	1516	0.05	0 to 45	N/A
FANUC	M-10iA/7L	6	130	7	2055	0.08	0 to 45	N/A
FANUC	M-16iB/10L	6	230	10	2496	0.1	0 to 45	IP54
FANUC	M-16iB/20	6	220	20	2496	0.06	0 to 45	IP54
FANUC	M-20iA	6	250	20	2304	0.08	0 to 45	N/A
FANUC	M-20iA/12L	6	250	12	2503	0.08	0 to 45	N/A
FANUC	M-20iA/20M	6	250	20	2308	0.08	0 to 45	N/A
FANUC	M-20iA/35M	6	252	35	2308	0.08	0 to 45	N/A
FANUC	M-430iA/2F	5	55	2	1455	0.5	0 to 45	IP67
FANUC	Paint Mate 200iA	6	35	5	1175	0.02	0 to 45	Various
FANUC	Paint Mate 200iA/5L	6	37	5	1364	0.03	0 to 45	Various
Kinova	JACO Rehab	6	5.7	1	1232	N/A	0 to 30	IPX2
Kinova	JACO2	6	5.3	1.5	1201	N/A	-10 to 40	IPX2
Kinova	MICO	6	5	0.8	997	N/A	-10 to 40	IPX2
KUKA	5 arc	6	127	5	1783	0.04	10 to 55	IP54
KUKA	KR 10 R1100 fivve	5	53	5	1591	0.03	5 to 45	IP54
KUKA	KR 10 R1100 sixx	6	55	5	1595	0.03	5 to 45	IP54
KUKA	KR 10 R900 sixx	6	52	5	1391	0.03	5 to 45	IP54
KUKA	KR 16	6	235	16	2232	0.05	5 to 55	IP65
KUKA	KR 16 arc HW	6	245	16	2295	0.04	10 to 55	IP54
KUKA	KR 16 L6	6	240	6	2489	0.05	5 to 55	IP65
KUKA	KR 16 L8 arc HW	6	240	8	2677	0.04	10 to 55	IP54
KUKA	KR 16 S	6	235	16	2232	0.05	5 to 55	IP65
KUKA	KR 5 arc HW/KR 5 arc HW-2	6	126	5	1876	0.04	10 to 55	IP54
KUKA	KR 6	6	235	6	2189	0.05	5 to 55	IP65
KUKA	KR 6 R700 fivve	5	48	3	1196	0.03	5 to 45	IP54
KUKA	KR 6 R700 sixx	6	50	3	1196	0.03	5 to 45	IP54
KUKA	KR 6 R900 fivve	5	51	3	1391	0.03	5 to 45	IP54
KUKA	KR 6 R900 sixx	6	52	3	1391	0.03	5 to 45	IP54
KUKA	KR 60 JET KR C2 Robot	6	435	60	2002	0.15	N/A	N/A
KUKA	LBR iiwa 14 R820	7	29.9	14	1306	0.15	5 to 45	IP54
KUKA	LBR iiwa 7 R800	7	23.9	7	1266	0.1	5 to 45	IP54
KUKA	youBot 5-DoF	5	6.3	0.5	655	1	N/A	N/A
Rethink	Baxter (one arm)	7	37.4	2.2	1041	N/A	N/A	N/A
Rethink	Baxter Research Robot	7	37.4	2.2	1041	N/A	0 to 40	IP50
Rethink	Sawyer	7	19	4	1026	N/A	N/A	IP54

and the maximum payload, and W_r as the ratio between the weight and the rated payload. Figure 3.11 shows W_m and W_r, in which $mean(W_m) \approx 7.346, std(W_m) \approx 4.534$ and $mean(W_r) \approx 20.489, std(W_r) \approx 12.342$. Notice that the supported payload in lightweight robots - such as Kinova JACO, JACO2 and MICO - is almost equal to their own weights. Small values of W_m and W_r are desired, since they represent a good efficiency between the robot weight and their supported payload. This characteristic is commonly present in manipulators whose links are constructed with lightweight and resistant materials, which permits the employment of heavier and more powerful actuators. Hence, most part of the robot weight is concentrated in the actuators (that provide energy), not on the links (that do not provide energy, but support high loads if their material is resistant).

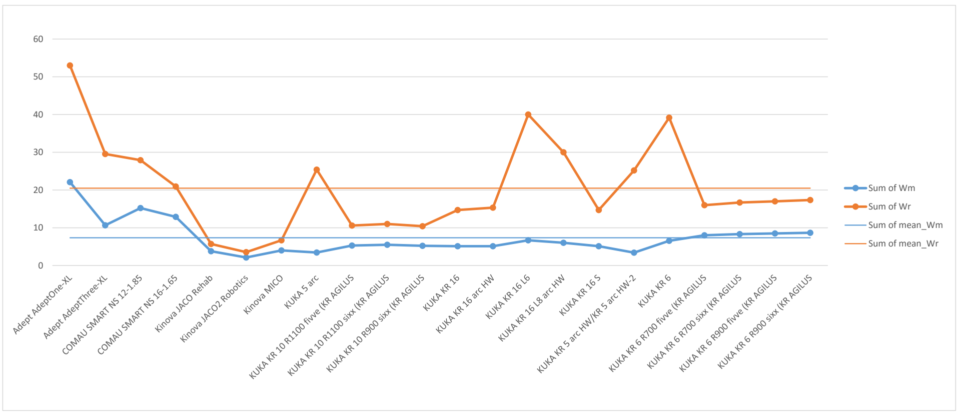


Figure 3.11: Ratios between weight and payloads.

3.2.3.2 Link length factor

The next matter to be investigated was to found if there exists a profile between the link lengths of the surveyed arms. First of all, is is worth stating that, in every surveyed model: (i) all its links are constructed of the same material; (ii) links diameter or cross sectional area were not considered. Then, to simplify the problem, we split the structure of each manipulator into three main *master links*. Technically, two links are considered to be part of the same master link if the rotation axis of the joint between them is collinear (or almost collinear) with both. Then, we defined the length proportional factors:

$$\gamma_{12} = \frac{L_2}{L_1}, \quad \gamma_{23} = \frac{L_3}{L_2}, \tag{3.1}$$

where L_i is the length of the master link i.

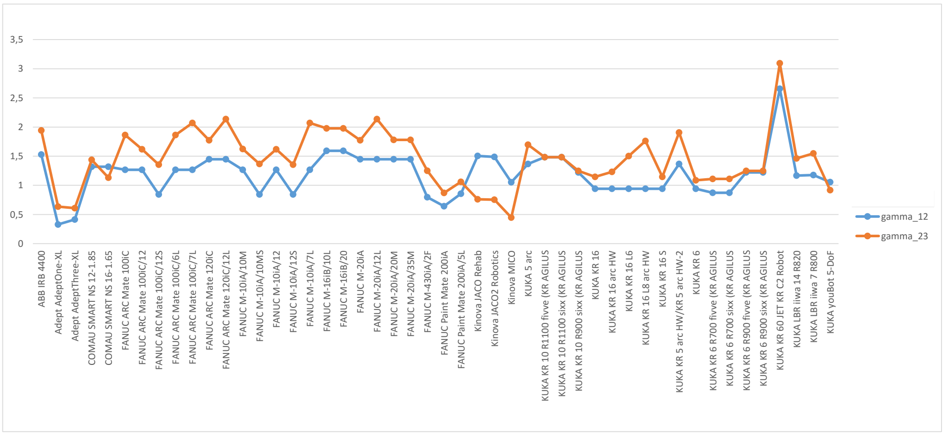


Figure 3.12: Proportional relationship between *master links* 1 and 2 (γ_{12}) and between 2 and 3 (γ_{23}).

Figure 3.12 shows γ_{12} and γ_{23} of several surveyed models. It is worth noticing that, in most cases, both γ proportionally factors follow the same profile. The relationship $\beta = \gamma_{23}/\gamma_{12}$ is found in Figure 3.13. Notice that $mean(\beta) \approx 1.264$, and $std(\beta) \approx 0.308$.

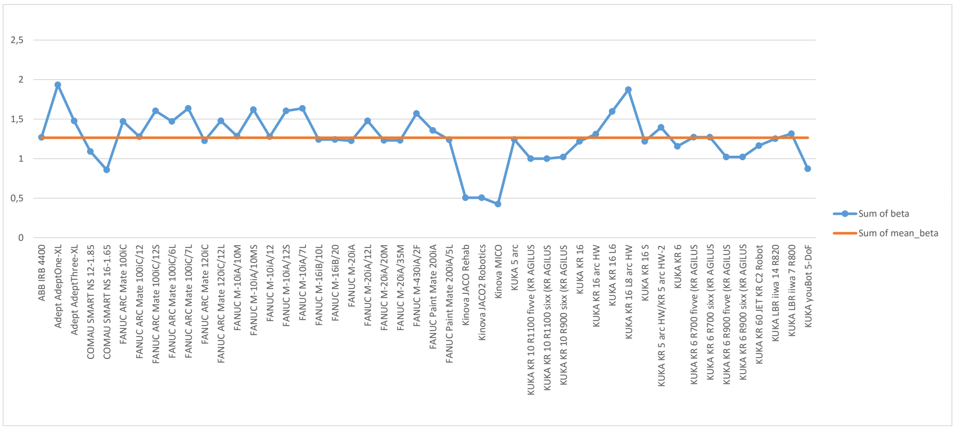


Figure 3.13: Relationship β between γ_{23} and γ_{12}.

Since we considered only link lengths but not their diameter or cross sectional area, this result is acceptable to understand that a proportional factor exists between the lengths of a serial chain manipulator. In general, the links next to the base (first links) are thicker and shorter than the links next to the end-effector (last links), since they are subjected to greater forces and torques, and hence, the first joints need to provide more torque.

In the situation of a fully stretched arm, the first links are subjected to both carried load and manipulator own weight, which is always greater than their supported payload, as shown in Figure 3.10. On the other hand, the last links must only support the carried load. Thus, longer and thinner links are permitted for them, since they are not be subjected to high torques and, consequently, do not break.

Hence, as a general geometry rule, the first links are short and thick in order to support the robot gross weight, whereas the last links are thin and long to provide a wide ranged workspace and not to add too much mass to the robot total weight (Figure 3.14).

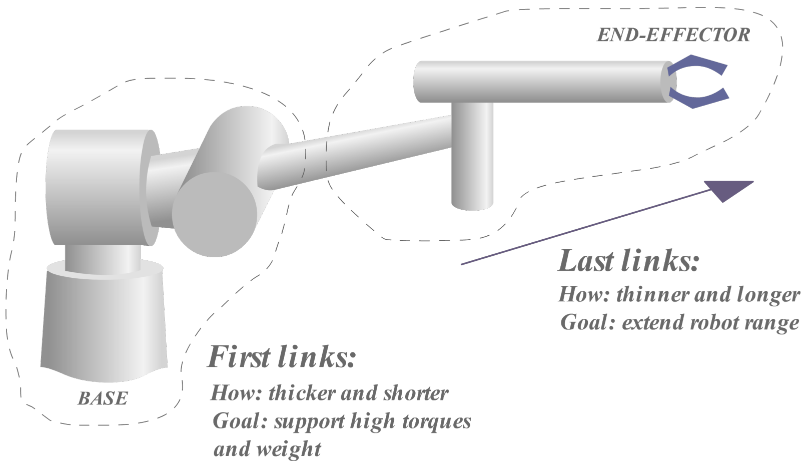


Figure 3.14: Geometry of first and last links.

3.2.3.3 Extended arm length vs. payload

Another investigated feature was the relationship between the length of the fully extended arm and its supported payload, which is shown in Figure 3.15.

Notice that these two curves follow a similar profile, which indicates that higher extended arm lengths are designed to support higher payloads. However, a common relationship between these two variables cannot be found, since it depends on how the manipulator structure is projected. For example, as discussed in 3.2.3.1, if we compare two manipulators A and B - where A is a lightweight arm whose links are constructed with soft but resistant materials, and B is a heavy industrial manipulator - then A is capable to provide the same payload as B with a longer extended range.

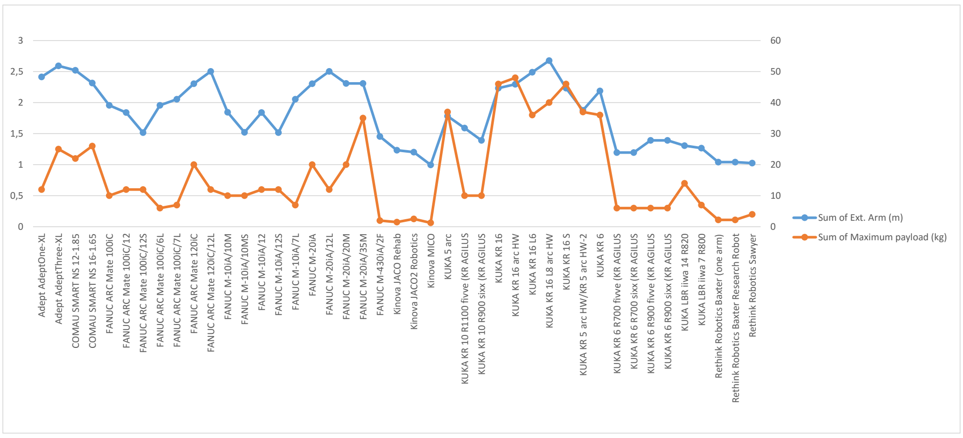


Figure 3.15: Relationship between extended arm length and payload.

3.2.3.4 Joint maximum velocities

Figure 3.16 shows the maximum velocities ω_i (in $^\circ/s$) of the i^{th} joint of several commercial manipulators. Notice that the joints next to the end-effector are capable to provide more speed than the joints next to the base. As discussed in 3.2.3.1, this occurs because the first joints are supposed to provide more torque rather then velocity. Generally, the goal of the first joints is to drive energy to support the weight of the own manipulator, whereas the last joints are supposed to provide enough velocity for the given tasks. Thus, as expected, a *trade-off* between torque/velocity is stated.

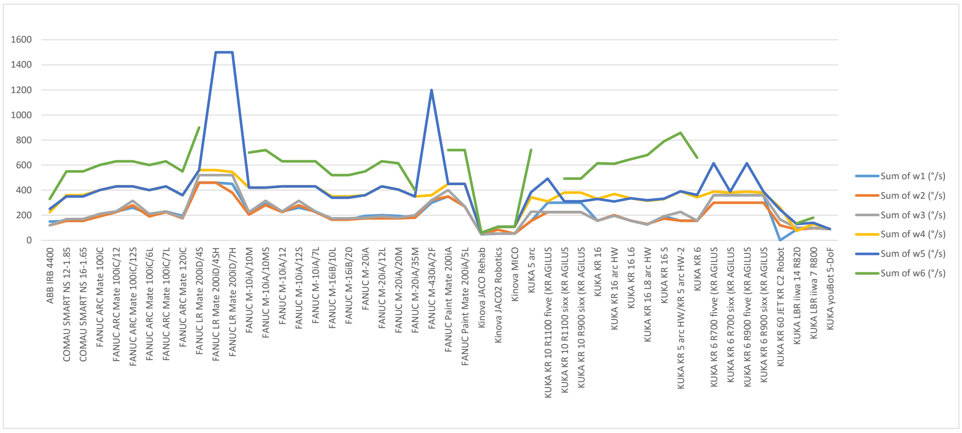


Figure 3.16: Joint maximum velocities

Observe that the maximum velocities are almost the same for all joints only in lightweight manipulators, such as from Kinova. Over again, this occurs because the links of these manipulators are made of lightweight and resistant materials, which allows the employment of powerful actuators to drive all joints. Notice further in Figure 3.17 that, in industrial anthropomorphic arms - such as those from FANUC - the velocity of the first three joints are almost the same.

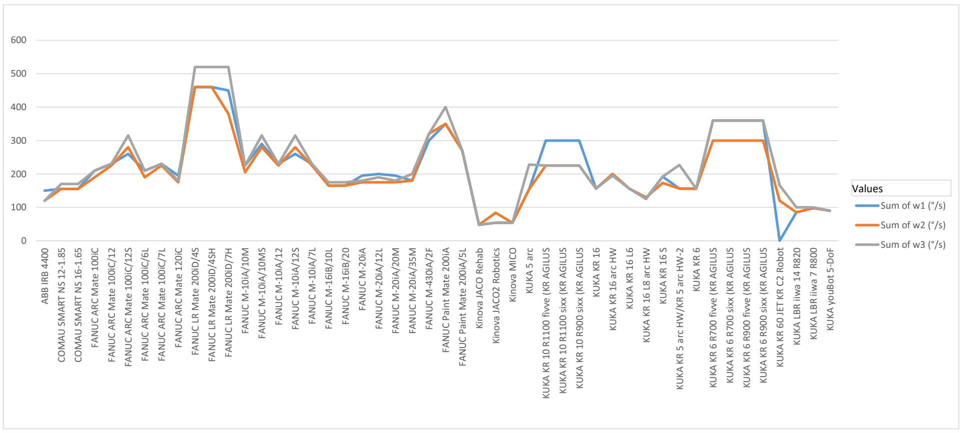


Figure 3.17: Maximum velocities of the three first joints.

3.2.4 Actuators

For a robust and reliable structure, TETIS revolute joints should be driven by very specific actuators with the following properties:

- High gear ratio, to make possible the control at the joint velocity level;
- Negligible backslash, to avoid non-linearities to be included in the system;
- Hollow shaft, to allow the routing of cables inside it, hence making the structure more compact;
- High accuracy and repeatability;
- Good *trade-off* between power capacity and compact structure.

Harmonic drives are strain wave gears, which are especial mechanisms that can improve several characteristics if compared to traditional transmission systems, such as helical and planetary gears. Their main advantages - such as no backslash, compactness, lightweight, high gear ratios (reconfigurable within a housing), excellent resolution/repeatability, coaxial input and output shafts, and high torque capability

- make then suitable for industrial applications that require high precision, such as robotics, aerospace, printing machines, motion control and automotive.

FHA Mini Servo Actuators from Harmonic Drive AG® (Figure 3.18a) are compatible with all the requirements above, and their commercial availability strongly motivated the start of DORIS manipulator design. Additionally, both models count with an incremental line driver encoder - located at the motor shaft before the reduction gear - and hall sensors for commutation of 3 windings. FHA-11C-100-D200-EKMI was selected to drive the first two joints each (i.e., joints 2 and 3, since joint 1 is prismatic), whereas FHA-8C-100-D200-EKMI was selected for the last two joints (4 and 5). The selection criteria was based on the calculation of the required torque for each joint, and it is described in details at A.1.

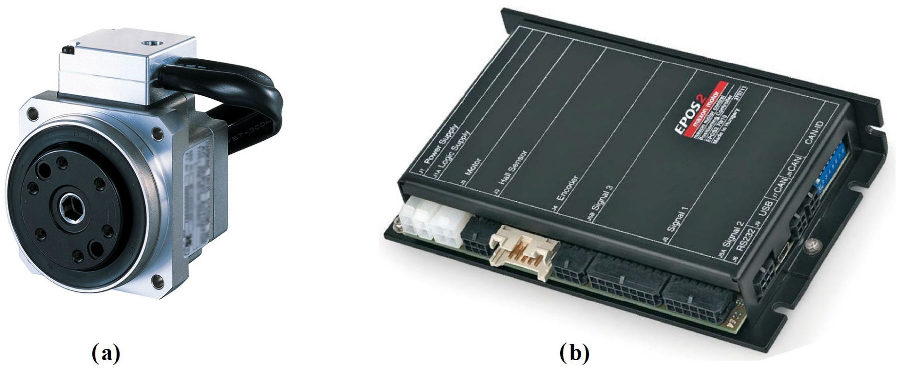


Figure 3.18: Actuators and drivers selected models: (a) Harmonic Drive AG® FHA Mini Servo Actuator; (b) Driver EPOS2 70/10 from Maxon Motor®.

3.2.5 Controller/power drivers

In order to achieve a satisfactory control at the joint level, the selected actuators demand a fine driver to be controlled. The good results from the utilization experience of Maxon Motor drivers in DORIS traction system, as well as their compatibility with CAN bus, motivate the use of the same driver model to actuate TETIS revolute joints. All of them are driven by a Maxon Motor EPOS2 70/10 P/N 375711 each (Figure 3.18b). For the proper selection of the driver model, some issues had to be taken into account, and they are explained in details at appendix A.2.

3.2.6 Retraction hook

Originally, once in retracted pose, the system should turn-off the manipulator control and, hence, stop driving power to the arm actuators. However, since the manipulator actuators do not have self-lock mechanism, it cannot be done, otherwise the manipulator structure would drop indefinitely downwards. One simple solution would be to keep controlling the joint actuators always to zero position and speed, even if the manipulator is not being operated. However, it would be extremely inefficient for energy consumption. A better solution was to fix on DORIS structure a special hook, so that the manipulator can be release over it and stay in an *nest* position for further calibration of the initial pose.

3.2.7 Materials

DORIS structure was mechanically designed to be made of the lightest and resistant possible materials. Its total weight was estimated in $3kg$, which stays within the permitted limit of $5kg$. However, as mentioned in Section 3.2.1, its parts were designed so as to permit expansion or change of the original structure.

3.2.7.1 Links

TETIS links are made of carbon fiber, which are enough lightweight and resistant for the given application. The links consist of hollow tubes that allow the passage of the manipulator cables (motor power, encoder signals, force sensor power and CAN, camera USB and vibration sensor power and signals) through inside them. Links 4 and 5 are tubes with $25.4mm$ diameter, while link 3 has $28.448in$ diameter. On the other hand, link 2 is not a tube, because it consists of the assembly of joints 2 and 3, whose diameter is approximately $60.1mm$ (according to the joint size).

3.2.7.2 Joints

TETIS joints (Figure 3.19) are made of 3D-printed Ti64 alloy (titanium, 6% aluminium, 4% vanadium). These joint parts were specially designed to be compatible with the designed link tubes and the selected motor models, and to permit cable routing through the holes of the motor shafts.

In previous project phases, the Mechanics team selected the aerospace aluminum alloy 7075, which weighs the same as the other aluminum alloys, but has a con-

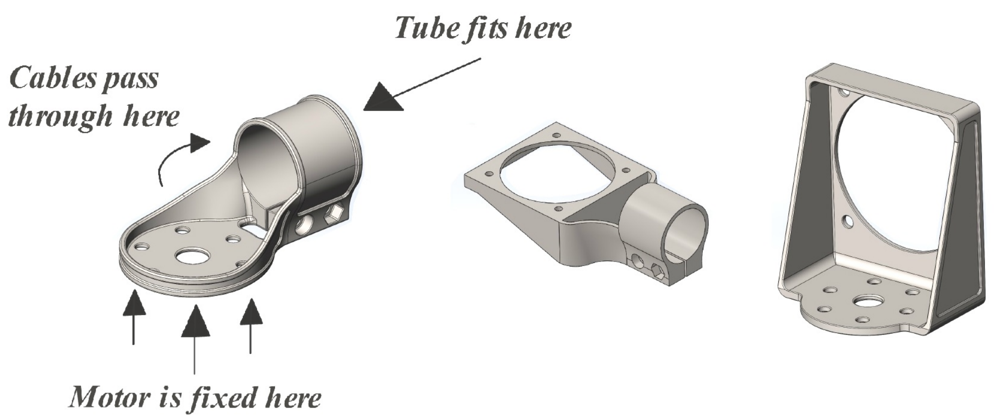


Figure 3.19: 3D illustration of TETIS joints.

siderably higher mechanical resistance (comparable to many steels), good fatigue strength, bigger Young Modulus, and average machinability. This material choice was changed later due to a good cost-effective opportunity that was offered to test titanium alloy 3D printing. Ti64 is denser than Al7075, but more resistant. The Mechanics team concluded later that the additional weight of Ti64 is negligible.

3.2.8 Analysis of TETIS case

TETIS is a lightweight structure, i.e., the results of same study performed in Section 3.2.3 applied to it are expected to be similar to those obtained for Kinova JACO, JACO2 and MICO. By considering the specifications defined in Sections 3.2.4, 3.2.5 and 3.2.7, TETIS main specifications are summarized at Table 3.2, where the 1st *masterlink* was considered to be the 3rd link, and so forth, since the 2nd link does not affect the calculation of payloads and extended arm range. Therefore, we define $\gamma_{12} = \frac{L_4}{L_3}$ and $\gamma_{23} = \frac{L_5}{L_4}$. As we can notice, we succeeded on defining both weight

Table 3.2: TETIS specifications

Weight (M)	Payload (m_p)	$W_m = \frac{M}{m_p}$	γ_{12}	γ_{23}	β	Ext. Arm	$\omega_{2,3,4,5}$
$2.5kg$	Payload $0.3kg$	8.6	0.7	0.7	1	$0.72m$	$360°/s$

and maximum payload compatible with TETIS requirements. We obtained a ratio $W = \frac{weight}{payload} \approx 8.6$, which is very close to the mean of the commercial models (7.346), as shown in Figure 3.11. Likewise, γ_{12}, $\gamma 23$ and β are satisfactory similar to the values found in the searched models, as can be checked in Section 3.2.3.2. It is interesting to observe that $\gamma_{12} = \gamma_{23}$, which indicates a preservation of the multiplying factor from the previous link to the next, and $\beta = 1$, whose value is closer to the ones found in lightweight manipulators (Kinova JACO, JACO2 and MICO). Finally, notice that the joint maximum velocities are the same for all joints, which occurs only for the surveyed lightweight models, as seen in Section 3.2.3.4.

At the current development and construction stage, some important TETIS specifications cannot be determined yet, such as its pose repeatability, exact weight (taking into account the final cable lengths), workspace volume, and exact mechanical joint limits.

3.3 Manipulator functionalities

The manipulator tasks demand the mounting of specific sensors or functionalities in the original structure, and each one requires a comprehensive study about the better existing application methods and commercial models. In the case of TETIS, vibration sensing, force sensing, a lightweight camera and a reliable network for real time control are needed.

3.3.1 Vibration inspection

On a platform topside, many constant *rotating machinery* deserve special attention, specially: compressors, generators, pumps, boilers, motors, gearboxes, and rolls. According to de Lima et al. (2013), their failure or malfunction is related to abnormal shaft unbalance (when the machine load weight is not equally distributed) or misalignment (when the rotor and the machine axes are shifted from their natural concentric positions). Unlike other equipment types, this fault lead to uncontrolled *shakes* that may cause an irreversible mechanical brake. This problem is specially concerned in compressors, where it is possible to occur a phenomenon called *surge*, which is a sudden change of compressed air consumption that causes an internal eddy/whirling. The surge, if not prognosticated, can seriously damage the equipment, which may lead to chain reaction that causes a partial plant halt (Mathas 2012), resulting in incalculable damage to the producing company.

3.3.1.1 Rotating machinery prognostic: vibration measuring

One established strategy for the prognostic of this problem is to introduce the monitoring and inspection of the vibration/shake profile of such equipment (de Lima et al. 2013, Emerson 2014). The criticalness of the rotating asset will determine the type of vibration measurement. Emerson (2014) disposes the equipment types into different ranking categories (see the pyramid in Figure 3.20), in which we can state the *critical* (turbines, generators, boiler feed pumps), *essential* (fans, pumps, motors, gearboxes, presses, rolls), *important*, *secondary* and *non-essential*. Critical, essential and some important equipment require *constant monitoring* due to their critical role at the plant process and to their expensiveness, while the others may require only *manual and sporadic inspection*.

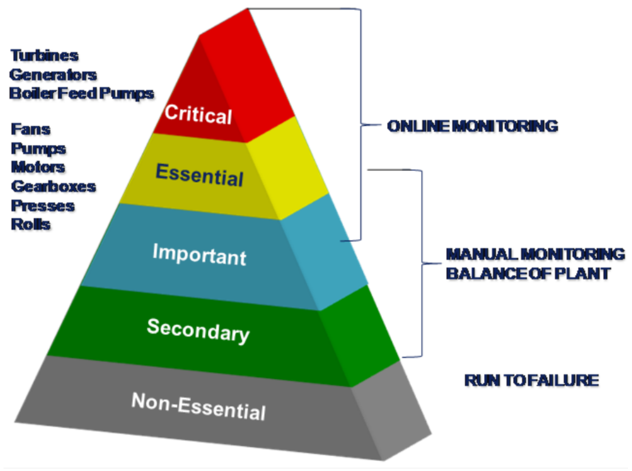


Figure 3.20: Asset pyramid.

3.3.1.2 Measuring variables to detect vibration

There are suitable interest variables for each required vibration monitoring task, such as the measuring of acceleration, velocity, displacement, rotation (speed/phase), pressure (static/dynamic), temperature (simple surface or infrared) and acoustic (Emerson 2014). For equipment with antifriction bearings - in which the mechanical components rotate in contact with each other and the shaft, undergoing stress and break down over time - the monitoring of acceleration, velocity or speed/phase is recommended. In equipment with fluid film sleeve bearings, the shaft rotates on a oil wedge and do not rotate along the bearing, hence, causing no wear out. Thus, the displacement measuring is also recommended in order to evaluate the relative position of the shaft and the bearing inner surface, hence ensuring a good oil wedge for a smooth shaft run.

3.3.1.3 Sensor principles

In addition, there are different measuring principles, which present a *relationship* between the vibration sensor *accuracy* and *intrusiveness.* For example, at low frequencies, *optical-based* vibration sensors (non-intrusive) are indicated, but at high frequencies, there are required more sophisticated and expensive optical mechanisms (Wilson 1999*a*,*b*,*c*). Nowadays, intrusive techniques are the most used in monitoring tasks that require high accuracy level in critical equipment. The most common intrusive vibration sensors are *accelerometers* based on many different principles, such as piezoelectric (shear, flexure, compression and charge modes), variable capacitance and servo force balance.

3.3.1.4 Mounting of intrusive sensors

In intrusive techniques, the sensor mounting over the equipment is a matter of concerning matter (Piezotronics 2016). Firstly, it is recommended the adequate surface preparation of the machine of interest. The surface, over where the accelerometer is to be attached, should be smooth, flat and free of foreign interfering particles of burrs. Sometimes, it is recommended the application of a thin layer silicon material between the mounting surface and the sensor base in order to improve high-frequency transmissibility.

The most common attachment procedures are (Figure 3.21): (i) *stud mounting*, for permanent installations; (ii) *screw mounting*, for twin-walled structures; (iii) *adhesive mount* (such as hot glue or wax), often used for temporary installations; (iv) *magnetic mounting*, for portable and temporary measurements; (v) *probe tips/handheld mounting*, useful when the other mounting techniques are impractical or the best location for installing the sensor is unknown of or is unsafely accessible. Probe tips are generally not recommended due to a variety of inconsistencies caused by manual misapplication - such as *bad-orientated mounting* (the best probe orientation should be *perpendicular* with respect to the machine surface) and amount of hand pressure - which create variables that affect the measurement accuracy, range and repeatability. Generally, this procedures is manually used only for low frequencies ($< 1000Hz$).

Constant machinery monitoring using intrusive sensors is a cost issue for producing companies, since it requires frequent maintenance (or even replacement/re-purchasing) of all the sensor network. One solution to reduce this cost would be to

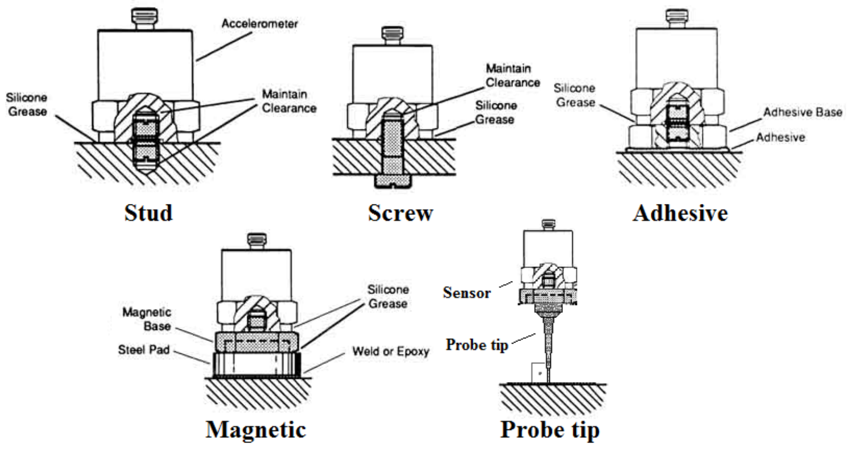


Figure 3.21: Intrusive mountings, from http://www.pcb.com/techsupport/tech_accel

perform sporadic inspection using just one (or few) sensor. In critical equipment, this may not be welcomed, since it would reduce the useful monitoring time, and, consequently, the prognostic accuracy, which could lead in the worst scenario to the mechanical breakage and to severe financial prejudice. Hence, it is worth investigating whether the constant monitoring or dealing with breakage hazard offers more cost-benefit. However, this solution would be feasible if: (i) the machine is not at the criticalness pyramid top (Figure 3.20) and do not require monitoring at multiple points over its surface; (ii) the machine is critical, but the cost of losing it or dealing with a plant halt caused by it is considered to be smaller than the maintenance of multiple sensors.

The sporadic inspection task can be realized by skilled operators (which can be a hazardous task if the machine is located at an harsh site) or by an autonomous or teleoperated robot, such as DORIS. The proposed solution for vibration inspection by DORIS (Galassi et al. 2014) is to couple an accelerometer with probe-tip at the end-effector of its manipulator, so that: (i) the robot range is extended; (ii) the robot can control the tip orientation to be correctly posed over the machine surface; (iii) the transmission of the machine vibration towards the robot may be damped through the manipulator structure; (iv) although not well recommended for accurate tasks, the probe tip is the most convenient mounting procedure for quick inspection. Thus, the vibration sensor selected model is an accurate intrusive accelerometer ICP® 623C00, from PCB® Piezotronics IMI Sensors (Figure 3.22), which can be used with the probe tips MH119-1A, MH119-2A or MH119-3A, according to the task and machine of interest.

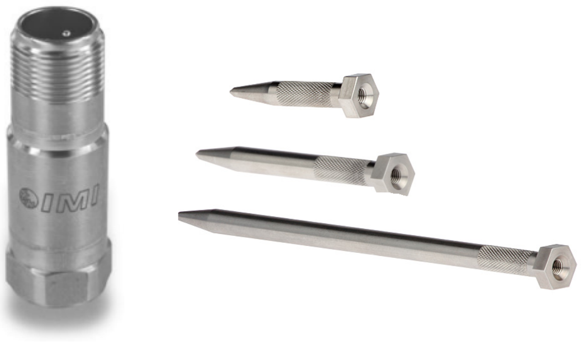


Figure 3.22: Vibration sensor 623C00 and probe-tips MH119 Series.

3.3.2 Stereoscopic camera

As aforementioned in this section, DORIS carries several camera types, such as high definition, thermal, fisheye and stereoscopic.

The high definition and the thermal cameras are heavy and spacious, though, they need to be mounted within the robot and at a fixed pose. This restricts the robot viewing area at a given point of the rail track, since the pose of the cameras cannot be changed until the robot is taken back to the operation base.

To circumvent this issue and offers the operator different sights/points-of-view from the robot at the same rail position, a proposed solution was to couple a lightweight camera at the manipulator end-effector, so that it can be freely moved throughout the arm workspace. The selected camera model is an stereoscopic 3D web-camera Minoru (Figure 3.23).

Figure 3.23: Minoru 3D webcam.

In further developments, the camera mounted at the robot end-effector can also be used for mapping of the platform unstructured environment, as already approached by Sujan & Meggiolaro (2005).

3.3.3 Force sensing

Force sensing and control concepts were covered in Section 2.7. As we can remind, one of the most important requirements for a proper manipulator design is its capacity to interact with the environment. In the specific case of TETIS, the tasks of touching a vibration sensor over nearby equipments and interacting with *touchscreens* requires the control of the contact force in order to: (i) maintain the integrity of both touched equipment and the manipulator itself, avoiding collisions and mechanical breakages; (ii) compensate design errors of position and orientation of nearby surfaces of interest. Parametric calibration using force measurements is also an important research topic, since it may be needed to compensate elastic effects due to contact forces/torques at the end-effector tip, even if the robot structure is considered rigid. Most of the calibration methods are based on complex or expensive strategies, such as using hooks to keep the arm in an nest pose (as mentioned in 3.2.6), and theodolites. But the utilization of wrist force/torque sensor information, as approached by Meggiolaro et al. (2000) and Lin & Lu (1997), may simplify the calibration procedure and improve its efficiency.

3.3.3.1 Sensing principles

There exist many types of force transducers with varying complexity of their sensing principles (Tekscan 2012, Siciliano & Khatib 2008), such as skin deflection, thermal, pressure, spring and optical. The factors that one must consider when specifying such a measurement include determining the proper output range, price, accuracy, size/weight, good signal conditioning electronics and ease of integration.

The most used manner to measure force is by measuring displacements. Basically, an object will change its shape upon the application of a force over it (e.g., bending of diving boards, coiling wires, spring compression, tilting a rubber, etc.). Materials can present different reactions according to how the force is applied, which are: (i) *tension*, when force pulls an object, increasing its length; (ii) *compression*, when force pushes against an object, shortening its length; (iii) *shear*, when equal and opposite forces are applied to the object opposing faces (Figure 3.24). The object shape changing degree is a function of the *strain* and *stress* on it. Strain is the relative change of shape/size of the object with respect to its original shape/size. Stress is the measure of the internal forces that acts within the object, causing hence the strain. In the real world (Figure 3.25), strain is proportional to stress

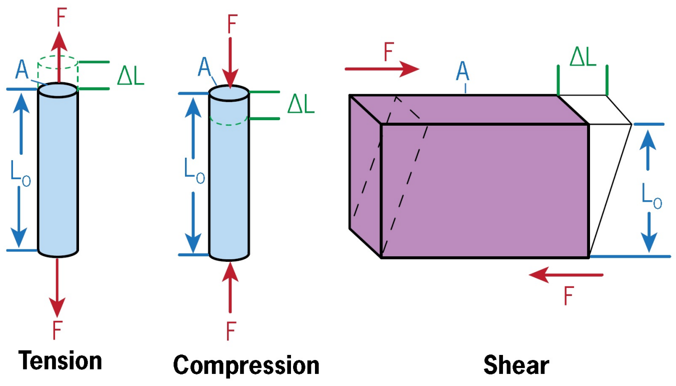


Figure 3.24: Force occurrence types.

(via Hooke's Law) when it is sufficiently small and, hence, do not exceed the *elastic limit* of the material. The elastic limit is the maximum stress that can be applied to a given object without the permanent change of its shape/size. It it is exceeded, the material enters the *plastic region*, and its shape changing is irreversible. If even more stress is applied, the material can reach the *breaking stress point*, which leads to its mechanical break. Every material has its own elastic modulus, limit and breaking stress. A proper force sensor has a sensing element in which: (i) upon a force applied to it, we can measure its strain; (ii) the elastic region is never left, otherwise the sensing element undergoes irreversible change, and the sensor gets uncalibrated/damage indefinitely. The most common method to measure strain is

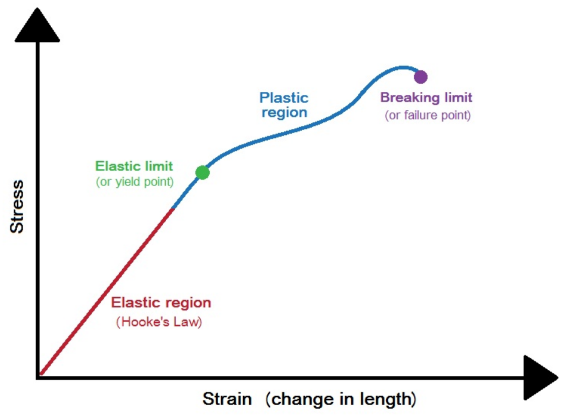


Figure 3.25: Stress × strain plot.

by using an electric resistance *strain gauge* (Figure 3.27a), which value is bonded to the strain of the elastic element. Its principle works as follows: a strain gauge (usually a metallic foil/wire or semiconductor) gets thinner and longer (under tension) or thicker and shorter (under compression), which changes its electrical resistance, since it depends on the conductor cross-sectional area, length and resistivity, i.e.,

$R = \rho \frac{d}{A}$ (Figure 3.26). Another relative newcomer method is the use of piezoresis-

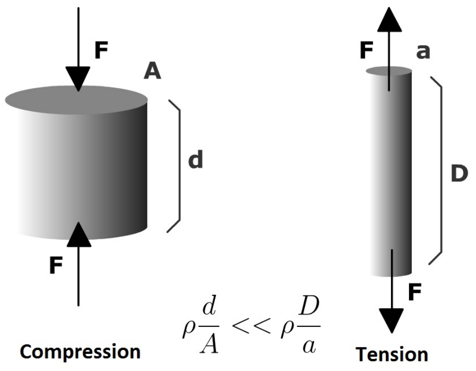


Figure 3.26: Change in resistance under stress.

tive crystals (Figure 3.27b) between two conductive plates. Piezoelectric elements are materials that generates an electric tension if strained, or, contrariwise, changes its shape/size if a voltage is applied to it. Basically, both element types are suitable for specific applications (HBM 2015), as summarized in Table 3.3. Recently,

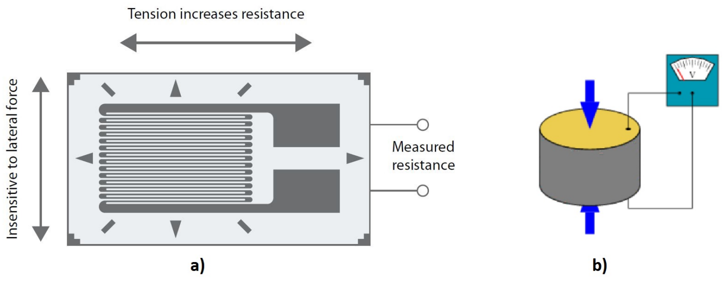


Figure 3.27: Schemes: (a) strain gauge; (b) piezoelectric crystal.

optic-based sensors (Figure 3.28) are gaining space in the market, since the sensing element deformation is measured by light (usually infrared). This means that the sensor is more robust, since the deforming surfaces are physically separated from the sensing element. Because of the very small changes in resistance, most common measuring circuits use a Wheatstone Bridge to provide an adequate electric signal as the sensor output. Nowadays, the most common force sensor category found in the market is the *load cell* (Fässler 2010), which is a mechanical structure with a protection housing that accommodates an arrangement of sensing elements (usually strain gauges, but also optic, piezoelectric, pneumatic, and hydraulic). It is constructed to support large forces and deformations in multiple directions, making them very robust, yet heavy and expensive. On the other hand, most piezoelectric sensors are available as single-sensing elements, whose disadvantage is that their movable part

Table 3.3: Strain gauge vs. Piezoelectric

Application	Strain gauge behavior	Piezoelectric behavior	Best
Static long-term monitoring	∗ No drift from normal op. after long use time	∗ High drift (about $1N/min$) ∗ Bad for measuring of small forces after long time	Strain gauge
Dynamic measurement	∗ Low stiffness (soft) ∗ Low resonance freq. ∗ High cut-off freq.	∗ Small deformation and high stiffness ∗ High resonance freq. ∗ Low cut-off freq.	∗ Strain (large forces) ∗ Piezo. (small forces)
Calibration tasks	∗ Conditioning circuit allows error/temperature compensation ∗ Spring can de designed for optimal reproducibility	-	Strain gauge
High initial load	-	∗ Output signal can be short-circuited even if under high initial loads	Piezoelectric
Harsh environments	∗ Some strain gauges provide IP68 protection degree and hermetically sealed enclosures	-	Strain gauge
High accuracy	∗ Modern models provide excellent individual errors of $220ppm$	∗ Sightly higher individual error (0.5% rel. to full scale ∗ High drift	Strain gauge
Constrained space	∗ Generally, requires load cell housing	Can be very compact	Piezoelectric

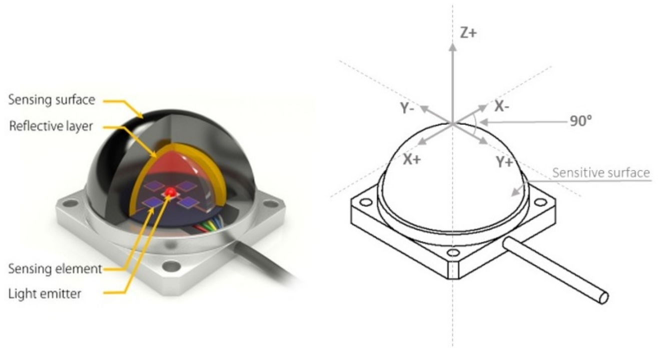


Figure 3.28: Deformation measuring using light.

can only perform small deflections. Depending on the application case, one have to consider to build an assembly of the sensor with a damping element (e.g., a spring).

3.3.3.2 Selected force sensor

As best detailed in A.3, an extensive market search was performed, considering TETIS requirement and the discussing about force sensing approached in this Section. The most suitable choice was the sensor OMD-20-FE-40N (Figure 3.30a), which is a variation of OMD-20-FA-200N, as listed in Table A.7. Together with the manufacturer, this sensor was customized to present the features listed in Table 3.4.

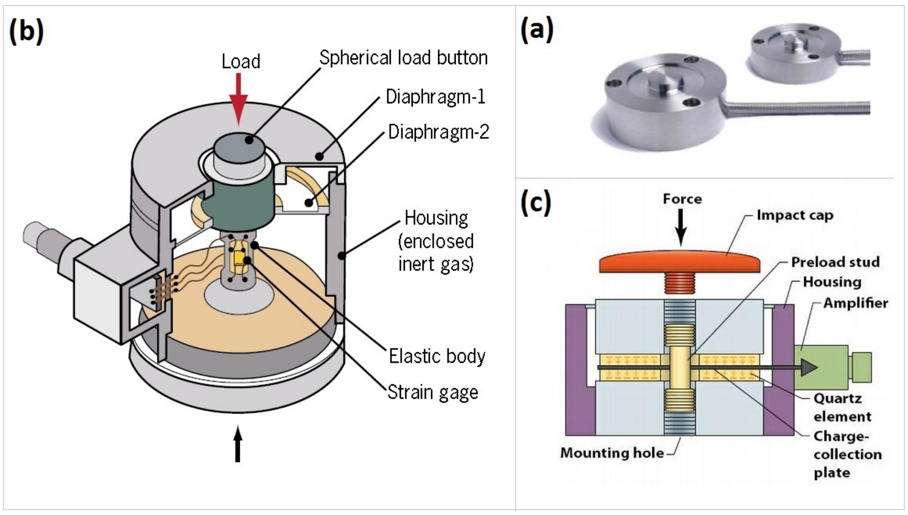


Figure 3.29: Load cell: (a) pictures; (b) scheme of a load cell housing accommodating strain gauges; (c) scheme of a piezoelectric element within a suitable sensor housing.

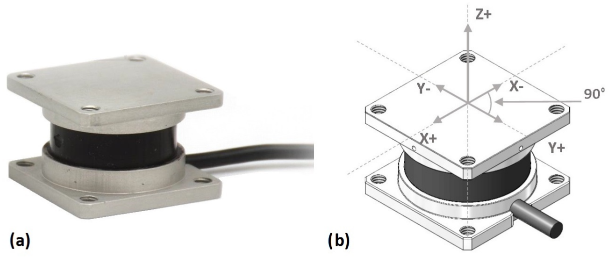


Figure 3.30: Optoforce OMD-20-FE-40N: (a) Illustration; (b) Axes scheme.

Table 3.4: Optoforce OMD-20-FE-40N - Nominal specifications

Model/Item	OMD-20-FE-40N	Justifications
Dimensions (mm)	$16 \times 25 \times 25$	$*$ Fits better on the end-effector
Weight (with $2m$ cable)	11 (35)	$*$ At least $1.5m$ cable is needed to be passed inside all the link tubes
Nominal capacity in z	F_z : $40N$	$*$ It is the softest model that Optoforce offers, which is more adequate for this application than the $200N$ version, since it gives better resolution and we only need 10 N capacity (see A.3)
Resolution	16 bits	-
Typical deformation (for $200N$)	1.2mm	-
Interfaces	CAN/USB	$*$ CAN for compatibility with DORIS actuation bus $*$ USB for quick tests on PC
Surface type	Flat	$*$ Flat surface is better for fixing objects over it, such as the vibration sensor

3.3.4 Integration to the robot system

TETIS control loop (actuators and force sensor) are all integrated in a *Controller Area Network* (CAN) , which is an established robust bus system for real-time control with strict error detection (Farsi et al. 1999). The robot computer is the master node of this network, collecting sensor data (encoder, hall sensor and force measurement) and sending control commands to the drivers. Also, the computer collects the video from the web-camera (directly via USB) and digitized vibration data from a USB Data Acquisition Board (DAQ), model Measurement Computer® USB-1608FS-Plus. The manipulator integration to the robot system is illustrated in Figure 3.31.

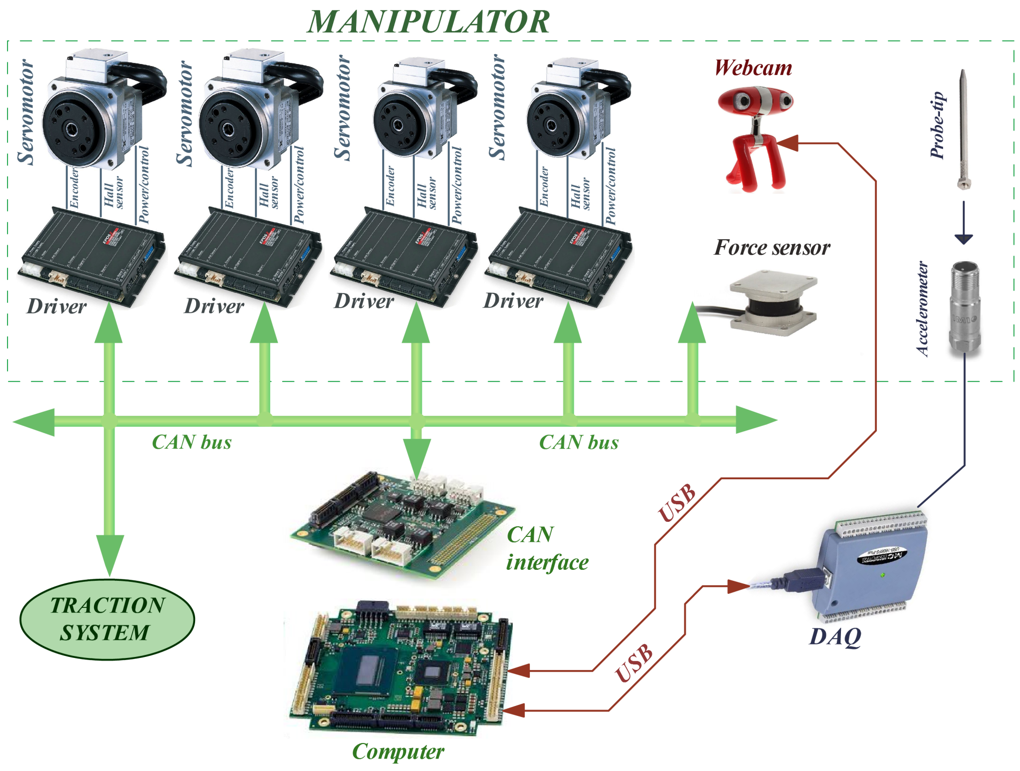


Figure 3.31: Integration to the robot system - Scheme.

3.4 Kinematic model and analysis

3.4.1 Forward kinematics

Together with the DORIS base movement (which we may call *Joint* 1), TETIS is a 5-DoF manipulator. Using the tools reviewed in Section 2.1, we can define its forward kinematics. In Figure 3.32, we can check the manipulator in its retracted pose and the assigned link frames using both DH Standard and Modified approaches. According to the mechanical project, the arm lengths are defined to be: $E_2 = 52.5mm$ (link 2 length), $E_3 = 320mm$ (link 3 length), $E_4 = 225mm$ (link 4 length), $E_5 = 167.25mm$ (link 5 length), $M_3 = M_4 = 55.775mm$ (joints 3 and 4 offsets), $M_5 = 57mm$ (joint 5 offset). Notice that the base frame was defined to be positive in x-axis direction towards the movement of DORIS base along the rail.

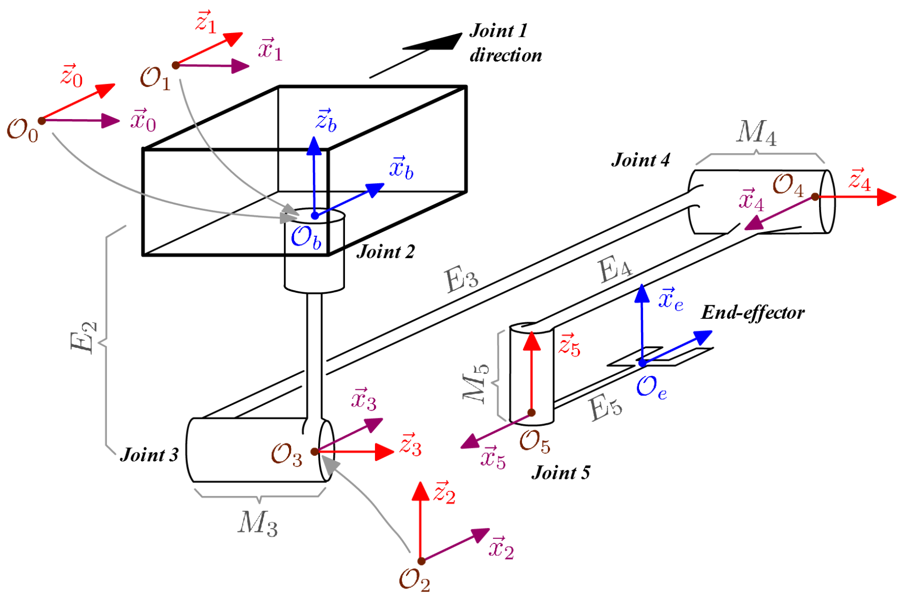


Figure 3.32: TETIS representation with DH Modified parameters.

Using DH Modified parameters, the transformation from the base frame to $\bar{E}_0$ is given by:

$$T_{b0}(q) = \begin{bmatrix} R_{b0} & (\vec{p}_{b0})_b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.2}$$

The transformation from $\bar{E}_5$ to the end-effector frame is given by:

$$T_{5e}(q) = \begin{bmatrix} R_{5e} & (\vec{p}_{5e})_5 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 & -E_5 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.3}$$

And finally, the DHM parameters are defined at Table 3.5.

Table 3.5: TETIS DHM parameters

Link i	a_{i-1}	α_{i-1}	d_i	θ_i
Link 1	0	0	q_1	0
Link 2	0	$\pi/2$	$-E_2$	$\pi/2 + q_2$
Link 3	0	$\pi/2$	0	q_3
Link 4	E_3	0	0	$\pi + q_4$
Link 5	E_4	$\pi/2$	$-M_5$	q_5

3.4.2 Preliminary analysis

This structure was the first purposed version of DORIS manipulator, and the most concerned features about it were its retraction capacity and limited weight/size. However, notice that some design flaws can be highlighted:

- *Dependence of the robot movement along the rail*: so far, we are considering that the movement of DORIS along rail is equivalent to an ideal prismatic joint. This is a dangerous assumption, since DORIS pose along the rail is very inaccurate, has low repeatability, may not be perfectly straight and limits the rail layout not to be curved near equipment of interest (Figure 3.3).
- *Non-functionally redundant structure*: the proposed manipulator has a 5-DoF structure for a 5-DoF task, which turns it an structure without functional redundancy (Sections 2.2.1 and 2.6.5). However, redundant structures are highly recommended, since the extra DoFs can be availed for achieving an extra control objective or to take the robot out of a singular configuration.
- *Absence of kinematic decoupling*: in an structure with kinematic decoupling: (i) the inverse kinematics is easier to be found; (ii) the singular configurations are well defined (Siciliano et al. 2009).

3.4.3 Geometric Jacobian and singularity analysis

According to the concepts approached in Section 2.5.1, the end-effector geometric Jacobian with respect to $\bar{E}_e$ is given by:

$$J = (J_{Ge})_e = \begin{bmatrix} J_1 & J_2 & J_3 & J_4 & J_5 \end{bmatrix}, \tag{3.4}$$

where:

$$J_1 = \begin{bmatrix} -c_2 s_{34} \\ s_2 c_5 + c_2 s_5 c_{34} \\ c_2 c_{34} c_5 - s_2 s_5 \\ 0 \\ 0 \end{bmatrix} \quad J_2 = \begin{bmatrix} E_5 s_{34} s_5 \\ -E_3 c_3 c_5 + E_4 c_{34} c_5 - E_5 c_{34} - M_5 s_{34} c_5 \\ s_5(E_3 c_3 - E_4 c_{34} + M_5 s_{34}) \\ c_{34} \\ s_5 s_{34} \end{bmatrix} \tag{3.5}$$

$$J_3 = \begin{bmatrix} E_3 c_4 - E_4 + E_5 c_5 \\ s_5(E_3 s_4 + M_5) \\ c_5(E_3 s_4 + M_5) \\ 0 \\ c_5 \end{bmatrix} \quad J_4 = \begin{bmatrix} -E_4 + E_5 c_5 \\ M_5 s_5 \\ M_5 c_5 \\ 0 \\ c_5 \end{bmatrix} \quad J_5 = \begin{bmatrix} 0 \\ -E_5 \\ 0 \\ 1 \\ 0 \end{bmatrix} \tag{3.6}$$

To analyze singularities, we can search for joint configurations q_s such that $det(J(q_s)) = 0$. The determinant of the above Jacobian is given by:

$$det(J) = -E_3^2 c_2 c_3^2 c_5 + E_3 E_4 (s_2 c_3 s_5 - s_2 c_{34} c_5 s_5 + c_2 c_3 c_{34} c_5) + E_3 M_5 (c_2 c_3^2 s_4 c_5 + s_2 s_{34} c_4 c_5), \tag{3.7}$$

or by the factored form:

$$det(J) = -E_3 c_2 c_3 c_5 (-E_3 c_3 + E_4 c_{34} + M_5 c_3 s_4) + E_3 s_2 [c_5(-E_4 c_{34} c_5 + M_5 s_{34} c_4) + E_4 c_3 s_5]. \tag{3.8}$$

3.4.3.1 Singularities

As we can notice, it is difficult to determine all the solutions for $det(J) = 0$. As discussed in Section 2.5.4, the complexity of $det(J)$ is reduced if the robot has kinematic decoupling of position and orientation, which is not TETIS case. Hence, it is worth seeking for some solutions by visual analysis of (3.8), among those we can highlight:

(i) $q_2 = \{0, \pi\}, q_3 = \{\pm\pi/2\}, q_4 \in [0, 2\pi], q_5 \in [0, 2\pi]$

(ii) $q_2 = \{0, \pi\}, q_3 \in [0, 2\pi], q_4 \in [0, 2\pi], q_5 = \{\pm\pi/2\}$

(iii) $q_2 \in [0, 2\pi], q_3 = \{\pm\pi/2\}, q_4 \in [0, 2\pi], q_5 = \{\pm\pi/2\}$

(iv) $q_2 = \{\pm\pi/2\}, q_3 = \{0, \pi\}, q_4 = \{\pm\pi/2\}, q_5 = \{0, \pi\}$.

In addition, notice that some classic arm singularities can be found only by visual inspection. For example, in spite of the advantages of an elbow configuration in joints 3 and 4 for better arm outstretching and compact retraction (as discussed in 3.2.2), their axes have the same direction, which almost characterizes an elbow singularity (Siciliano et al. 2009).

3.4.4 Suggestions for modifications of the original design

In view of the problems listed above (difficulty of finding explicit expressions for singularity analysis and inverse kinematics, many singular configurations found, etc.), we propose design changes in the first TETIS version in order to overcome these flaws.

3.4.4.1 Modification 1 - Joint swap

A first modification would be to swap the 4^{rd} and 5^{th} joints, is order to eliminate the situation of two consecutive joints with parallel axes. In order to keep the original arm dimensions, the new robot TETIS version would be as illustrated in Figure 3.33a. Possible disadvantages of this new configuration would be: (i) reduction of mechanical limits for the 5^{th} joint; (ii) mechanical challenges of altering the original tube/joint design, as detailed in Section 3.2.7.

3.4.4.2 Modification 2 - Extra degree-of-freedom

A second modification would be to include an extra DoF in the original structure, so that the manipulator becomes redundant, hence resulting in the advantages described in Section 3.4.2. The best position to place the new revolute joint would be between the 3^{rd} and 4^{th} joints (Figure 3.33b), which would eliminate the same problem described above (Section 3.4.4.1). Possible disadvantages of this new configuration would be the increase in the total weight and occupied space, besides the mechanical complexity of redesign.

3.4.4.3 Modification 3 - Inclusion of offsets

As we can notice, the lack of angle offsets (especially if $a_i = k\pi/2,\ k \in \mathbb{Z}$) increases the chance of undesired situations like parallel of crossing axes of two consecutive joints. Thus, it is worth including small length and angle offsets in the manipulator original structure, like illustrated in Figure 3.33c. Likewise, this would increase the complexity of the tubes and joints mechanical redesign.

3.4.4.4 Modification 4 - Kinematic decoupling

A great change in TETIS would be to render it to an structure with kinematic decoupling, in which we would take great avail due to the advantages presented in Section 2.5.4. For this, we must define a point in which the rotation axes of the last two joints remain crossed at a single point regardless of the joint space configuration. A simple solution would be to bring the 5^{th} joint next to the 4^{th} joint, as illustrated in Figure 3.33d. However, a priori, this would eliminate the length of the 4^{th} link, which would reduce the robot extended arm range.

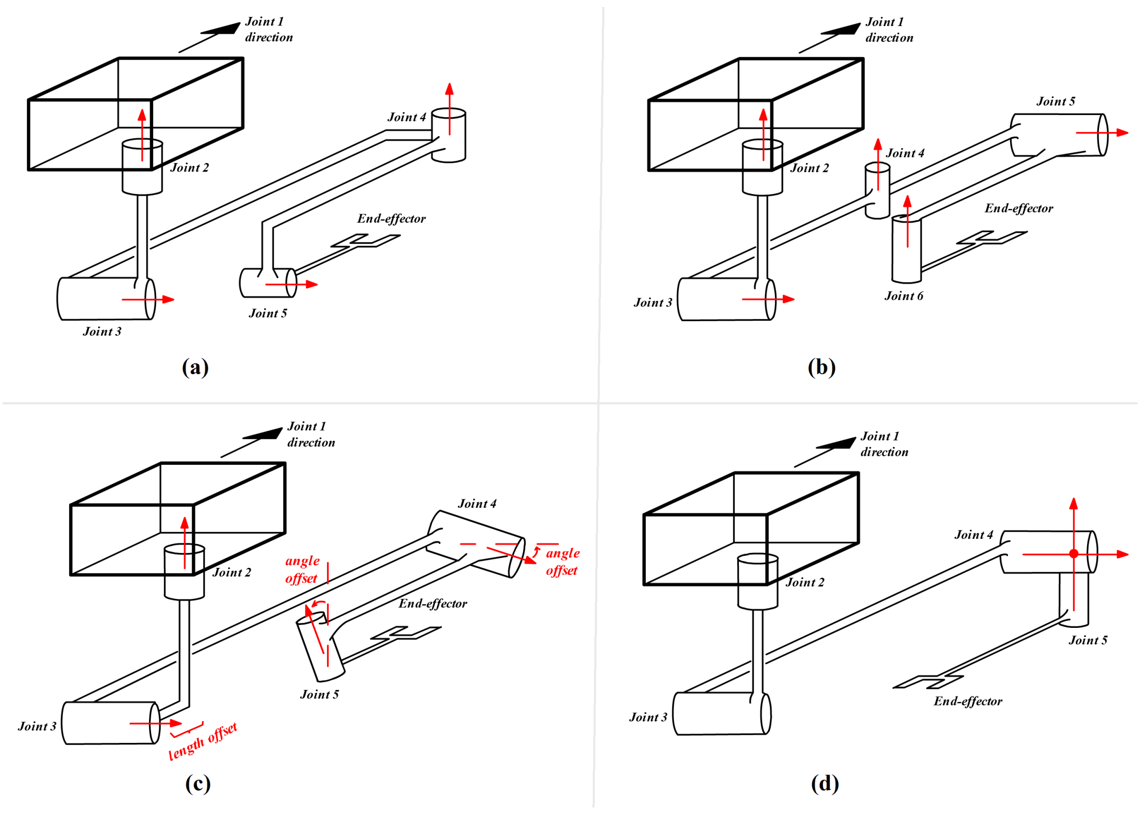


Figure 3.33: Proposed modifications of TETIS design: (a) joint swap; (b) extra DoF; (c) offsets; (d) kinematic decoupling.

3.4.4.5 Final proposal - All modifications

Synthesising the optimal structure for a specific task is a challenging activity, which is an important research subject in Robotics area. For example, Meng et al. (2007) proposes an theoretical and closed-form algorithm for synthesis and dimensional optimization of sub-6 DoF parallel manipulators based on the DoFs of the required task and other design constraints. However, in practice, the application of this algorithm is very difficult and demands advanced knowledge of Lie algebra[3], which is not being approached in this work.

In our scope, a better solution would be the to combine all the above suggestions, in order to take avail of the advantages arising from each one. In view of this, two different structures are proposed (Figure 3.34) and modelled with DHM parameters, as shown in Tables 3.6 and 3.7, respectively. The fixed homogeneous transformations T_{b0} and T_{6e} for both structures are given by:

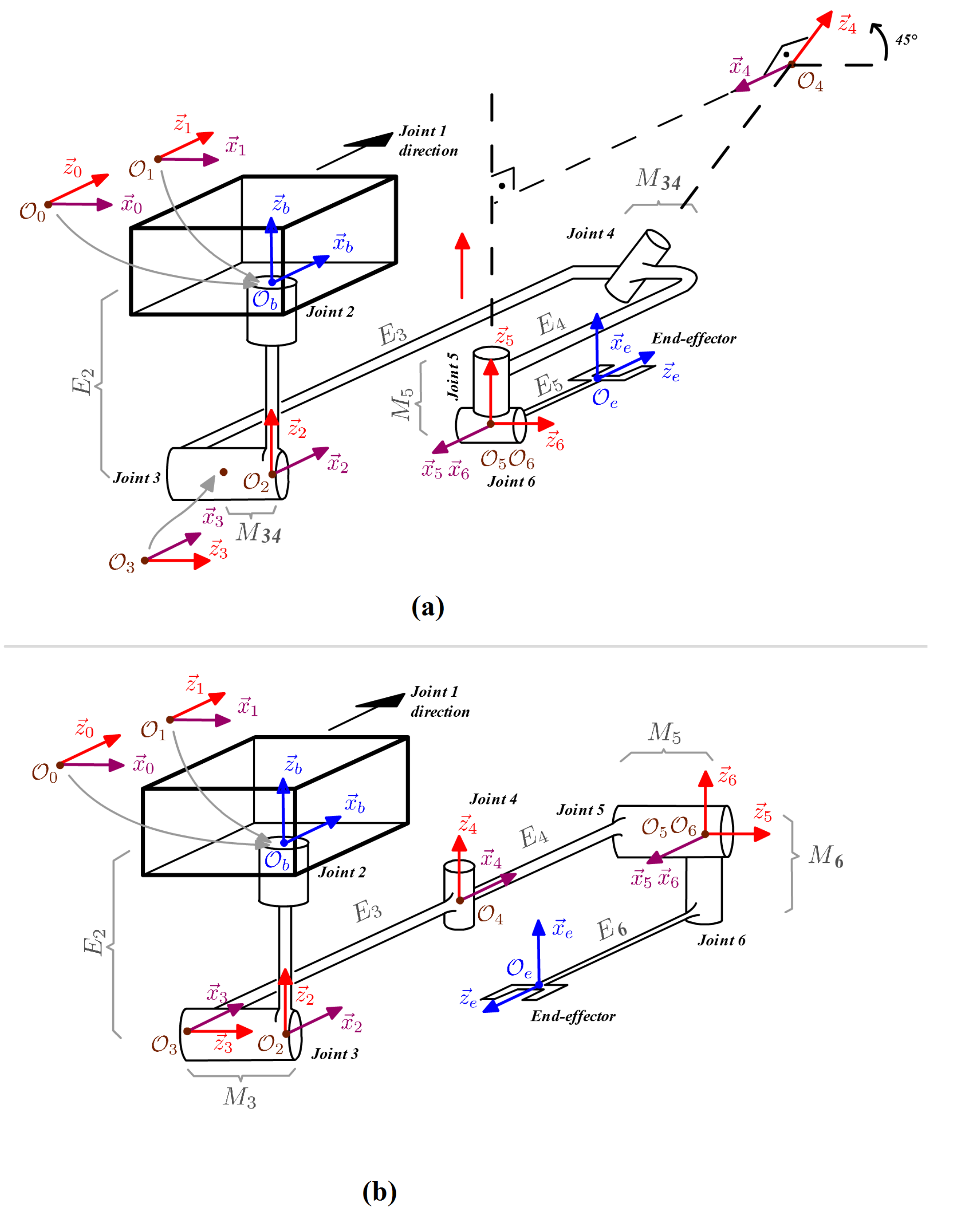
$$T_{b0,both}(q) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

$$T_{6e,struc.1}(q) = \begin{bmatrix} 0 & 0 & -1 & -E_5 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_{6e,struc.2}(q) = \begin{bmatrix} 0 & 0 & 1 & E_6 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -M_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

and the robot dimensions are maintained as possible. Structure 1: $E_2 = 52.5, E_3 = 320, E_4 = 225, E_5 = 167.25, M_{34} = 27.888, M_5 = 55.775$. Structure 2: $E_2 = 52.5, E_3 = 160, E_4 = 160, E_6 = 225, M_3 = 55.775, M_5 = 55.775, M_6 = 57$.

As we can notice, the first structure has one extra DoF, an angle offset at 4^{th} joint, maintains the original dimensions and has a kinematic decoupling point caused by the crossing of axes $\vec{z}_5$ and $\vec{z}_6$. However, the new pose of the 4^{th} joint deteriorates the range of both links E_3 and E_4 fully outstretched. On the other hand, this does not happen in the second structure, but in practice, it has one less link. Depending of the project dimensions for each link, the second structure seems the best one, due to its simplicity.

[3]Lie algebra (From 2010) is a math branch for study of infinitesimal rotations, and another powerful tool for the study of rigid body motion and Robotics.



(a)
$\vec{z}_0$ $\vec{x}_0$ $\mathcal{O}_0$
$\vec{z}_1$ $\vec{x}_1$ $\mathcal{O}_1$
Joint 1 direction
$\vec{z}_b$ $\vec{x}_b$ $\mathcal{O}_b$
Joint 2
Joint 3
Joint 4
Joint 5
Joint 6
End-effector
E_2 E_3 E_4 E_5
M_{34} M_5
$\vec{z}_2$ $\vec{x}_2$ $\mathcal{O}_2$
$\vec{z}_3$ $\vec{x}_3$ $\mathcal{O}_3$
$\vec{z}_4$ $\vec{x}_4$ $\mathcal{O}_4$
45°
$\vec{z}_5$ $\vec{z}_6$ $\vec{x}_5$ $\vec{x}_6$ $\mathcal{O}_5\mathcal{O}_6$
$\vec{x}_e$ $\vec{z}_e$ $\mathcal{O}_e$
(b)
$\vec{z}_0$ $\vec{x}_0$ $\mathcal{O}_0$
$\vec{z}_1$ $\vec{x}_1$ $\mathcal{O}_1$
Joint 1 direction
$\vec{z}_b$ $\vec{x}_b$ $\mathcal{O}_b$
Joint 2
Joint 3
Joint 4
Joint 5
Joint 6
End-effector
E_2 E_3 E_4 E_6
M_3 M_5 M_6
$\vec{z}_2$ $\vec{x}_2$ $\mathcal{O}_2$
$\vec{z}_3$ $\vec{x}_3$ $\mathcal{O}_3$
$\vec{z}_4$ $\vec{x}_4$ $\mathcal{O}_4$
$\vec{z}_5$ $\vec{z}_6$ $\vec{x}_5$ $\vec{x}_6$ $\mathcal{O}_5\mathcal{O}_6$
$\vec{x}_e$ $\vec{z}_e$ $\mathcal{O}_e$

Figure 3.34: Final version of proposed modifications in TETIS design: (a) structure 1; (b) structure 2.

Table 3.6: Final proposal 1: DHM parameters

Link i	a_{i-1}	α_{i-1}	d_i	θ_i
Link 1	0	0	q_1	0
Link 2	0	$\pi/2$	$-E_2$	$\pi/2 + q_2$
Link 3	0	$\pi/2$	$-M_{34}$	q_3
Link 4	E_3	$-\pi/4$	$M_{34}\sqrt{2}$	$\pi + q_4$
Link 5	E_4	$\pi/4$	$-M_{34}$	q_5
Link 6	0	$-\pi/2$	0	q_6

Table 3.7: Final proposal 2: DHM parameters

Link i	a_{i-1}	α_{i-1}	d_i	θ_i
Link 1	0	0	q_1	0
Link 2	0	$\pi/2$	$-E_2$	$\pi/2 + q_2$
Link 3	0	$\pi/2$	$-M_3$	q_3
Link 4	E_3	$-\pi/2$	0	q_4
Link 5	E_4	$\pi/2$	M_5	$\pi/2 + q_5$
Link 6	0	$\pi/2$	0	q_6

Chapter 4

Manipulator Operation and Control

This chapter presents the operation modes of TETIS, the manipulator of DORIS robot. This includes the description and flowchart that depicts the manipulator operation modes, tasks and phases, and the control schemes used in each phase (with the assistance of the concepts reviewed in Chapter 2).

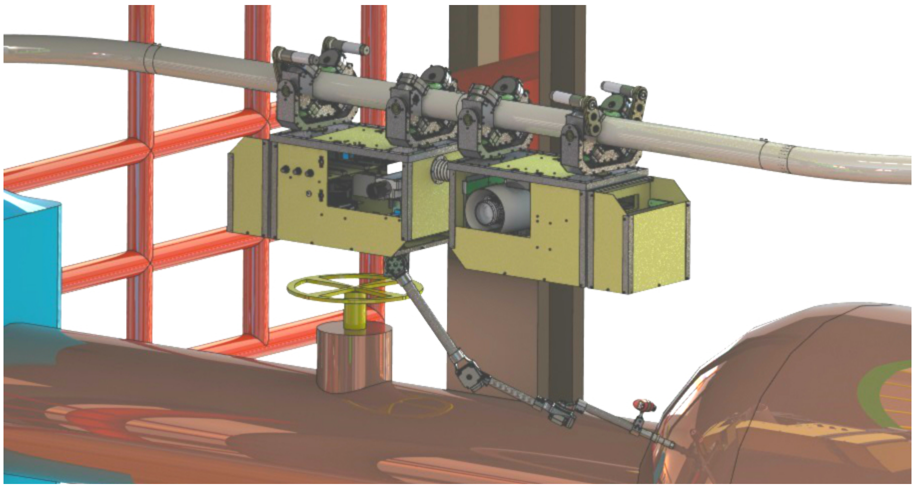


Figure 4.1: Illustration of DORIS manipulator being operated during a DORIS patrol throughout a plant.

4.1 Summary of requirements

This section summarizes the required tasks and initial conditions/constraints for the manipulator operation.

4.1.1 Manipulator tasks

***Automatic posing* of vibration sensor 623C00 (with probe tip) over the surface of a nearby machine of interest**

Notice:

- For appropriate measuring, the probe must be posed perpendicularly over the surface, regardless of the *roll* orientation around $\vec{z}_e$-axis.
- The probe tip must remain fully in contact with the surface and standstill during the entire procedure.

Automatic interaction* of special *fingertip* over nearby *touchscreens

Notice:

- The robot must "know" exactly what the fingertip should do over the *touch-screen* device. It includes the finger touching, withdraw and traversing a path over the surface.
- An appropriate orientation control will guarantee that the finger remains always posed perpendicularly with respect to the surface, regardless of the *roll* orientation around $\vec{z}_e$ axis.

***Manual driving* of a stereo camera through nearby space**

Notice:

- The camera manual moving task is the manipulator teleoperation via *joystick* from DORIS remote base, where the operator moves the end-effector guided by the real-time video transmitted it.
- The camera video should undergo real-time linear transformations in order to correct adjust the video alignment with the operator screen.

4.1.2 Conditions/constraints

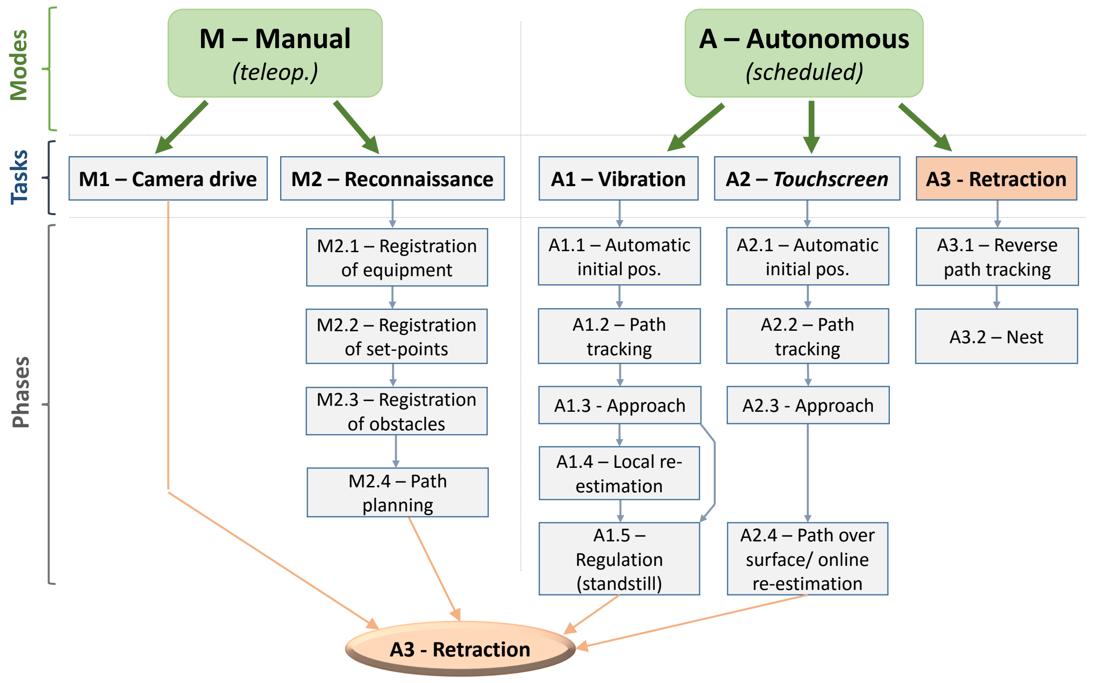
The following conditions and constraints should be highlighted:

- The manipulator structure cannot collide under any circumstances with any platform structure/equipment. The only part of the manipulator system that can controllably touch an environment object is the end-effector tip.

- Neither the equipment surface nor the sensor/manipulator must be damaged due to uncontrolled/forced contact.
- Only small movements of the DORIS robot along the rail are permitted (prismatic joint). Thus, the rail section near equipment of interest cannot be curved.
- The manipulator must remain in retracted position during the motion of the DORIS robot along the rail, as well as DORIS cannot travel if the manipulator is operating.
- The parameters of the environment, the DORIS robot and the TETIS manipulator are considered to be fully known, as defined in Section 3.2.1.

4.2 Operation modes - Flowchart

Figure 4.2 summarizes in a flowchart the operation modes and procedures of TETIS manipulator, which are fully described in the next sections.



Modes
Tasks
Phases
M – Manual
(teleop.)
A – Autonomous
(scheduled)
M1 – Camera drive
M2 – Reconnaissance
A1 – Vibration
A2 – Touchscreen
A3 - Retraction
M2.1 – Registration of equipment
M2.2 – Registration of set-points
M2.3 – Registration of obstacles
M2.4 – Path planning
A1.1 – Automatic initial pos.
A1.2 – Path tracking
A1.3 - Approach
A1.4 – Local re-estimation
A1.5 – Regulation (standstill)
A2.1 – Automatic initial pos.
A2.2 – Path tracking
A2.3 - Approach
A2.4 – Path over surface/ online re-estimation
A3.1 – Reverse path tracking
A3.2 – Nest
A3 - Retraction

Figure 4.2: Operation modes - Flowchart.

4.3 Mode M: Manual (or *teleoperated*)

In the *manual* mode, skilled personnel (located at a remote operation base) *teleoperate* DORIS robot and and TETIS manipulator via joystick to perform different tasks. This procedure has the assistance of TETIS stereo camera, automatic force control in interactions, visual/audible alarms, blocking mechanism and force feedback, which may avoid rough contacts between the end-effector and goal surfaces and warn the operator if the arm structure gets very close to any platform object. Each input given by the operation is transduced to a position and orientation set point, to where, afterwards, the manipulator is automatically controlled.

Before any manual task, the operator should choose a suitable reference *stop point* (P_1) along the rail, and takes the DORIS robot up to this point, which is from where they will command the initial approach of towards an equipment of interest (E_1) or camera movimentation. The the first joint of TETIS manipulator is prismatic (see Section 3.2.1), thus: (i) P_1 must be located on a straight section of the rail, i.e., away from corners; (ii) DORIS robot movements close P_1 must be small enough so that the robot never reach a curved rail section.

Point P_1 serves as the manipulator base frame $\bar{E}_b$ origin for any future automatic tasks (e.g., vibration inspection of equipment E_1). Thus, considering that the robot position P_1 and orientation R_{wb} with respect to a global inertial frame $\bar{E}_w$ of the platform is known, then, we can define the homogeneous transformation of $\bar{E}_w$ to $\bar{E}_b$ as:

$$T_{wb}(P_1) = \begin{bmatrix} R_{wb}(P_1) & P_1 \\ 0 & 1 \end{bmatrix}. \tag{4.1}$$

After the manual task, the teleoperator must trigger the autonomous mode to set the retraction task A3 (see Section 4.4.3).

4.3.1 Task M1: Camera drive

In this task, the TETIS end-effector is teleoperated to arbitrary positions and orientations so that the operator can register video from different sights of nearby areas. The camera frame $\bar{E}_c$ is solidary to the end-effector frame $\bar{E}_e$, whereas the operator screen $\bar{E}_t$ is solidary do the platform world frame $\bar{E}_w$. Thus, the net transformation between the screen and the image is given by:

$$T_{te} = T_{tw} T_{wb}(P_1) T_{be}(q) T_{ec}. \tag{4.2}$$

Once the operator finishes this task, the manipulator is automatically retracted to its initial pose, which procedure is described by task A3 (Section 4.4.3).

4.3.2 Task M2: Reconnaissance

In *reconnaissance* task, the manipulator is taken to several platform sites so that the operator can register necessary data of platform machine of interest. The equipment can be a *touchscreen* - which requires registration of the path that the *fingertip* should follow over, or a rotating machine, whose vibration inspection is of interest. The collected data is necessary to calibrate inputs for the automatic tasks A1 and A2 (Section 4.1.1). This task should be performed whenever:

(i) DORIS robot is operating for the first time in an unknown offshore environment;

(ii) New and unknown equipment of interest or *touchscreen* surfaces are inserted into the platform environment;

(iii) An interest in existing platform equipment shows up, but their information are still unknown.

Task M2 should be performed several times during the same patrol of DORIS robot along the rail if there are multiple equipment of interest (e.g., E_1, E_2, E_3...). For a general equipment E_1, this procedure consists of the following phases:

4.3.2.1 Phase M2.1: Registration of equipment

The operator registers in DORIS mission control system the following information:

- *World-to-base homogeneous transformation* $T_{wb}(P_1)$, as given by (4.3).
- E_1 *type* (e.g.: gas compressor, pump, valve, oven, *touchscreen*, etc.).
- E_1 *tag* (should be unique in the industrial plant).
- E_1 *purpose* (vibration inspection or *touchscreen* interaction).
- E_1 *3D model* (if available).
- E_1 *surface material* and end-effector type (fingertip or probe-tip), which is important to estimated the *coefficient of static friction* μ_{s1} between the end-effector tip and E_1 surface. The friction coefficients are necessary to determine E_1 *cone of friction* (see Section 2.7.3.2). The cone friction define

$\alpha_{s1} = tan^{-1}(\mu_{s1})$ as the maximum limit for the angle β between the tool $\vec{z}_e$ axis and the surface normal $\vec{z}_s$. The angle β is defined as $tan(\beta) = \frac{\|\vec{z}_e \times \vec{z}_s\|}{\|\vec{z}_e \cdot \vec{z}_s\|}$, and the manipulator tool will not slide if $\beta \leq \alpha_{s1}$.

4.3.2.2 Phase M2.2: Registration of set-points

The operator approaches the TETIS end-effector towards a desired point over E_1 surface where it is supposed to be touched first. Beforehand, the end-effector approach must occurs within E_1 cone of friction limits, otherwise the it slides indefinitely over the surface. Once touched, the end-effector automatically adjusts itself to be perpendicular posed with respected to the surface. Then, the following data are registered: At this moment, the operator registers the , which consists of:

- *Touch set-point position* $T_1 = \begin{bmatrix} Tx_1 & Ty_1 & Tz_1 \end{bmatrix}^T$ with respect to $\bar{E}_b$.
- *Approach set-point position* $H_1 = \begin{bmatrix} Hx_1 & Hy_1 & Hz_1 \end{bmatrix}^T$ with respect to to $\bar{E}_b$.
- *Surface orientation matrix* $R_{bs1} = \begin{bmatrix} (\vec{x}_{s1})_b & (\vec{y}_{s1})_b & (\vec{z}_{s1})_b \end{bmatrix}$ with respect to $\bar{E}_b$.

It is worth highlighting that H_1 is a point slightly perpendicularly distant by δ from T_1 to where the TETIS end-effector is taken before the final straight approach until it touches the surface (Figure 4.3a).

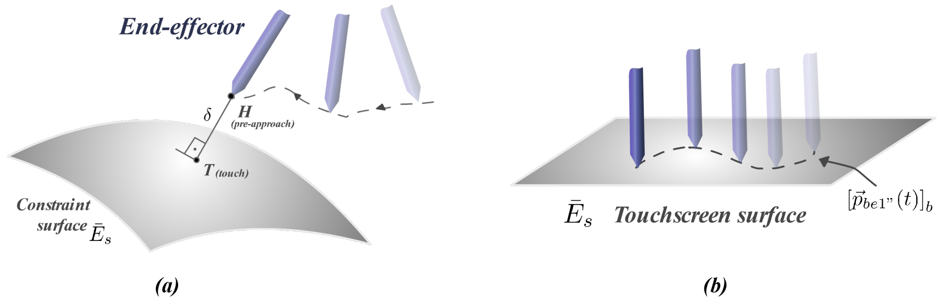


Figure 4.3: Concepts: (a) approach distance; (b) path over surface.

Automatically, the a flat contact surface γ_1 is defined as:

$$\gamma_1 = \left\{ p \in \mathbb{R}^{3\times1}, (\vec{z}_{s1})_b^T \left[p - T_1 \right] = 0 \right\}. \tag{4.3}$$

If the desired task is the interaction with *touchscreens*, the operator should also control the end-effector to follow a desired flat trajectory $\Pi" = \left[\vec{p}_{be1"}(t)\right]_b \in \gamma_1$ (Figure 4.3b) over the surface, which is also registered by the system.

4.3.2.3 Phase M2.3: Registration of obstacles

The operator registers all the nearby obstacles that should be avoided during the manipulator control towards H_1, such as: piping, frames, valves, walls, or other equipment out of context. The obstacles are represented by a set of n points $\{O1_1, O2_1, O3_1, \cdots On_1\}$ in a Cartesian space with respect to $\bar{E}_b$. The identification of obstacles is assisted by the stereo camera video.

4.3.2.4 Phase M2.4: Path planning

Using all the registered information, the system automatically plans a suitable path:

$$\Pi' = \left\{ [\vec{p}_{be1'}(t)]_b \in \mathbb{R}^3, R_{be1'}(t) \in \mathbb{R}^{3\times3} \right\}, \tag{4.4}$$

as illustrated in Figure 4.4, to be followed by the TETIS end-effector from the retracted position towards the set-points, which should avoid the obstacles and take into account the joint limits and the robot boundaries. If no suitable path can be defined, an alarm is emitted and the system automatically ban future scheduled tasks for the given equipment.

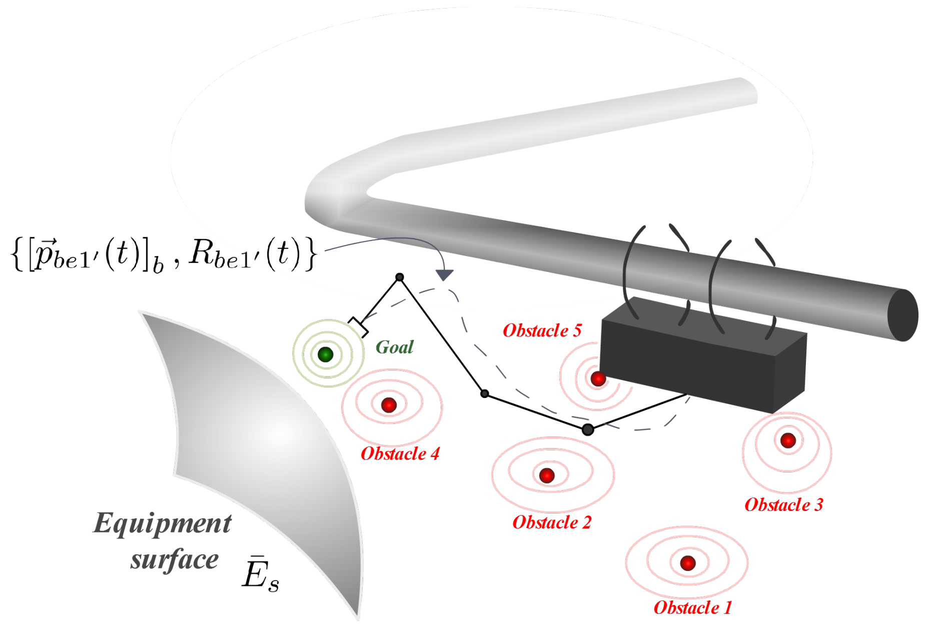


Figure 4.4: Path planning avoiding obstacles.

After this step, task M2 is finished for E_1, and it may be repeated for other equipment, if needed. Once this phase is finished, the manipulator is automatically retracted to its initial pose, which procedure is described by task A3 (Section 4.4.3).

4.4 Mode A: Autonomous (or *scheduled*)

In this mode, the robot performs autonomously pre-scheduled tasks of vibration inspection or interaction with *touchscreens*. Basically, these two tasks works in similar way, since TETIS end-effector undergoes kinematic control to reference points (comparison in Table 4.1).

Table 4.1: Comparison between vibration and *touchscreen* interaction tasks

Phase	Vibration	*Touchscreen*	Comments
Initial stop	P_1	P_1	Next moves are referenced to $\bar{E}_b$
Path tracking towards the surface	Kinematic control	Kinematic control	Position, roll and pitch
Approach until touch	Hybrid control of force and position	Hybrid control of force and position	Contact force depends on the surface type
Surface re-estimation	Yes, using techniques from Sections 2.7.9.1 and 2.7.9.2	See next phase	Machine surfaces allow small movements of the end-effector tip over them, but touchscreen surfaces do not
Path over surface	Standstill (regulation) for certain time until vibration measuring ends	Path tracking (if path is not a single point, online estimation is allowed using Section 2.7.9.1 method)	Both need to be realized always in full contact and perpendicularly posed to the surface

A *scheduled task* occurs at some moment during the patrol task carried out by DORIS robot throughout the platform environment. For a certain equipment $E_1$1, this procedure can *only* be triggered if task M2 (reconnaissance) is successfully concluded for E_1 already.

4.4.1 Task A1: Vibration inspection

In this task, the vibration sensor prob tip must be positioned in fully contact with the external surface of machinery to be measured during an arbitrary time. For this, the following phases are followed:

4.4.1.1 Phase A1.1: Automatic initial positioning

The DORIS robot is automatically taken up to the *stop point* P_1 and stops. If the stored data from E_1 and the environment are represented in the platform/world frame $\bar{E}_w$, they must undergo the transformation $T_{wb}(P_1)$ before the execution of next phases. If E_1/environment data were collected during the reconnaissance task (Section 4.3.2), they are already represented in the manipulator base frame $\bar{E}_b$.

In many commercial models, there exist the concepts of *nest* (see 4.4.3.2) and *ready* poses. *Nest* is a rest pose q_{nest} in which the structure can be released when it is not being operated, which would be the case when the DORIS robot is travelling along the rail. Generally, the nest pose is over or very near singular configurations. Taking the robot out of this configuration may be a difficult task with high power consumption. Therefore, this commercial models have internal routines that control the manipulator from the nest to a *ready* pose q_{ready}, which is a configuration far from singularities from where the robot can depart a more comprehensive path tracking.

4.4.1.2 Phase A1.2: Path tracking

The manipulator is automatically controlled from the nest or ready poses so that the end-effector frame $\bar{E}_e$ follows a defined path $\{[\vec{p}_{be1'}(t)]_b \in \mathbb{R}^3, R_{be1'}(t) \in \mathbb{R}^{3\times 3}\}$ (Section 4.3.2.4) that leads it safely to the approach point H_1 and aligned with the surface, i.e., $T_{be} \to T_{bs1}$.

The kinematic control law is defined as in Section 2.6.2.1:

$$u = \dot{q} = (\bar{J}_{Ge})_e^{\dagger} \bar{\nu}_e, \tag{4.5}$$

where $\bar{J}_{Ge}$ is the manipulator geometric Jacobian expressed with respect to the frame $\bar{E}_e$ and $\bar{\nu}_e$ is the Cartesian control law to be designed. Here, the following assumptions are considered:

(i) The end-effector roll angle around $\vec{z}_e$-axis is not controlled. This decoupling can be simply achieve by eliminating the last row of the complete Jacobian $(J_{Ge})_e$, i.e., $(\bar{J}_{Ge})_e = \begin{bmatrix} I^{5\times 5} & 0^{5\times 1} \end{bmatrix} (J_{Ge})_e$.

(ii) The orientation will be represented in unit quaternions (Siciliano et al. 2009).

The Cartesian control law $\bar{\nu}_e = \begin{bmatrix} I^{5\times 5} & 0^{5\times 1} \end{bmatrix} \nu_e$ is defined as follows:

$$\nu_e = \begin{bmatrix} (\nu_p)_e^T & (\nu_o)_e^T \end{bmatrix}^T, \quad (\nu_p)_e = R_{eb}(\nu_p)_b. \tag{4.6}$$

The position control law is defined as:

$$(\nu_p)_b = (\dot{p}_d)_b + K_p(e_p)_b, \tag{4.7}$$

where $(p_d)_b = [\vec{p}_{be1'}(t)]_b$ $p_b = [\vec{p}_{be1}(t)]_b$, $(e_p)_b = (p_d)_b - p_b$ and $K_p = K_p^T > 0$ is a diagonal positive-definite matrix. The orientation control law is defined as:

$$(\nu_o)_e = (\omega_d)_e + K_o e_\phi, \tag{4.8}$$

where $K_o = K_o^T > 0$ is a positive-definite matrix, $(\omega_d)_e$ is defined from $R_{be1'}(t)$, and e_ϕ is the orientation error (Section 2.6.4.2), defined using the *body frame approach* in terms of the quaternion formulation, as follows:

$$e_\phi = \eta\epsilon_d - \eta_d\epsilon + \hat{\epsilon}_d\epsilon, \tag{4.9}$$

where $Q_d = (\eta_d, \epsilon_d)$ is the quaternion representation of the desired pose $R_d = R_{be1'}(t)$, and $Q = (\eta, \epsilon)$ is the quaternion representation of the current pose $R = R_{be}(t)$. Since we are using the body frame approach, the orientation error e_ϕ is already represented in the tool frame.

When the robot follows the desired path until H_1, this phase is completed and the next phase is triggered.

4.4.1.3 Phase A1.3: Approach

The TETIS end-effector is controlled from the point H_1 by regulating its position and orientation in the direction of the x and y-axes, and regulating the force in the direction of the z-axis.. This means that, once pointing to the surface with the perpendicular orientation, the end-effector will approach to T_1 trying to maintain the projected position in the x and y-axes over the surface, but controlling the contact force to a desired value F_{c1} considered safe enough to not perform a rough hit. This phase ends when the robot evaluates that the end-effector tip is pressing against the surface with the desired force F_{c1}.

As discussed in Section 2.7.2, unlike compliance/stiffness control, the hybrid control approach is a more suitable methodology to carry out interaction tasks that demand an accurate force control, since it deals directly with the force measurement. Hence, similarly to (4.5), the control law is defined as:

$$u = \dot{q} = (\bar{J}_{Ge})_e^\dagger \bar{\nu}_e, \tag{4.10}$$

where $\bar{\nu}_e = \begin{bmatrix} I^{5\times5} & 0^{5\times1} \end{bmatrix} \nu_e$. Then, the Cartesian control is composed by the hybrid control signal and the orientation control signal, as follows:

$$\nu_e = \begin{bmatrix} (\nu_h)_e^T & (\nu_o)_e^T \end{bmatrix}^T. \tag{4.11}$$

The hybrid control signal (see 2.7.8) is defined as:

$$(\nu_h)_e = (\nu_{hf})_e + (\nu_{hp})_e, \tag{4.12}$$

where

$$\begin{aligned}(\nu_{hf})_e &= R_{es} S_f R_{se} (\nu_f)_e \\ (\nu_{hp})_e &= R_{es} S_p R_{sb} (\nu_b)_b,\end{aligned} \tag{4.13}$$

are respectively the force and position control signals decoupled at the constraint surface frame $\bar{E}_{s1}$. The position control law is defined as:

$$(\nu_b)_b = \cancel{(\dot{p}_d)_b} + K_p (e_p)_b, \tag{4.14}$$

where $(\dot{p}_d)_b = 0$, since it is a position regulation phase, $(e_p)_b = T_1 - (\vec{p}_{be})_b(t)$, and $K_p = K_p^T > 0$ is a positive-definite matrix.

The force control law is defined as:

$$(\nu_f)_e = \cancel{(\dot{\vec{f}}_d)_e} + K_f (e_f)_e + K_{if} \int_{0^-}^{\infty} (e_f)_e, \tag{4.15}$$

where $(\dot{\vec{f}}_d)_e = 0$, since it is a force regulation phase, $(e_f)_e = (\vec{f}_d)_e - F_m$, $F_m = (\vec{f})_e(t)$ is the contact force measured in the frame $\bar{E}_e$ at the force sensor tip, $(\vec{f}_d)_e$ is the desired contact force for the first touch, and $K_f = K_f^T > 0, K_{if} = K_{if}^T > 0$ are diagonal positive-definite matrices. Notice that the contact force is considered to be the reaction force that the surface is exerting against the tool.

The orientation control law is defined as:

$$(\nu_o)_e = \cancel{(\omega_d)_e} + K_o e_\phi, \tag{4.16}$$

where $(\omega_d)_e = 0$, since the orientation is being regulated, $K_o = K_o^T > 0$ is a diagonal positive-definite matrix, $(\omega_d)_e$, and e_ϕ is the orientation error (see Section 2.6.4.2), defined using the *body frame approach* in terms of the quaternion formulation as follows:

$$e_\phi = \eta \epsilon_d - \eta_d \epsilon + \hat{\epsilon}_d \epsilon, \tag{4.17}$$

where $Q_d = (\eta_d, \epsilon_d)$ is the quaternion representation of the desired pose $R_d = R_{bs}$, and $Q = (\eta, \epsilon)$ is the quaternion representation of the current pose $R = R_{be}(t)$. Once again, since the body frame approach is already being used, the orientation error e_ϕ is already represented in the tool frame.

When the position, force and orientation errors are all zero, this phase is completed and the next phase is triggered. Since there is a compliant structure between the tool and the surface along $\vec{z}_e$, the real end-effector position will be $\tilde{H}_1 \neq H_1$, which can be measured via forward kinematics.

4.4.1.4 Phase A1.4: Local surface re-estimation

This phase should be triggered *only* if the surface orientation R_{bs1} is considered to be slightly different from the actual. Normally, as mentioned in Section 3.2.1, we may consider the surface orientation as known.

However, small deviations from the original calibration may accumulate with repeated operations of the DORIS robot. This situation is very undesired, since an erroneous value of R_{bs1} turns impossible the end-effector to be adequately posed perpendicular to the surface. Further, if R_{bs1} is significantly deviated from its original value, the end-effector can even touch the surface outside the cone of friction, which leads it to slide indefinitely. As discussed in Section 2.7.9, two different techniques can be used for this case to estimated $\tilde{R}_{bs1}$ from R_{bs1}.

In the *first approach* (Section 2.7.9.1), the end-effector performs a *small displacement* Δ over the surface of interest. For this, an arbitrary small tangent vector $\Delta_b = \delta u_t, \delta > 0$ is defined, where u_t is an unit tangent vector defined from an arbitrary composition of x, y components of the original matrix $R_{bs1} = \begin{bmatrix}(\vec{x}_{s1})_b & (\vec{y}_{s1})_b & (\vec{z}_{s1})_b\end{bmatrix}$, i.e.:

$$u_t = \frac{c}{\sqrt{c^T c}} \quad c = \alpha(\vec{x}_{s1})_b + \beta(\vec{y}_{s1})_b, \quad \alpha, \beta > 0. \tag{4.18}$$

Hence, the end-effector will be controlled towards the point $M_1 = T_1 + \Delta_b$. The actual displacement will be measured from the forward kinematics as $\tilde{\Delta}_b = \tilde{M}_1 - \tilde{T}_1$, where $\tilde{T}_1$ is the last tool position (in frame $\bar{E}_b$) before the execution of this phase, and $\tilde{M}_1$ is the tool position (in frame $\bar{E}_b$) after the tool displacement is completed. Then, we define the estimated tangent unit vector as:

$$t = \frac{\tilde{\Delta}_b}{\sqrt{\tilde{\Delta}_b^T \tilde{\Delta}_b}}. \tag{4.19}$$

The contact normal force is estimated by:

$$(\vec{f}_n)_b = \left(I - \frac{tt^T}{\sqrt{t^T t}}\right)(\vec{f}_c)_b, \tag{4.20}$$

where $(\vec{f}_c)_b = R_{be}(\vec{f}_c)_e$ is the contact force sensed at the tool tip and transformed to the frame $\bar{E}_b$. Then, the estimated surface is given by:

$$\tilde{R}_{bs1} = \begin{bmatrix} t & \hat{t}\frac{(\vec{f}_n)_b}{\sqrt{(\vec{f}_n)_b^T(\vec{f}_n)_b}} & -\frac{(\vec{f}_n)_b}{\sqrt{(\vec{f}_n)_b^T(\vec{f}_n)_b}} \end{bmatrix}. \tag{4.21}$$

In the *second approach* (Section 2.7.9.2), the end-effector touches *three non-collinear points* over the surface. One solution for this is to control the tool position to three arbitrary points A, B, C over the original surface $\bar{E}_s$, which can be deployed over a circumference of radius δ around T_1 and separated by 120° offset. To do this, firstly, we arbitrarily define an unit vector:

$$t_1 = \delta \frac{(\vec{x}_s)_b}{\sqrt{(\vec{x}_s)_b^T (\vec{x}_s)_b}}. \tag{4.22}$$

Then, we define the rotated vectors around k as $t_2 = R_k(2\pi/3)t_1$ and $t_3 = R_k(2\pi/3)t_2$, where $k = (\vec{z}_s)_b$. Hence, we finally define $A = T_1 + t_1$, $B = T_1 + t_2$ and $C = T_1 + t_3$.

Actually, the tool will be positioned in $\tilde{A}, \tilde{B}, \tilde{C}$ when controlled to A, B, C respectively, and these points are measured via forward kinematics. Then, we define the vectors $v = \tilde{B} - \tilde{A}$, $w = \tilde{C} - \tilde{A}$ and $n = \hat{v}w$. Then, the estimated surface is given by:

$$\tilde{R}_{bs1} = \begin{bmatrix} \frac{v}{\sqrt{v^T v}} & -\widehat{\left(\frac{v}{\sqrt{v^T v}}\right)}(\vec{z}_e)_b & sign\left[n^T (\vec{z}_e)_b\right] \frac{n}{\sqrt{n^T n}} \end{bmatrix}. \tag{4.23}$$

The general control scheme is the same from the previous phase, except that the position set-points are changed, as well as the desired contact force, which can be different from the one expected for the first touch. Generally, a greater force is desired for the task over the surface, and a smaller force is expected for the first touch.

4.4.1.5 Phase A1.5: Regulation at standstill pose

The end-effector remains in a constant pose (standstill) at $\tilde{T}_1$ and perpendicularly positioned with respect to the surface of interest. The hybrid force/position control laws are the same as those used in phases A1.3 and A1.4, except for the force reference, that can be changed as desired.

Although the orientation control scheme is also the same, the orientation error depends if the normal vector of the surface at the contact point was:

(i) Re-estimated, i.e., phase A1.4 was performed, and $R_d = \tilde{R}_{bs1}$.

(ii) Not changed, i.e., phase A1.4 was skipped, and $R_d = R_{bs1}$.

Notice that, when $R_{be} \to R_d$, the contact tangent/friction forces (F_x, F_y) are cancelled, so that there is only a contact force in z-axis, and hence, the end-effector pose is assumed to be perpendicular. This phase lasts until the vibration measuring is arbitrarily stated as concluded. Once finished, the manipulator is retracted to its initial pose, which procedure is described by task A3 (Section 4.4.3).

4.4.2 Task A2: *Touchscreen* interaction

In this task, a special fingertip for *touchscreens* is attached to the end-effector and posed in fully contact with a nearby surface. In *touchscreen* devices, we can experience a variety of functionalities and interface patterns, such as button pressing and finger sliding. Once in contact with the screen, the manipulator finger should follow a preestablished reference path. For this, the following phases are defined:

4.4.2.1 Phases A2.1, A2.2 and A2.3

These phases works exactly the same as A1.1, A1.2 and A1.3. However, after the execution of phase A2.3 (approach), the end-effector cannot perform *surface normal re-estimation* (A1.4), the way it is done in vibration inspection. This is because phase A1.4 requires the end-effector to perform arbitrary small movements over the surface, which is not allowed in *touchscreen* tasks, since the finger motion over it is defined by another preestablished path.

4.4.2.2 Phase A2.4: Tracking path over surface/ online re-estimation

Once the previous phases are finished, the tool follows a reference path $p_d = [\vec{p}_{be1'}(t)]_b \in \gamma_1$ (Section 4.3.2.2). Similarly to phase A1.5, this tracking is performed so that the end-effector keeps itself always perpendicularly with respect to the surface.

Unlike the vibration inspection phase A1.5, which needs to force the tool to remain at rest (standstill), phase A2.4 permits the tracking of a trajectory where $\dot{p}_d \neq 0$. This allows the surface normal vector to be online estimated during the trajectory following using the infinitesimal displacement technique (Section 2.7.9.1).

To achieve this, the robotic system shall measure at each iteration of the control algorithm the displacement $\tilde{\Delta} = (\vec{p}_{be})_b[(k+1)T] - (\vec{p}_{be})_b[kT]$, where $k \in \mathbb{N}$ is non-positive integer, and T is the period of the system iteration. If $\tilde{\delta} = \left\|\tilde{\Delta}\right\| > \tilde{\delta}_{tol}$, being $\tilde{\delta}_{tol}$ an arbitrary and sufficient small tolerance, the system understands that a new estimation is required. Notice that $\tilde{\delta}$ is considered as a tangent vector along the real surface.

Similarly to the algorithm presented in phase A1.4 (see 4.4.1.4), we define the following variables:

$$t = \frac{\tilde{\Delta}}{\sqrt{\tilde{\Delta}^T\tilde{\Delta}}}, \quad (\vec{f_n})_b = \left(I - \frac{tt^T}{\sqrt{t^Tt}}\right)(\vec{f_c})_b, \quad (\vec{f_c})_b = R_{be}(\vec{f_c})_e, \tag{4.24}$$

wherein $(\vec{f_c})_b = R_{be}(\vec{f_c})_e$ is the contact force sensed at the tool tip and expressed in the frame $\bar{E}_b$. Then, the estimated surface orientation is given by:

$$\tilde{R}_{bs1} = \begin{bmatrix} t & \hat{t}\frac{(\vec{f_n})_b}{\sqrt{(\vec{f_n})_b^T(\vec{f_n})_b}} & -\frac{(\vec{f_n})_b}{\sqrt{(\vec{f_n})_b^T(\vec{f_n})_b}} \end{bmatrix}. \tag{4.25}$$

Once this phase finished, the manipulator is retracted to its initial pose, which procedure is described by task A3 (Section 4.4.3).

4.4.3 Task A3: Retraction

This task is automatically triggered after any manual or autonomous scheduled task, and it is divided into the following phases:

4.4.3.1 Phase A3.1: Reverse path tracking

The manipulator is automatically controlled so that the end-effector frame $\bar{E}_e$ follows exactly the opposite path $[\vec{p}_{be1'}(t)]_b \in \mathbb{R}^3$ (Section 4.3.2.4), which will take it safely from the contact surface towards the retracted pose, in which all joint variables are zero, i.e, $q = (0, 0, 0, 0, 0)$. If the previous task has been A2 (*touchscreen*), then, the end-effector current position may be different from the one in which it touched the surface for the first time, which was also the end of path $[\vec{p}_{be1'}(t)]_b$. Hence, in this case, it should first perform a kinematic control towards H_1 before starting the retraction.

4.4.3.2 Phase A3.2: Nest

As seen in Section 3.2.6, the arm cannot only achieve the *zero pose* and then be released. When it is near the final position, the manipulator should be controlled to a slightly different pose so that, when released, it ties at the hook and do not drop indefinitely.

This end-effector pose, as explained in Section 4.4.1.2, is the manipulator *nest* in which the joint variables q are considered to be known. Once at the nest, the joint angular positions are reset to zero, and the manipulator is calibrated consequently. This procedure is necessary for the operation performed with TETIS manipulator, since its joints are not equipped with absolute encoders[1]. For simplicity, in this work,

[1]Relative incremental encoders provide the motor position, but they are reset when the system powers down. Hence, once the joints are re-powered, their position data are unknown. This problem does not occur in absolute encoders, although they are very expensive.

the *nest pose* will be considered as the retracted pose $q = (0, 0, 0, 0, 0)$ depicted in Section 3.4.1.

As mentioned in Section 4.4.1.2, some commercial manipulators already have a pre-programmed routine to place the robot in the nest pose from the ready pose. In this case, phase A3.2 should control the robot to the ready pose.

4.5 Replacement of Jacobian pseudo-inverse by filtered inverse

As discussed in Section 2.8, the utilization of the FI method has several alleged motivations over the classic Jacobian pseudo-inverse, especially:

- Ability to pass near (or even over) singular configurations, providing good computational performance and accuracy near unfeasible solutions.
- Definition of additional constraint problem, which consists on an objective function to be optimized together with the robot control objectives.

The utilization of the Filtered Inverse solution instead of the Jacobian pseudo-inverse matrix in TETIS operation phases is particularly interesting for the second reason, since the maximization of additional objectives using the classic method (Section 2.6.5) requires redundant DoFs. However, as shown in Chapter 3, the redundancy characteristic cannot be exploited by using the TETIS manipulator since it has only 5-DoFs. On the other hand, according to Vargas (2013), there are no stated limitations on the number of DoF for a given robot in order to apply the FI method to achieve distinct simultaneous objectives.

Other great concern is about the robot internal singularities (Siciliano et al. 2009), mainly during the movement from the nest to the ready poses (as explained in Section 4.4.1.2), that can demand a high energy consumption, especially during the patrolling tasks along the rails, in which DORIS performs successive arm extensions and retractions. Analogously to a car, it is the same situation of taking the car successive times out of inertia, which demands much more fuel consumption if compared to a car already in fast movement. As motivated in Section 1.2, power consumption is very limited for mobile vehicles, hence, it is valid to investigate the application of the Filtered Inverse method (Vargas 2013) to control the pose of theTETIS manipulator and propose/test an energy cost function.

4.5.1 Challenges for the original structure

As seen in Section 3.4.2, our original kinematic structure has many peculiarities that were not observed and taken into account in the first case studies carried out for the Filtered Inverse method (Vargas 2013). Indeed, only planar and anthropomorphic kinematic structures were investigated.. Thus, it is not clear that this method will work as expected for the TETIS manipulator, which also counts with a prismatic joint, that was also not tested yet. In addition, a choice for the robot and filter initial condition $q(0)$ and $\Theta(0)$ may be a challenging task, since the robot has many unknown internal singularities. As discussed before, the uniqueness and convergence rate of Θ strongly depends on the initial condition $\Theta(0)$ and how far the robot is initially from a singular configuration.

4.5.2 Energy consumption

Energy consumption is one of the most concerning issues in autonomy of mobile robots, since their power supply are limited. Some recent studies, such as Vergnano et al. (2012), already approaches algorithms for optimization of power consumption in many robot systems.

For actuators without self-lock mechanism, the consumed power of a manipulator is directly related to the velocity and the torque τ provided by its joints, which can be calculated by the *robot dynamics* as follows (Siciliano et al. 2009):

$$B(q)\ddot{q} + C(q,\dot{q})\dot{q} + F_v(q)\dot{q} + F_s sign(\dot{q})G(q) = \tau - J^T(q)h_{ext}, \tag{4.26}$$

where $B \in \mathbb{R}^{n\times n}$ is the *manipulator inertia matrix*, $C \in \mathbb{R}^{n\times n}$ is the *matrix of Coriolis and centripetal forces*, $G \in \mathbb{R}^{n\times 1}$ is the *gravity vector*, $F_v \in \mathbb{R}^{n\times n}$ denotes a diagonal *matrix of viscous friction coefficients*, $F_s \in \mathbb{R}^{n\times n}$ denotes a diagonal *matrix of static Coulomb friction torques* (simplified model), and $h_{ext} \in \mathbb{R}^{6\times 1}$ is the vector of forces and torques exerted by the end-effector on the environment.

However, this is a very complicated equation to be considered for an initial investigation of a cost function proportionally related to energy, since it involves the complete computation of the manipulator dynamics and the objective function presented in Section 2.8.5 depends only of q, while (4.26) depends of $q, \dot{q}, \ddot{q}$. Hence, it is valid to propose a simpler approach.

4.5.2.1 Moving away high-torque configurations

One strategy would be to keep the manipulator far from those configurations that clearly demand more energy from its more powerful joints. In the case of the original kinematic structure of the TETIS manipulator (5-DoF, Figure 3.32), by the rough analysis of its and considering the gravity vector pointing downwards (same direction of $-\vec{z}_b$), we can see that:

- The 3^{rd} joint may demand more energy the more close this joint is to $q_3 = 0, \pi$, since the torque provided by the next links would be higher.
- Analogously, if the 3^{rd} joint is released downwards ($q_3 \approx -\pi/2$), the 4^{th} joint must be as far as possible from $q_4 = \pm\pi/2$.

Hence, a possible cost function involving q_3 and q_4 would be:

$$f(q_3, q_4) = \alpha_3(q_3 + \pi/2)^{2k_1} + \alpha_4 q_3^{2k_2}(q_3 + \pi)^{2k_3} q_4^{2k_4}, \tag{4.27}$$

which is very similar to the joint limit function (see Section 2.8.6), and its parameters $\alpha_i, k_i > 0$ depend on the robot dimensions and other specific objectives. Notice that, if we consider the modified 6-DoF structure in Figure 3.34b, the cost function would be the same.

4.5.3 Potential energy

Another possible and more comprehensive index for measuring energy would be via calculation of the net potential energy, which depends only on the joint variables q. According to Siciliano et al. (2009), the net potential energy of a n-joint manipulator structure is given by:

$$\mathcal{U} = -\sum_{i=1}^{n}(m_i g_b^T p_i + \bar{m}_i g_b^T \bar{p}_i), \tag{4.28}$$

where g_b is the gravity acceleration vector represented in an arbitrary frame $\bar{E}$, m_i and $\bar{m}_i$ are the total mass of link i and joint i, respectively, and p_i and $\bar{p}_i$ are the distance between the center-of-mass of the i^{th} link and joint actuator, respectively, both represented in $\bar{E}$.

For simplicity, to calculate the net potential energy $\mathcal{U}$ of our structure, let us consider the center-of-mass of each link at its middle point, the "M_i" offsets equal to zero, and $g_b = (0, 0, -g), g \approx 9.8$. In the case of TETIS original structure (5-DoF, Figure 3.32), the joints and links masses are given by $\bar{m}_2 = \bar{m}_3 = 620, \bar{m}_4 = \bar{m}_5 =$

400, $m_2 = 75, m_3 = 202, m_4 = 157, m_5 = 75$ (all in g, according to A.1), and the objective function which we want to minimize is given by:

$$\begin{aligned} f = \mathcal{U} = g\, \big[&-m_2 \tfrac{E_2}{2} + m_3 \left(\tfrac{E_3 s_3}{2} - E_2\right) - \bar{m}_3 E_2 + m_4 \left(\tfrac{-E_4 c_{34}}{2} + E_3 s_3 - E_2\right) + \\ &\bar{m}_4 \left(E_3 s_3 - E_2\right) + m_5 \left(\tfrac{E_5 s_{34} c_5}{2} - E_4 s_{34} + E_3 s_3 - E_2\right) + \\ &\bar{m}_5 \left(-E_4 s_{34} + E_3 s_3 - E_2\right) \big] . \end{aligned} \tag{4.29}$$

In case of the modified 6-DoF structure of Figure 3.34b, the joints and links masses are given by $\bar{m}_2 = \bar{m}_3 = \bar{m}_4 = 620, \bar{m}_5 = \bar{m}_6 = 400, m_2 = 75, m_3 = 101, m_4 = 101, m_6 = 157$ (all in g), and the objective function which we want to minimize is given by:

$$\begin{aligned} f = \mathcal{U} = g\, \big\{ &-\tfrac{m_2 E_2}{2} + m_3 \left(\tfrac{E_3 s_3}{2} - E_2\right) - \bar{m}_3 E_2 + m_4 \left(\tfrac{E_4 s_3 c_4}{2} + E_3 s_3 - E_2\right) + \\ &\bar{m}_4 \left(E_3 s_3 - E_2\right) + \bar{m}_5 \left(E_4 s_3 c_4 + E_3 s_3 - E_2\right) + \\ &m_6 \left[\tfrac{E_6}{2} \left(s_3 c_4 s_5 c_6 + c_3 c_5 c_6 + s_3 s_4 s_6\right) + E_4 s_3 c_4\right] + \\ &\bar{m}_6 \left(E_4 s_3 c_4 + E_3 s_3 - E_2\right) \big\} . \end{aligned} \tag{4.30}$$

Chapter 5

Simulation Results

In this chapter, we present numerical simulations with the mathematical model of the DORIS manipulator performing interaction tasks on contact surfaces with known and uncertain geometries. The results are shown to illustrate the performance and the feasibility of the proposed control scheme. In particular, this chapter also presents the preliminary simulations of FI for optimizing additional system constraints.

As discussed in Section 1.2, simulations in computing environment are extremely important to test the behaviour of the manipulator control and operation, as well to tune the control parameters before a field test. Simulations can aid the robot designer to identify possible project flaws, and to propose modifications to fix them or improve the robot functionalities. In addition, computational simulations may warn the operator about possible security violations during the robot operation, which is of critical concern at a real platform environment.

Since the DORIS manipulator is still at construction phase (Figure 5.1), this chapter aims to present preliminary simulations of the end-effector pose control to detect significant levels of vibration.

5.1 Simulation environment

In this work, we employ the Matlab/Simulink (The MathWorks, Inc.) as a programming and development environment for the computational simulations.

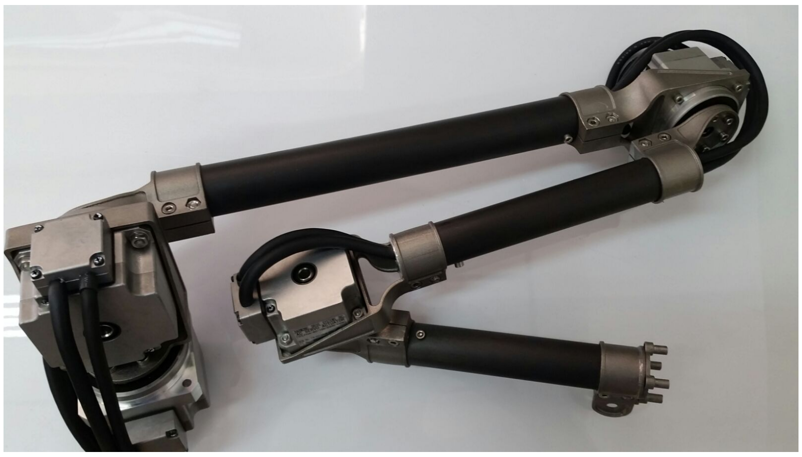


Figure 5.1: Manipulator TETIS under construction at a bench test.

5.1.1 Robot environment

There are several approaches to model the robot kinematics (and dynamics) in Matlab/Simulink. One established package is the Robotics Toolbox (Corke 2011), which includes important functionalities, such as: (i) modeling of kinematic structure via DH parameters; (ii) modeling of robot dynamics; (iii) computation of geometric and analytical Jacobian (in world or tool frame); (iv) computation of manipulability; (v) simple graphic plot of the robot structure; (vi) ready-to-use blocks for Simulink.

Another important subject of concern for manipulator simulations is the 3D visualization. The most recent version of the Robotics Toolbox offers a complete 3D plot of the robot structure, but this functionality demands great memory allocation and it is still unstable for running in most computer configurations. Vargas (2013) proposes a 3D modeling based on two Matlab basic functions: *patch* and *makehgtform*.

In this work, we will illustrate the robot operation with Matlab/Simulink *plot function*, which will interact with a flat unlimited surface described as:

$$\gamma = \left\{ P = (x, y, z) \in \mathbb{R}^3,\ y = 400 \right\}. \tag{5.1}$$

5.2 Force sensor model

In *Simulink* standard package, the SimMechanics package provide some ready-to-use blocks which permit the utilization of different soft contact and friction models, such as those studied by Bibalan & Featherstone (2009). Since these blocks comprise excessive complexity for our application, we must investigate a simple mathematical

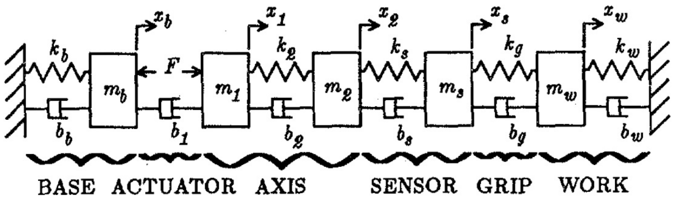


Figure 5.2: Mass-spring-damper model from Craig & Raibert (1979).

description of the force sensor to be used in the simulation environment of DORIS manipulator.

To model a force sensor, we should first understand the mechanics of the end-effector interaction with the surface to be touched. Bonfadini (2001) and Leite (2005) propose a configuration in which the force sensor is represented by a mass-spring-damper system placed at the end-effector tip. A similar system was also assigned to the contact surface, even though it is considered to be rigid in most of practical cases. In their proposal, it is assumed that the sensor can approach/touch the surface only in the perpendicular direction to it. Moreover, it is assumed that only the perpendicular force (F_z) can be sensed, while F_x and F_y are estimated via noise injection.

Other approaches, such as Eppinger & Seering (1987), Craig & Raibert (1979), propose more comprehensive models that consider the mass-spring-damping effect between each system part (base, actuator, sensor, grip and environment, see Figure 5.2) and also an investigation of the force control cases effect in the closed-loop bandwidth. This model, however, is limited to the direction of only one axis. On the other hand, there are recent approaches, such as from Luo et al. (2007), that provides a complete modeling of a six-axis force/moment sensor for robot finger. Such models are more suitable to be used in simulations with robots whose dynamics is fully known a priori.

In this work, we propose a simplified model based on the models presented in Bonfadini (2001) and Leite (2005)), in which the spring deformation occurs only along z-axis, but the sensor can approach and touch the surface in any direction and the forces F_x, F_y are more reliably estimated by the calculation of the friction and normal forces. Although compliance effects can be useful in several cases, to simplify our simulations, we suppose that our contact surface is perfectly rigid, i.e., k_s (surface spring constant) and b_s (surface viscous damping coefficient) tend to infinity.

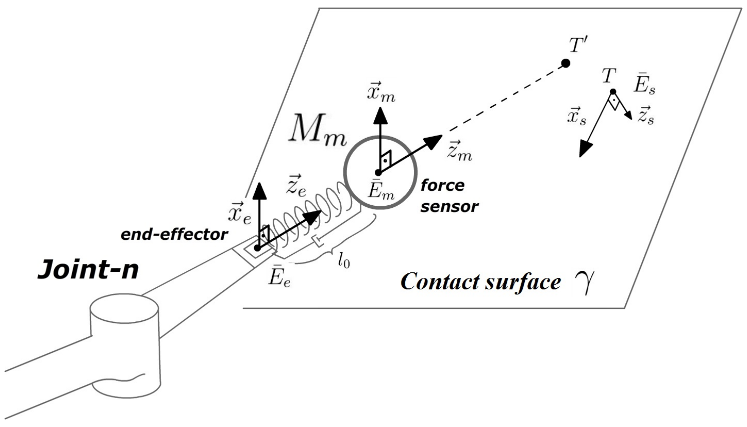


Figure 5.3: Force sensor dynamic model for simulation - Overview of coordinate frames.

5.2.1 Initial assumptions

As illustrated in Figure 5.3, the dynamic model of the force sensor considers that:

- The load cell is a mass-spring system with mass M_m, and l_0 is the spring length when it is in fully relaxed state.
- $\bar{E}_e$ is the end-effector frame, $\bar{E}_m$ is the force sensor frame, and $\bar{E}_s$ is the the frame of the contact surface γ.
- The origin of the frame $\bar{E}_m$ is located at the sensor mass. Also, $\bar{E}_m$ is aligned with $\bar{E}_e$ and can only move in the direction of $\pm\vec{z}_e$.
- $\vec{z}_s$ is perpendicular to the surface and points towards its *interior*. In other words, both $\vec{z}_e$ and $\vec{z}_m$ form an acute angle with $\vec{z}_s$, i.e., $\vec{z}_e, m \cdot \vec{z}_s > 0$.
- $\vec{x}_s$ and $\vec{y}_s$ are arbitrarily defined.
- The sensor is displaced from the end-effector by l_0 in the z direction when the sensor spring is fully relaxed. Thereby, one spring end is located exactly at $\bar{E}_e$ origin, and the other end is located at the sensor mass ($\bar{E}_m$ origin).
- T is the *touch set point*, i.e., the point over the surface γ where the end-effector is supposed to reach first. T' is the *touch projected point*, i.e., the point where an imaginary straight line starting from the end-effector tip crosses the surface γ. In practice, T and T' are very close. Therefore, it is considered that these two points belong to the same surface, which is locally flat.

Now, looking at the system from another perspective as in Figure 5.4:

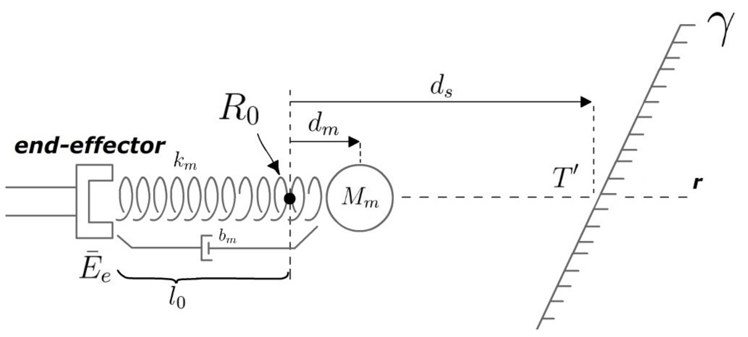


Figure 5.4: Force sensor dynamic model for simulation - References.

- k_m is the spring constant of the sensor. It is assumed that the sensor has no damping effect.
- Define an imaginary straight line r starting from the end-effector/sensor towards the surface γ, reaching it at T'. Set R_0 as the point in which the spring is fully relaxed. This point acts as a *zero* reference along r and the point where the distance to the end-effector is exactly l_0.
- Define d_s as the distance between R_0 and T', and d_m as the distance between R_0 and the origin of the frame $\bar{E}_m$.
- It is assumed that the spring can only be compressed along r.
- It is worth highlighting: $d_s > 0$ indicates that the sensor has not touched the surface yet. Moreover, if $d_m = 0$, the sensor spring is at its relaxed state.
- The end-effector orientation with respect to the surface is within the static friction cone (see Section 2.7.3).

5.2.2 Extracting the force measurement

Two possible situations can occur: (i) the sensor is not touching the surface (i.e., $d_s > 0$); (ii) the sensor is touching the surface (i.e., $d_s = d_m$). When the sensor is in contact, d_m assumes a non-zero value, which indicates that the spring is compressed. Thus, we can estimate the reaction force that the surface applies on the end-effector by $F_c = k_m d_m$.

However, the direction of this force depends on the direction that the end-effector is pushing against the surface. This direction can be calculated by the end-effector position velocity. Since this vector can point either towards (approaching) or backwards (moving away) the contact surface, we must concern about both situations.

If the end-effector is attempting to approach, $\vec{v}_e$ point towards the surface, i.e., $\vec{v}_e \cdot \vec{z}_s > 0$. Then, the contact force is estimated by $\vec{F}_c = k_m d_m \frac{\vec{v}_e}{\|\vec{v}_e\|}$. Otherwise, if the end-effector is withdrawing the its tip away, then $\vec{v}_e$ point away from the surface ($\vec{v}_e \cdot \vec{z}_s < 0$), and $\vec{F}_c = -k_m d_m \frac{\vec{v}_e}{\|\vec{v}_e\|}$. Notice that the reaction force applied by the surface against the end-effector is equal in magnitude and opposite in direction as the force that the end-effector is setting over the surface, since we are assuming that the touching procedure occurs within the friction cone.

The *force measurement* that the real sensor provide to the robotic system is actually the contact force of the surface over the sensor represented in the end-effector frame $\bar{E}_e$. It can be calculated as follows:

$$(\vec{F}_c)_e = k_m d_m R_{eb} \frac{(\vec{v}_e)_b}{\|(\vec{v}_e)_b\|}, \text{ if } (\vec{v}_e)_b^T (\vec{z}_s)_b > 0, \tag{5.2}$$

$$(\vec{F}_c)_e = -k_m d_m R_{eb} \frac{(\vec{v}_e)_b}{\|(\vec{v}_e)_b\|}, \text{ if } (\vec{v}_e)_b^T (\vec{z}_s)_b < 0. \tag{5.3}$$

5.3 Simulations of manipulator tasks (with Jacobian pseudo-inverse)

This Section presents simulation results that illustrate - without concerning about any obstacles, joint limits or other control objectives - the manipulator tasks A1 and A2. For those simulations, the geometric Jacobian pseudo-inverse will be utilized in the control strategy. The default simulation parameters were set as follows:

- Force sensor parameters:
  - Spring constant $k_m = 0.93\ N/mm^2$
  - Sensor offset with respect to end-effector frame $l_0 = 5\ mm$
- Surface:
  - Flat plane equation $ax + by + cz + d = 0 \Rightarrow a = c = 0,\ b = 1,\ d = -400$
  - Surface rotation with respect to base $R_{bs} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$
- Control gains:

  - Path tracking phase - Position proportional $K_{pp,p} = 1.7$
  - Path tracking phase - Orient. proportional $K_{po,p} = 20$
  - Approach/regulation - Position proportional $K_{pp,a} = 1.7$
  - Approach/regulation phase - Force proportional $K_{pf,a} = 40$
  - Approach/regulation - Force integral $K_{if,a} = 0.4$
  - Approach/regulation - Orient. proportional $K_{po,a} = 20$

- Tolerances for error norms:
  - Position error $tol_p = 1\ mm$
  - Orientation error (quaternion error) $tol_o = 0.044$ ($0.5°$ only around $\vec{x}_e$ and $\vec{y}_e$)
  - Force error $tol_f = 0.01\ N$ (is only used to trigger the switch from approach to regulation phase)

- Goals:
  - Distance between between the surface and the approach point $\delta = 100\ mm$
  - Set-point (in mm) for regulation phase $T = (10, 400, -50)$
  - Set-point (in mm) for path tracking phase $H = T - \delta(\vec{z}_s)_b = (10, 300, -50)$
  - Desired orientation at H and $T \Rightarrow R_d = R_{bs}$
  - Desired force (in N) in $\bar{E}_e$ at first touch (end of approach phase) $F_{d,a} = (0, 0, -1)$
  - Desired force (in N) in $\bar{E}_e$ at the regulation phase $F_{d,t} = (0, 0, -2.5)^T$

- Joint space initial configuration $q(0) = (0, 120°, -60°, -120°, 120°)^T$
- Simulation time $t \in [0, 100]$ seconds
- Simulation step size $T = 0.01$ seconds

The desired force set-points were arbitrarily defined by roughly considering what would be a soft contact force. The optimal force set-points strongly depend on the material of the contact surface and the maximum allowed force against the manipulator structure, which could be estimated in practical experiments. For a further and better visualization of the simulations overall results, a summary of the simulations parameters, purposes and utilized strategies is shown in Table 5.1.

Table 5.1: Summary - Simulations

Simulation	Sub-simulation	Purpose	Utilized Strategy
1	-	Illustration of vibration inspection	Jacobian pseudo-inverse
2	2A	Estimation of certain surface	Jacob. pseudo-inverse/3-points method
	2B	Estimation of uncertain surface	Jacob. pseudo-inverse/3-points method
3	-	Follow path over *touchscreen*	Jacobian pseudo-inverse
4	4A,4B,4C,4D	Test FI method in Matlab	FI method
5	-	*Nest* to *ready* poses	FI method
6	-	*Ready* to *approach* poses	FI method
7	-	Obj. func. of joint limit avoidance	FI method / Aug. Jacob.
8	8A	Obj. func. of energy consumption	FI/Aug. Jacob./Avoid high torque config.
	8B		FI/Aug. Jacob./Min. potential energy

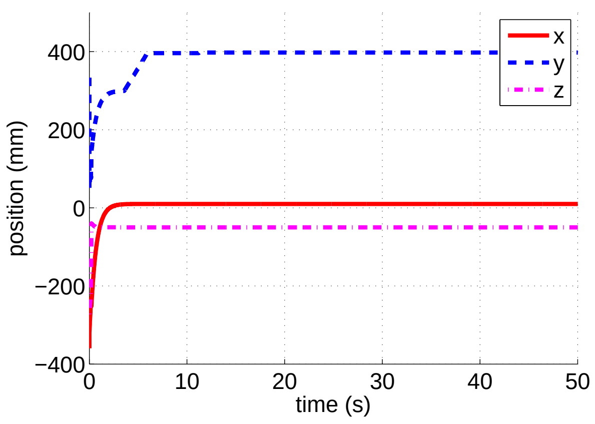
5.3.1 Simulation 1: Vibration inspection

This simulation goal is to illustrate task A1 of vibration inspection (Section 4.4.1), in which the robot end-effector is controlled for regulation over a single point over the surface of interest.

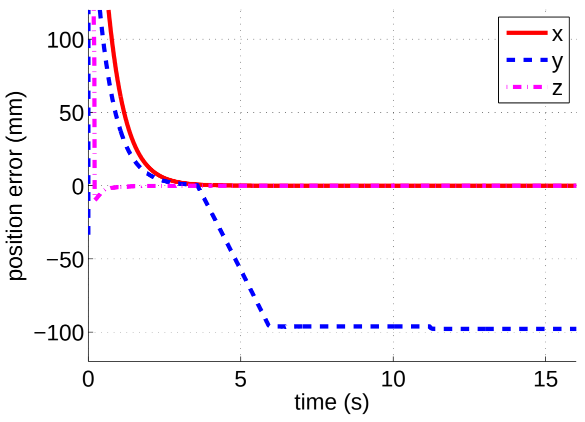
Figures 5.5a, 5.5b and 5.6a show the position of the robot tool with respect to base and the position error, while Figure 5.6b shows the tool orientation error in quaternion representation. As expected, at a first time, the tool was controlled to $H = (10, 300, -50)$, with no concerning about the path towards it. Notice that, once both position and orientation errors reach zero (about 3.57 s), the system switches the control to the *approach phase*. The orientation errors around x and y-axes are taken to zero, since the roll around z-axis is not controlled.

Notice in Figure 5.6a that the tool *approaches* towards $y = 400$, reaching it by $t \approx 6$ s. At a first time, the tool exerts 1 N force against the surface until the system evaluate that the force is controlled ($t \approx 11.2$ s), so that it can switch to the *regulation phase*, in which the tool exerts a 2.5 N force. Since the end-effector is displaced from the sensor by an offset and a spring, its y position with respect to the surface is slightly different from zero (about 3.9 mm in approach phase, and 2.3 N in regulation phase, where the exerted force is stronger).

Figure 5.7a shows the measured force in the tool frame, as well as the sensor spring displacement d_m and the tool distance to the surface d_s. We can check that force measurements only arise when $d_s < 0$, i.e., the tool is touching the

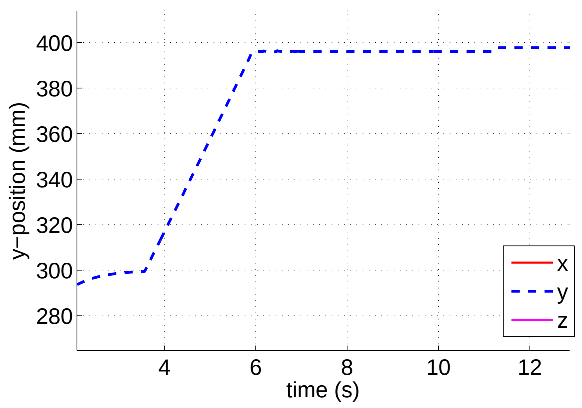


(a) End-effector position

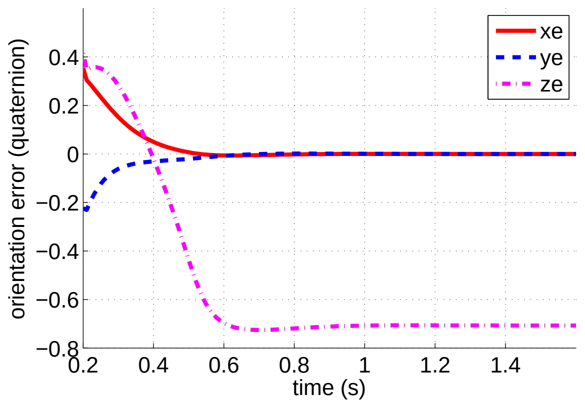


(b) Position error

Figure 5.5: Simulation 1 Plots - Vibration task.



(a) End-effector position (zoom)

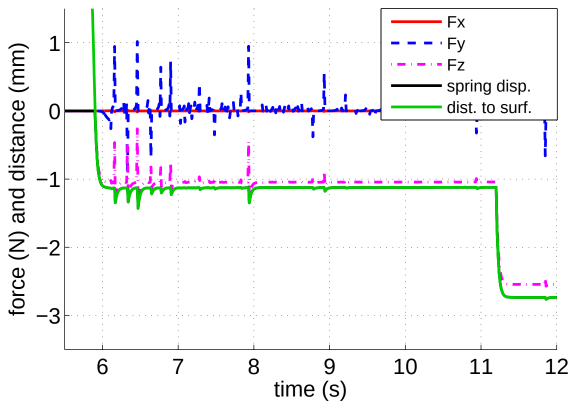


(b) Orientation error

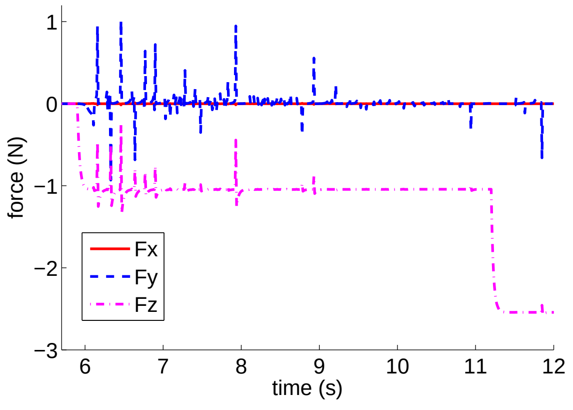
Figure 5.6: Simulation 1 Plots - Vibration task.

surface. Notice that the F_z measurement is slightly smaller than $d_m = d_s$, since $F_z = k_m d_m$ and we are considering $k_m = 0.93$. In Figure 5.7b, a close sight of the force components is shown. Notice that the F_z is the only component that changes significantly, since it is being controlled. In spite of the sensor noise, the components x, y behave about zero, which indicates that the tool is posed perpendicular with respect to $y = 400$, and, hence, $R_{be} \rightarrow R_{bs}$.

Figure 5.8a shows the force error with respect to time. Notice the presence of a small offset though. This is probably caused by the integral error accumulation in the force control, which was not properly reset when the approach phase was triggered. To circumvent this problem, the integral error is set to zero at the integrator port until the approach phase starts, avoiding, hence, a non-zero initial condition is this



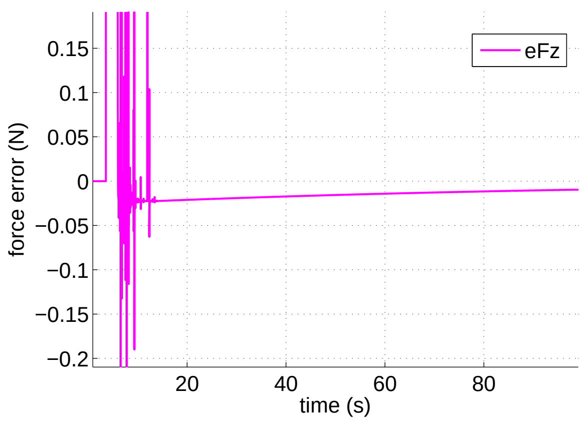
(a) Force sensor



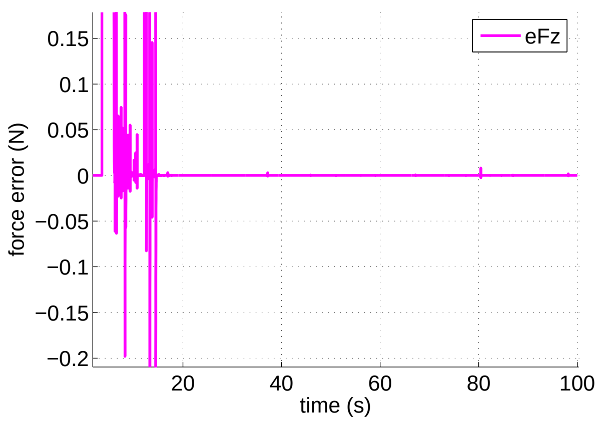
(b) Force sensor (zoom)

Figure 5.7: Simulation 1 Plots - Vibration task.

integrator by a kind of anti-reset windup (AWR) approach. As seen in Figure 5.8b, this problem was solved, since the force error is controlled to zero. Another simpler



(a) Force error with no reset



(b) Force error with reset

Figure 5.8: Simulation 1 Plots: (a) no reset; (b) integrator with anti-reset before the approach phase.

approach would be not to use integral force control, which would eliminate the error accumulation in such integrator. However, as discussed in Section 2.7.7, it is preferable to use PI force control for a faster response in position control and smaller offset in force control.

5.3.2 Simulation 2: Vibration task with surface re-estimation using three-point method

This simulation goal is to illustrate the three-point technique (Section 2.7.9.2) for re-estimation of the normal vector at the surface, whether it is considered to be slightly different from the previously known vector. This simulation is divided into two parts.

Firstly, we test the method for the case in which the desired orientation $R_d = R_{bs}$ is already correct. The simulation results will make us investigate how different will be the new estimation $\tilde{R}_{bs}$ from the original one. Different contact points will be evaluated in order to demonstrate that a more accurate estimation can be achieved if these points are more distant from each other. Secondly, we will apply the method for the case that $R_d \neq \tilde{R}_{bs}$, i.e., the new estimation should provide an orientation matrix of the frame $\bar{E}_s$ with respect to the frame $\bar{E}_b$ more accurate than what was known. It is worth mentioning that this simulation has used the same set of parameters adopted for the Simulation 1.

5.3.2.1 Simulation 2A: previously known surface

As discussed in Section 4.4.1.4, the selection of the three non-collinear point can be solved by choosing a triangular arrangement, in which the points are equally distant by δ from the touch point $T = (10, 400, -50)$ and an offset of 120° between each other. For $\delta = 50$, the given points are $A = (-40, 400, -50)$, $B = (35, 400, -6.7)$ and $C = (35, 400, -93.3)$.

Figure 5.9a shows the end-effector position in the x and y-axes, while Figure 5.9b shows the end-effector position in the y-axis.. Notice that the tool was taken to five different positions:

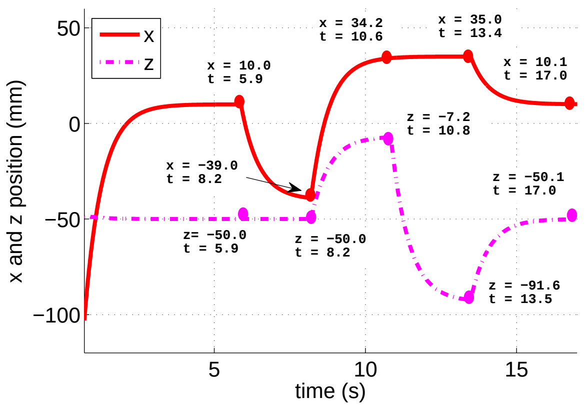
(i) First touch point $\tilde{T} \approx (10.0, 396.2, -50)mm$ in $t \approx 5.9s$

(ii) Point $\tilde{A} \approx (-39.0, 396.3, -50)mm$ in $t \approx 8.2s$

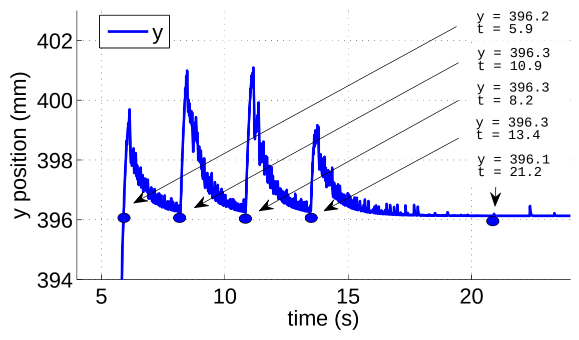
(iii) Point $\tilde{B} \approx (34.2, 396.3, -7.2)mm$ in $t \approx 10.9s$

(iv) Point $\tilde{C} \approx (35.0, 396.3, -91.6)mm$ in $t \approx 13.4s$

(v) again, the touch point $\tilde{T} \approx (10.1, 396.1, -50.1)mm$ in $t \approx 20s$



(a) Position (x,z)



(b) Position (y)

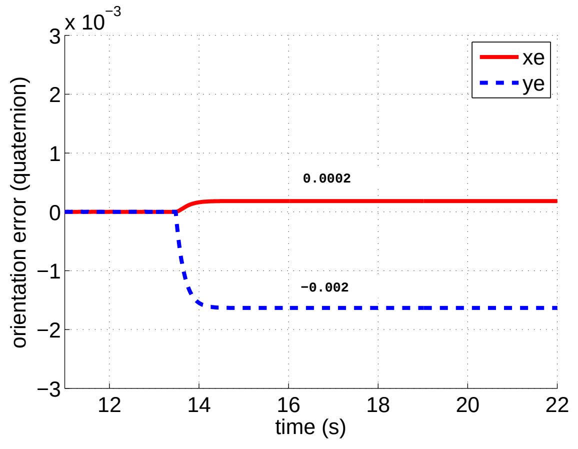
Figure 5.9: Simulation 2A Plots - Vibration task with re-estimation.

Notice that the y-axis position in noisier, since this direction of movement is the one that most interacts with the environment.

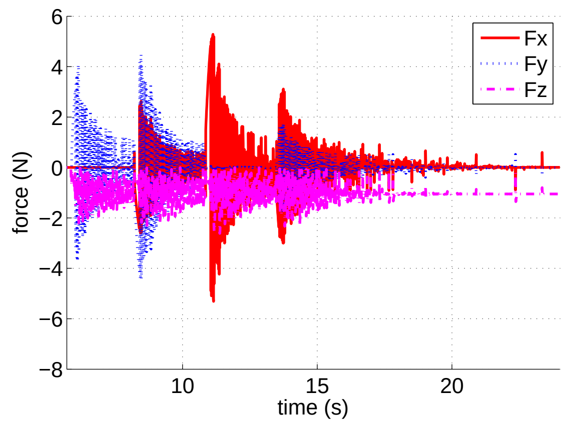
According to equation 4.23, this leads the to the following estimation for the normal vector:

$$\widetilde{(\vec{z}_s)}_b = \begin{bmatrix} -0.0020 & 0.9999 & 0.0026 \end{bmatrix}^T, \tag{5.4}$$

which is sufficiently close to the real value $(\vec{z}_s)_b = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$. The vectors $(\vec{x}_s)_b, (\vec{y}_s)_b$ do not matter for this estimation, since the surface orientation if defined only by $(\vec{z}_s)_b$.



(a) Orient. error (xe,ye)



(b) Force

Figure 5.10: Simulation 2A Plots - Vibration task with re-estimation.

Figure 5.10a shows the quaternion error of the orientation around $\vec{x}_e$ and $\vec{y}_e$ with respect to the original reference $R_d = R_{bs}$. Notice that the error is zero until about 13.5 s. From this instant, the robot starts to control the tool to a new desired orientation $R_d = \tilde{R}_{bs}$. As we can notice, the error around $\vec{x}_e$ and $\vec{y}_e$ are, respectively, 1.8×10^{-4} and -1.63×10^{-3} (or $0.02°$ and $-0.2°$), which are sufficiently small. Figure 5.10b shows the force measured at the tool in the three directions. Notice that the signal is noisier when the end-effector is moving towards a desired point. When the re-estimation is over, the force components in x and y-axes are taken again to zero with a small offset of 3×10^{-3} N and -2×10^{-3} N due to the estimation small inaccuracy, which indicates that the tool is sufficiently posed perpendicularly to the surface.

Let us now increase the distance between the contact points to $\delta = 150$. The new estimated normal vector was:

$$\widetilde{(\vec{z}_s)}_b = \begin{bmatrix} -0.0002 & 0.9999 & 0.0007 \end{bmatrix}^T, \tag{5.5}$$

and the orientation error around $\vec{x}_e, \vec{y}_e$ are $0.02°$ and $-0.04°$ respectively, which shows that, the more distant the three points are from each other, the more accurate is the new estimation, as expected. This method is suitable for use over the external surfaces of the machinery with little restriction about the contact points where the end-effector can be positioned.

5.3.2.2 Simulation 2B: uncertain surface

This simulation will reproduce the same algorithm in simulation 2A, but the tool will first be controlled to a slightly uncertain orientation $R_d \neq R_{bs}$, so that the new estimation fix this misdirection. Our goal is to illustrate the real situation in which the orientation of the surface, given by its normal vector, is assumed to be known is slightly different from the real one. If no re-estimation is performed, the tool would be not posed perpendicularly to the true normal vector of the surface.

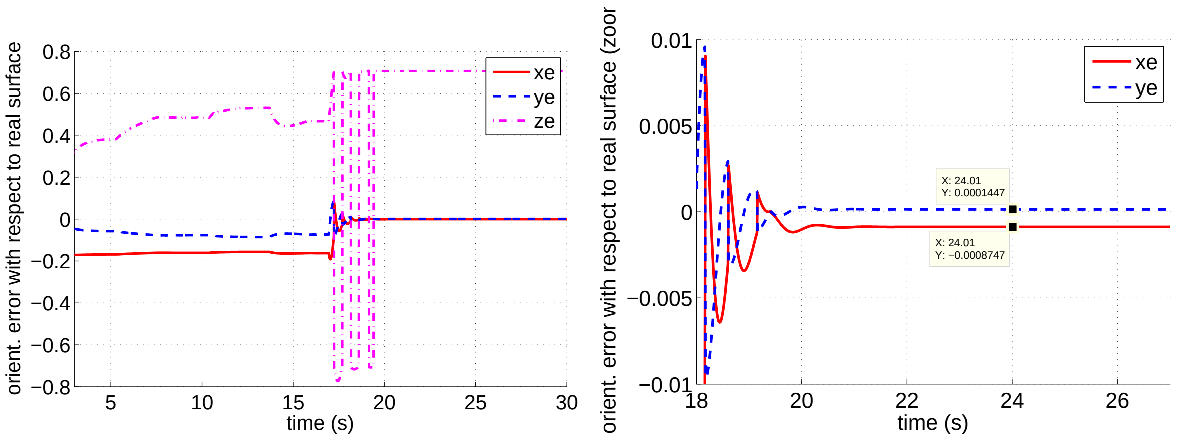
Let us consider the *real surface* again as $y = 400$. Yet, let us assume that the robotic system will first attempt to control the tool a slightly different surface defined by $0.02x + 0.8y - 0.3z = 398$. A rotation matrix compatible with this surface orientation would be:

$$R_d = R_{bs} = \begin{bmatrix} -0.9997 & 0.0088 & 0.0234 \\ 0.0250 & 0.3509 & 0.9361 \\ 0 & 0.9364 & -0.3510 \end{bmatrix}, \tag{5.6}$$

in which $(\vec{z}_s)_b = \begin{bmatrix} 0.0234 & 0.9361 & -0.3510 \end{bmatrix}^T$. Figure 5.11a shows the orientation error in tool frame with respect to the orientation of the original surface. Notice that an orientation errors around $\vec{x}_e$ and $\vec{y}_e$ exist until $t \approx 17\ s$, since the manipulator pose is being controlled to a wrong orientation. However, after that instant, we manipulator is controlled to an estimated orientation, and the orientation error tends to zero approximately. Figure 5.11b shows in more details that the final orientation errors around $\vec{x}_e$ and $\vec{y}_e$ are about -8.747×10^{-4} and 1.447×10^{-4} (or $-0.1°$ and $0.02°$), which are sufficiently small. Again, notice that the orientation about $\vec{z}_e$ axis is not controlled. The normal vector estimation is given by:

$$\widetilde{(\vec{z}_s)_b} = \begin{bmatrix} 0.0014 & 0.9999 & -0.0010 \end{bmatrix}^T, \tag{5.7}$$

which is sufficiently close to the real value.



(a) Orient. error (xe,ye,ze) (b) Orient. error zoom (xe,ye)

Figure 5.11: Simulation 2B: Orientation error (represented in quaternion) with respect to the correct surface normal vector.

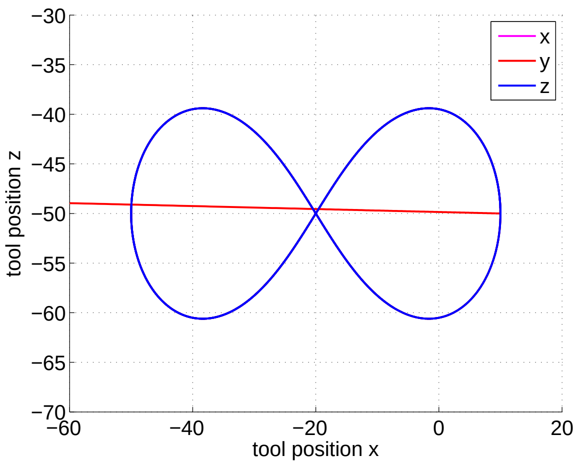
5.3.3 Simulation 3: Trajectory tracking over a *touchscreen*

This simulation goal is to illustrate task A2 (Section 4.4.2), which consists on tracking a trajectory over a *touchscreen* surface. The reference trajectory was defined as an *infinity symbol* - also known as *Bernoulli's Lemniscata* - over the surface $y = 400$ to be tracked after $t = 40$, and it is described as follows:

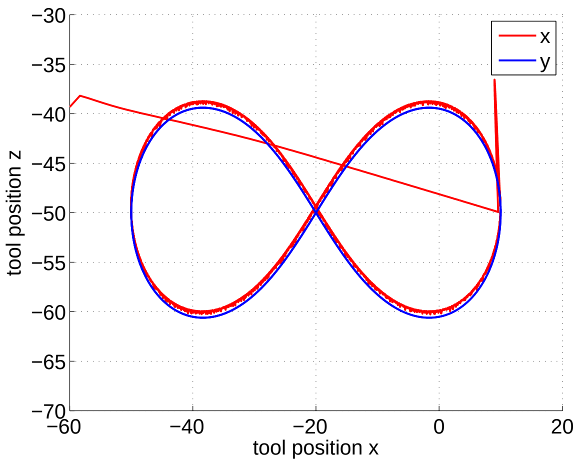
$$p_d = \begin{bmatrix} -20 + \frac{30cos(0.3t)}{sin^2(0.3t+1)} & 400 & -50 + \frac{30sin(0.3t)cos(0.3t)}{sin^2(0.3t+1)} \end{bmatrix}^T, \tag{5.8}$$

$$\dot{p}_d = \begin{bmatrix} \frac{9sin(0.3t)(sin^2(0.3t)+2cos^2(0.6t)+1)}{[sin^2(0.3t)+1]^2} & 0 & \frac{13.5cos(0.6t)-4.5}{[sin^2(0.3t)+1]^2} \end{bmatrix}^T, \tag{5.9}$$

Figure 5.12a shows the $y = 400$ plane projection in x and z-axes. The blue plot denotes the desired trajectory, and the red plot shows the actual path tracked by the origin of $\bar{E}_e$. As expected, the trajectory was followed with zero error. The orientation error around $\vec{x}_e$ and $\vec{y}_e$ axes also remained zero during the entire simulation, and the uncontrolled orientation error around $\vec{z}_e$ axis remained constant during the path tracking ($t \geq 40$).



(a) x-z plot (correct surface)



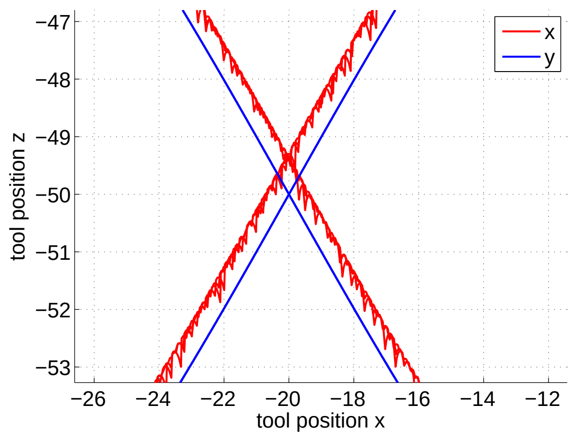
(b) x-z plot (wrong surface)

Figure 5.12: Simulation 3 Plots - *Touchscreen* task.

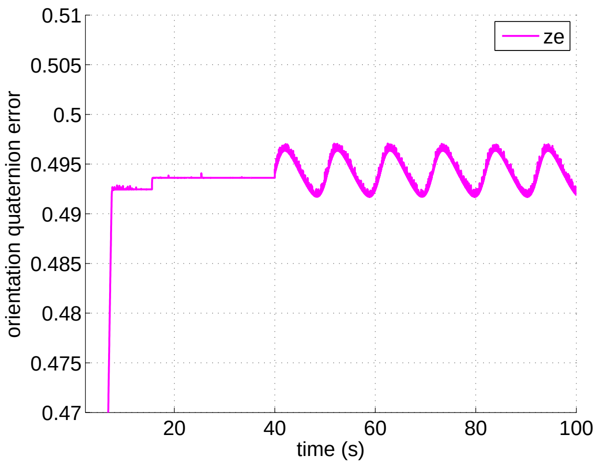
Now, let us consider that the surface of interest is defined as $0.02x+0.8y-0.3z = 398$, which is slightly different from the one that is assumed to be known, and no surface re-estimation is applied. Figure 5.12b shows the desired and the real tracked path. It is worth notice that a small offset of approximately 0.5 mm arises (as better viewed in Figure 5.13a). Also notice in Figure 5.13b that the uncontrolled orientation error around $\vec{z}_e$ axis can freely varies, especially in $t \geq 40$, which is the period that the desired path tracking begins.

5.4 Simulations of Filtered Inverse method

This Section presents several simulations that illustrate the application of the Filtered Inverse FI method (see 2.8) for the online dynamic computation of the Jacobian pseudo-inverse. To simplify the method application and illustration of the obtained results, the operation phases that involve the interaction tasks - i.e., force control and normal vector re-estimation - will not be considered in this Section.



(a) x-z plot zoom (wrong surface)



(b) Orient. error (ze)

Figure 5.13: Simulation 3 Plots - *Touchscreen* task.

5.4.1 Simulation 4: Preliminary implementation of FIA in 5-DoF structure

This simulation aims to test FI to control our original TETIS structure (Figure 3.32).

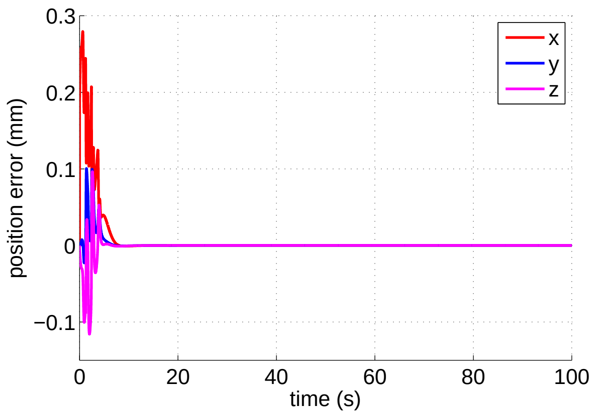
5.4.1.1 Simulation 4A - Standstill at initial pose

Initially, let us define the filter gain as $\gamma = 1$, the simulation step size as $T = 10^{-2}$, the control gains as the same as defined in Section 5.3, and $\Theta(0) = J^{\dagger}[q(0)]$. The initial control goal is to regulate the current robot pose to its initial pose $q_0 = q(0) = (0, 0, 0, 0, 0)$.

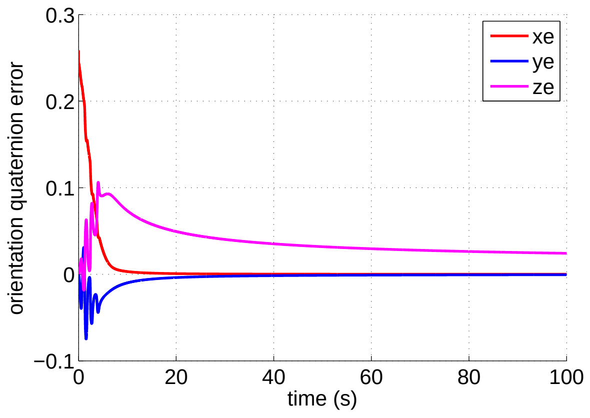
After running the simulation, Matlab/Simulink generated an error that indicated not-finite derivative in the low pass filter/integrator. This error only stopped occurring after reducing either the simulation step size or the update gain γ, from which we conclude that this gain is intrinsically related to the period of the discrete filter algorithm Vargas (2013). The low pass filter/integrator has used the position and orientation errors, e_p and e_o, which were kept in zero with $\gamma = 10^-3$.

5.4.1.2 Simulation 4B - Change of orientation

Setting the step time as $T = 10^{-3}$ and $\gamma = 10^{-3}$, our goal now is to maintain the end-effector at the initial pose, but rolling to an orientation rotated 30° around the $\vec{x}_e$ axis. The position and orientation errors are shown in Figures 5.14a and 5.14b, respectively.



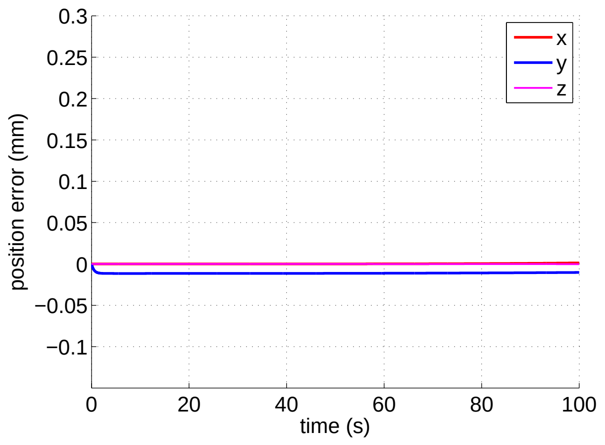
(a) Position error



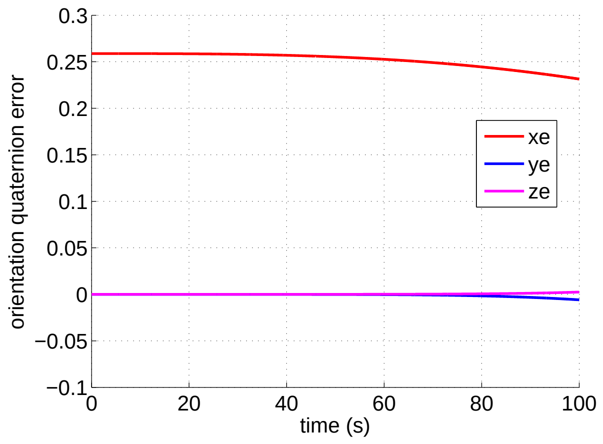
(b) Orient. error

Figure 5.14: Simulation 4B: Plots.

Notice that both errors converge to zero, and the robot slightly translates from the initial position (about 0.3), and then returns back. However, if we change the filter initial condition to $\Theta(0) = 0_{5\times5}$, a small offset in the position error and a large offset in the orientation error emerge, as shown in Figures 5.15a and 5.15b, respectively. Thus, we conclude that the choice of $\Theta(0)$ is of great importance for the Filtered Inverse method.



(a) Position error $\Theta(0) = 0$

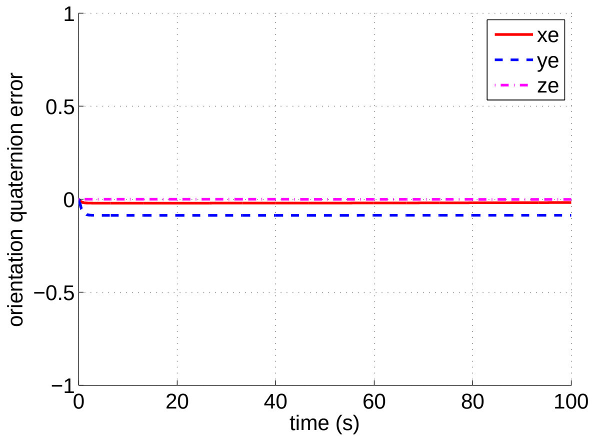


(b) Orient. error $\Theta(0) = 0$

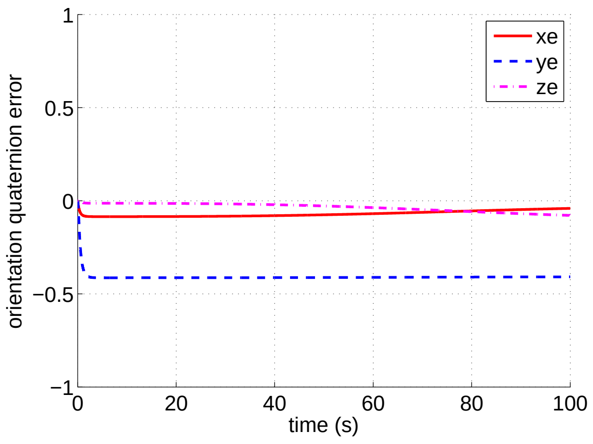
Figure 5.15: Simulation 4B: Plots.

5.4.1.3 Simulation 4C - Change of position

Keeping the same parameters above, this simulation aims is to maintain the end-effector at the initial orientation, but translates about $\delta = (10, 10, 10)$ with respect to its initial position. After running the simulation, the position error always converges to zero, but the orientation error does not. This behavior was observed both to $\Theta(0) = J^{\dagger}(q_0)$ and for other choices of $\Theta(0)$. Figures 5.16a, 5.16b, 5.17a and 5.17b show that the orientation offset is greater the higher is the displacement introduced in the position.

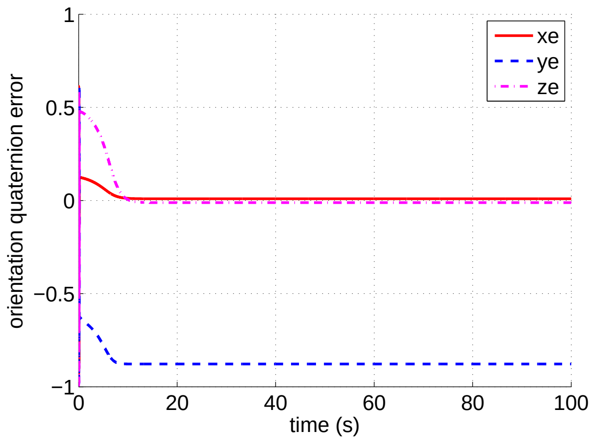


(a) $\Theta(0) = 0_{5\times5}$, $\delta = (10, 10, 10)$

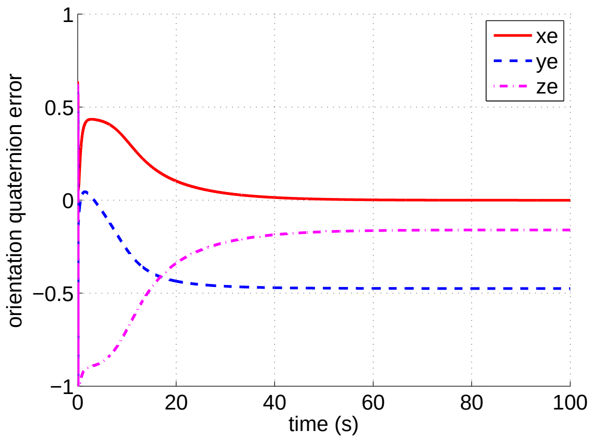


(b) $\Theta(0) = 0_{5\times5}$, $\delta = (50, 50, 50)$

Figure 5.16: Simulation 4C: Orientation error plots.



(a) $\Theta(0) = I_{5\times5}$, $\delta = (10, 10, 10)$



(b) $\Theta(0) = I_{5\times5}$, $\delta = (50, 50, 50)$

Figure 5.17: Simulation 4C: Orientation error plots.

These results led to the investigation of the behaviour of the control signal ν, Θ and $J(q_0)$. We stated that, for all simulation attempts, the control signal ν_4 and ν_5 (responsible for the orientation) did not converge to zero, but the last two rows of Θ did. This showed us that no action was being applied to the signal for controlling the orientation part. Further, an analysis of the Jacobian conditioning was made. The singular values obtained from the SVD decomposition of $J(0)$ (see Section 2.5.5) was given by:

$$\sigma_1 = 311.05,\ \sigma_2 = 272.25,\ \sigma_3 = 67.01,\ \sigma_4 = 0.31,\ \sigma_5 = 0.02 \tag{5.10}$$

in which we can observe very small σ values for the orientation DoFs. One possible solution to overcome this situation would be to start the robot from a pose with all higher singular values. After a survey through all the possible configurations of $q_2, q_3, q_4, q_5 \in (0, 2\pi)$ with offset of $20°$ and not considering q_1 (which does not contribute to J), the minimum and maximum singular values found were:

$$\begin{array}{lllll} \sigma_{1min} = 226.5 & \sigma_{2min} = 56.0 & \sigma_{3min} = 2.6 & \sigma_{4min} = 0.0 & \sigma_{5min} = 0.0 \\ \sigma_{1max} = 669.2 & \sigma_{2max} = 606.1 & \sigma_{3max} = 224.7 & \sigma_{4max} = 1.4 & \sigma_{5max} = 0.2 \end{array} \tag{5.11}$$

Unfortunately, we state that σ for the first three joints are significantly higher than those for the last joints in all configurations, which does not occurs with conventional manipulators, such as anthropomorphic arms, Zebra-Zero, SCARA, Stanford, and others. One possible justification for this high ratio between σ_1 and σ_5 is the presence of a prismatic joint. By replacing TETIS first joint by a revolute type with the same vector $\vec{z}_1$, the minimum and maximum singular values found were:

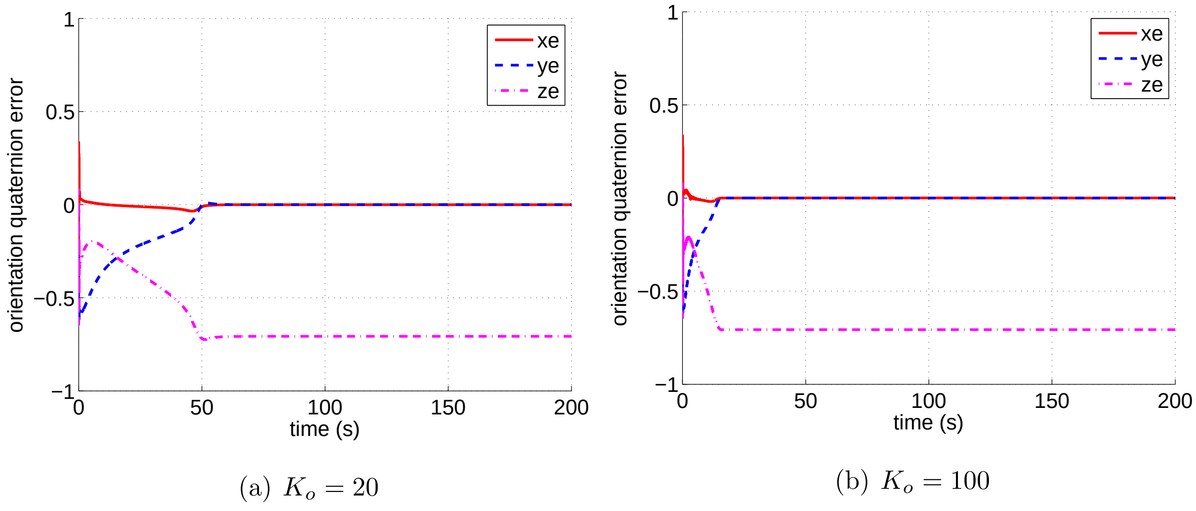
$$\begin{array}{lllll} \sigma_{1min} = 226.5 & \sigma_{2min} = 56.0 & \sigma_{3min} = 2.6 & \sigma_{4min} = 0.2 & \sigma_{5min} = 0.0 \\ \sigma_{1max} = 934.5 & \sigma_{2max} = 659.7 & \sigma_{3max} = 269.5 & \sigma_{4max} = 1.6 & \sigma_{5max} = 1.2 \end{array} \tag{5.12}$$

Notice that $\frac{\sigma_5}{\sigma_1} \approx 3345$ for TETIS original structure, and $\frac{\sigma_5}{\sigma_1} \approx 778$ for the same structure with the revolute joint, which has one less order of magnitude. The dependence of the ratio of the singular values with respect to the type of joints in the robot kinematic structure will be investigated in a future work.

5.4.1.4 Simulation 4D - Changing of integration/solver method and Γ

One possible way to overcome the problem found in above simulation was to introduce a higher gain for the orientation rows of Θ compared to the position rows, i.e., $\gamma_o >> \gamma_p$, and $\Gamma = diag(\gamma_p, \gamma_p, \gamma_p, \gamma_o, \gamma_o)$. However, after many attempts with different control gains and initial condition $\Theta(0)$, the Filtered Inverse method presented no failures only if extremely small step-sizes were used, which did not aggregate practical value.

This result leads us to consider that the Filtered Inverse method are composed of *stiff equations*, i.e., differential equations for which certain integration methods are numerically unstable, unless the step size is extremely small. This motivated the exchange from Bogacki-Shampine (fixed-step) to a variable-step method in Matlab/Simulink environment, in which Adams-Bashforth-Moulton method[1] (ode113) was chosen. For $\gamma_p = 10^{-4}$, $\gamma_o = 10^{-3}$, $q(0) = (0, 0, 0, 0, \pi)$, $\Theta(0) = J_0^\dagger = J^\dagger[q(0)]$, $p_d = (10, 400, -50)$ and $(\vec{z}_e)_d = \vec{y}_b$, we observe that the orientation error converges to zero, as seen in Figures 5.18a and 5.18b for $K_o = 20$ and $K_o = 100$, respectively. The position error also converges to zero in all simulations.



(a) $K_o = 20$ (b) $K_o = 100$

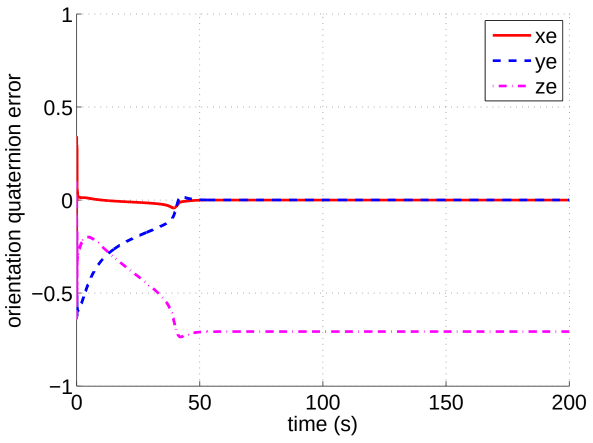
Figure 5.18: Simulation 4D: Orientation errors.

Moreover, considering $K_o = 100$, Figures 5.19a and 5.19b show the orientation error for $\Theta(0) = 0.5J_0^\dagger$ and $\Theta(0) = 0.3J_0^\dagger$, respectively. Notice that the error takes more time to converge to zero the more distant $\Theta(0)$ is from its real initial value.

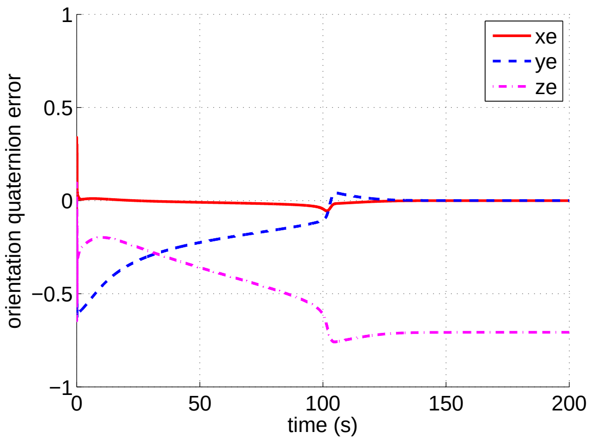
For all above simulations, it was observed that the integration method operated significantly slower at the beginning of the running time until the error convergence to zero, which indicated that a significantly higher effort was provided to adapt Θ 4^{th} and 5^{th} rows until the orientation is controlled, and, consequently, Θ stabilized. As we can notice, if $\Theta(0)$ is very small (Figure 5.20a) or even $\Theta(0) = 0_{5\times5}$ (Figure 5.20b), the error may not converge or either become unstable, regardless of the control gain K_o.

We can conclude that the performance of the Filtered Inverse method depends

[1]Matlab implements several routines for solving ordinary differential equations (ODEs), in which we can highlight Runge-Kutta (ode45 and ode23), Adams (ode113), ode15 and others. Adams method is a multi-step method considered the most efficient at stringent tolerances and in ODEs expensive to be evaluated.

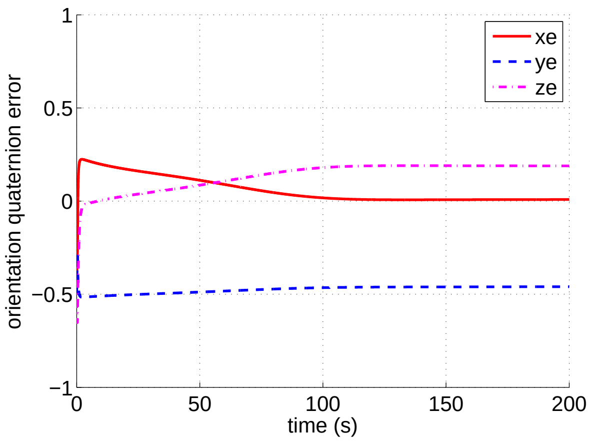


(a) $\Theta(0) = 0.5J_0^\dagger$

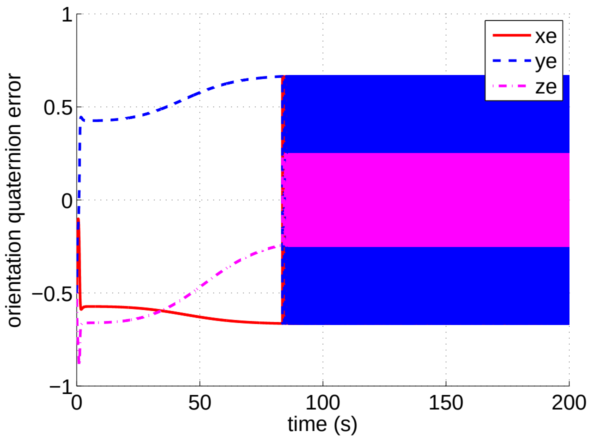


(b) $\Theta(0) = 0.3J_0^\dagger$

Figure 5.19: Simulation 4D: Orientation errors.



(a) $\Theta(0) = 0.01J_0^\dagger$



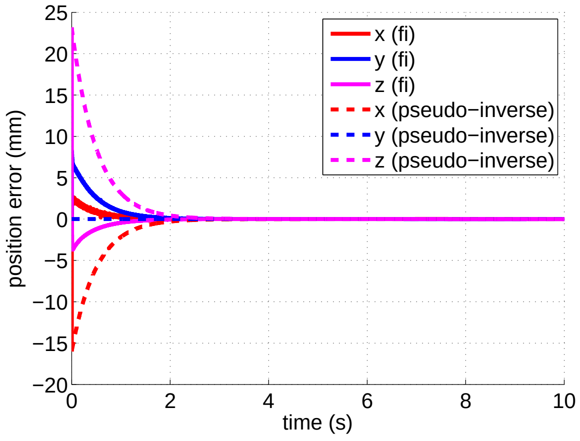
(b) $\Theta(0) = 0_{5\times5}$

Figure 5.20: Simulation 4D: Orientation errors.

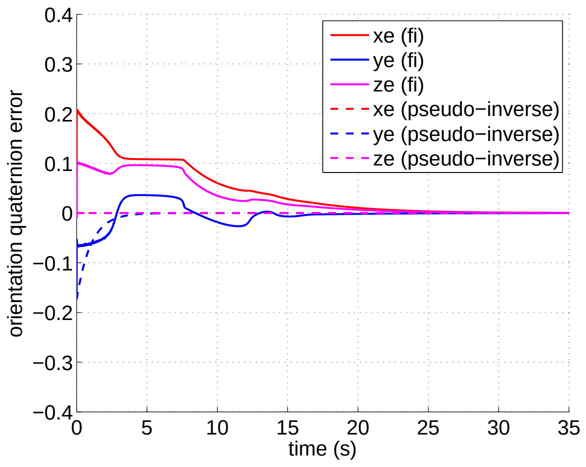
on: (i) the suitable selection of the control gains K_p, K_o; (ii) the inverse initial value $\Theta(0)$, which is recommended not to be near singularities; (iii) the use of a proper numerical method or extremely small step sizes, especially for robots with bad-conditioned Jacobian for all configurations; (iv) the suitable selection of γ, which is recommended to be different for each Θ row, according to the order of magnitude of the Jacobian matrix corresponding singular value (γ_i should be higher for the rows with the smaller σ_i).

5.4.2 Simulation 5: Nest to ready pose

This simulation aims to illustrate TETIS control from nest ($q_{nest} = (0, 0, 0, 0, 0)$) to a ready pose, as motivated in Section 4.5. The ready pose was chosen as the one in which σ_5 is the maximum between all configurations, i.e., $q_{ready} = (0, 0, 0, 340°, 0)$. Setting $K_p = 2$, $K_o = 2$, $\gamma_p = 10^{-4}$, $\gamma_o = 10^{-2}$, the position error, orientation error, Cartesian control signal (ν) and manipulability are shown in Figures 5.21a, 5.21 b, 5.22a and 5.22b, respectively, in which we can compare the application of both Filtered Inverse method (solid lines) and the classic pseudo-inverse Jacobian method (dashed lines).



(a) Position error

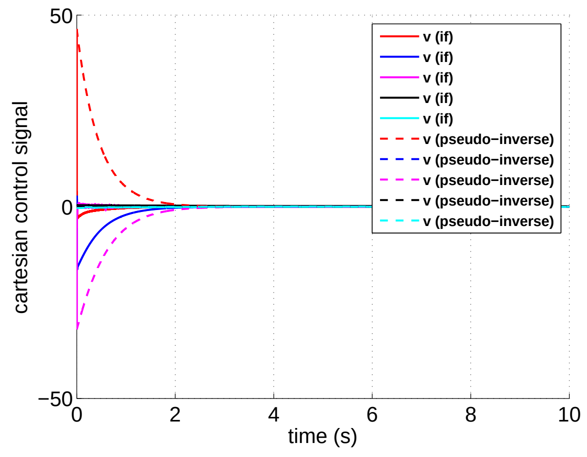


(b) Orientation error

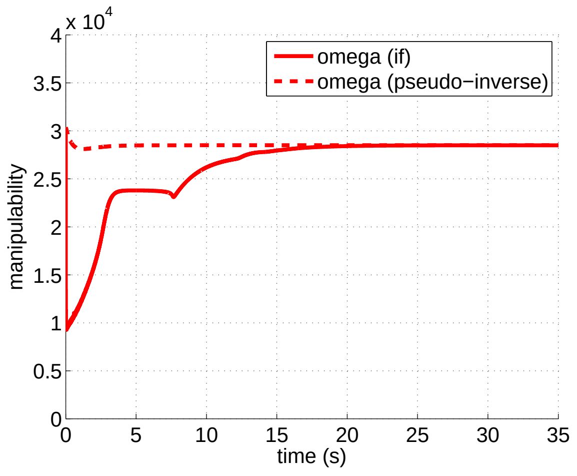
Figure 5.21: Simulation 5: Nest to ready pose.

Notice that the position and orientation errors converge to zero in both methods, being slower for the FI method. However, we can observe that the control effort represented by the Cartesian control signal was smaller in FI method. Notice further that the use of the pseudo-inverse quickly ran away the poses with lower manipulability already in the simulation beginning, whereas FI allowed a temporary manipulability decrease, which was the same moment that a higher effort for the Jacobian inverse adaptation was being provided.

Now, let us consider the case in which the robot depart from $q = (0, 0, 0, 0, 0)$ to $p_{be} \to p_d = (-30.2, -329.5, -521.7)$ and $(\vec{z}_e)_b \to (-0.13, -0.77, -0.62)$, with $K_o = 5$. This trajectory may pass over or very near an internal singular configuration, in which $q_s = (0, 100°, 240°, 160°, 120°)$, which is associated to a manipulability of $\omega' \approx 4.49$ and singular values $\sigma_4' \approx 0.39$, $\sigma_5' \approx 0.00$. Figures 5.23a, 5.23b, 5.24a



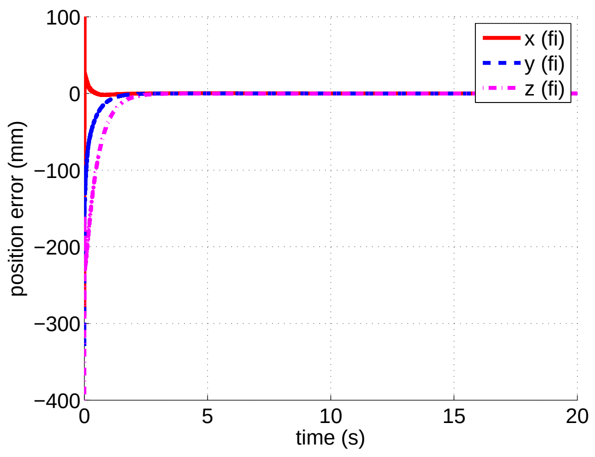
(a) Cartesian control signal



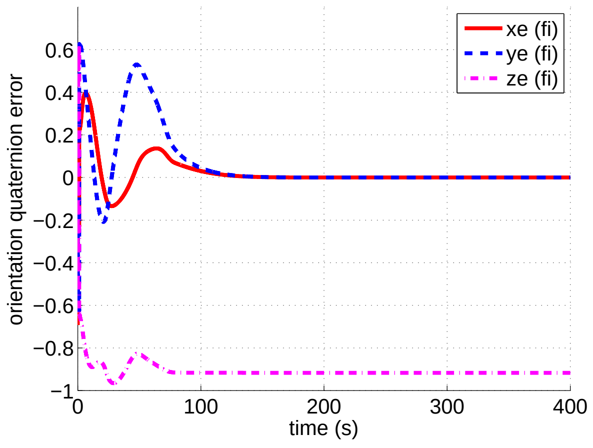
(b) Manipulability

Figure 5.22: Simulation 5: Nest to ready pose.

and 5.24b show the position and orientation error (both converging to zero), the Cartesian control signal ν and the manipulability ω, respectively. Notice that, at the simulation beginning, $\omega \to 0$, which means that the robot passed over a singular configuration.



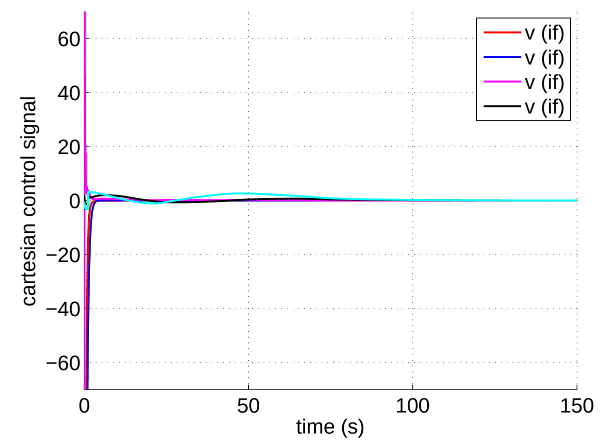
(a) Position error (passing near singularity)



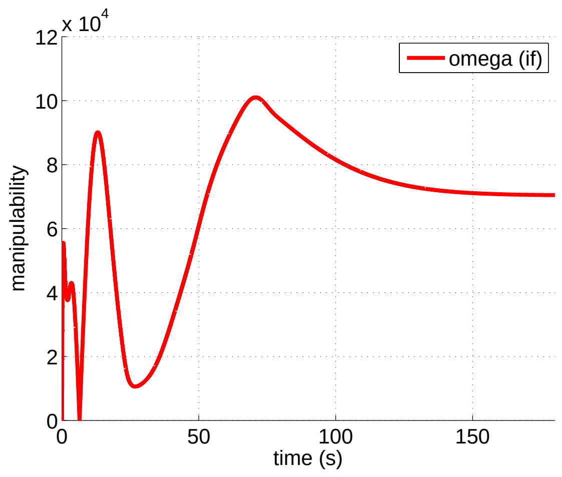
(b) Orient. error (passing near singularity)

Figure 5.23: Simulation 5: Nest to ready pose.

When the Jacobian pseudo-inverse was utilized, the simulation environment could not compute $J^{\dagger}$, since it began too close to a singular matrix and could not numerically escape from this internal singularity.



(a) Cartesian control signal (passing near singularity)

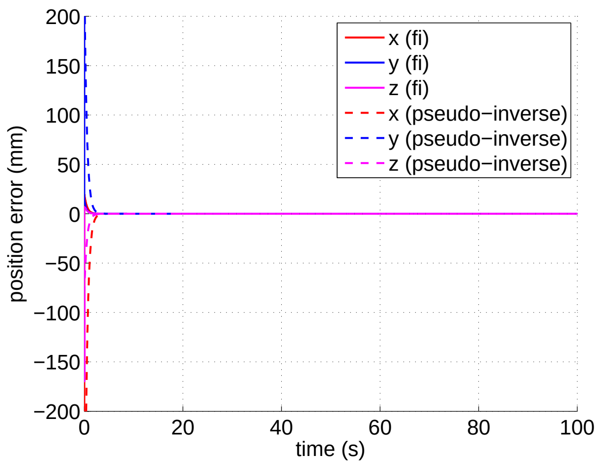


(b) Manipulability (passing near singularity)

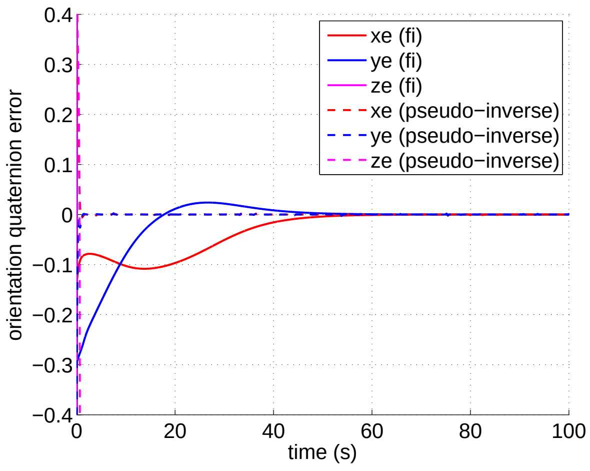
Figure 5.24: Simulation 5: Nest to ready pose.

5.4.3 Simulation 6: Ready pose to approach point

This simulation aims to illustrate TETIS control from the ready pose $q_{ready} = (0, 0, 0, 340°, 0)$ to a desired position and orientation at the approach point H_1, which, for this simulation, we will consider as $p_{be} \to p_d = (-40, 250, -100)$ and $(\vec{z}_e)_b \to (0, 1, 0)$. By setting $K_o = 10$ and changing γ_5 to 0.1 (which provided a better filter performance, as observed), the position and orientation error converged to zero for both Filtered Inverse and pseudo-inverse methods (as seen in Figures 5.25a and 5.25b, respectively).

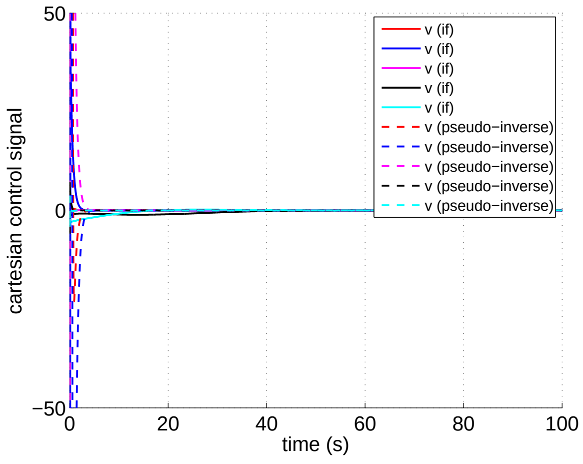


(a) Position error

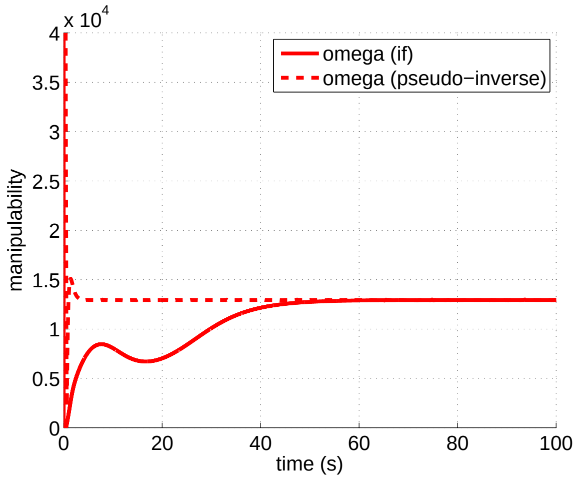


(b) Orient. error

Figure 5.25: Simulation 6: Ready pose to approach point.



(a) Cartesian control signal



(b) Manipulability

Figure 5.26: Simulation 6: Ready pose to approach point.

Notice that the behavior of the position error convergence is almost the same for both techniques, but once again, the orientation error converges faster with the pseudo-inverse, which is explained by the great effort provided by the filter to adapt Θ rows related to σ_4 and σ_5 (as concluded in 5.4.1.4). Observe further in Figure 5.26b that the robot ran away low manipulability almost instantaneously when using $J^{\dagger}$, but this does not occurs for FI, as expected.

5.4.4 Conclusions about the utilization of the Filtered Inverse instead of pseudo-inverse

Considering the results of simulations 5 and 6, we state that the utilization of the Filtered Inverse method allows a wider range of paths towards the goal pose with low computational effort near singularities. This may not consider the planning of trajectories that pass away from internal singular configurations, which are difficult to identify, especially in a structure like TETIS original design, which has some design issues analyzed in Section 3.4.2.

The main difficulty for its utilization may be the tuning of γ, which we concluded that depends on the ratio of magnitude of the Jacobian singular values. Also, it is important to start the filter state $\Theta(0)$ with the values close to $J^{\dagger}(0)$, unless the start pose is near a singularity.

5.5 Simulations of objective functions

This Section presents the simulation results of the application of FI with the augmented Jacobian matrix to satisfy an extra control objective, as explained in Section 2.8.5.

5.5.1 Simulation 7: Mechanical joint limit avoidance

For this simulation, our goal is to keep the manipulator joints within its mechanical limits using the objective function presented in Section 2.8.6.

Thus, by observing our test structure in Figure 3.32, let us assume the following limits:

- $-20 \leq q_1 \leq 20$, to keep DORIS within the straight rail section
- $-240° \leq q_3 \leq 60°$, to avoid collision with DORIS wagon

The selection of the values of both objective function parameters and the proportional gain for orientation control is an empiric task. The f parameters were defined as $\bar{q}_1 = 0, \bar{q}_3 = -\pi/2$, $\delta_1 = 500, \delta_3 = 5\pi/6$, $\alpha_i = 5$, $k_i = 10$, and the orientation proportional gain was set to $K_o = 10$. Hence, we have the following objective function:

$$f(q_1, q_3) = 5\left(\frac{q_1}{20}\right)^{20} + 5\left(\frac{q_3 + \pi/2}{5\pi/6}\right)^{20}. \tag{5.13}$$

Our goal is to achieve the same goal of simulation 6, i.e., depart from $q_{ready} = (0, 0, 0, 340°, 0)$ configuration and regulate the position to $p_{be} \to p_d = (-40, 250, -100)$ and $(\vec{z}_e)_b \to (0, 1, 0)$. Notice in Figure 5.27 that, when no objective function is used, the prismatic joint q_1 achieves large values, which indicates that DORIS travels a considered distance along the rail. This is dangerous, since it can reach a curved section and, consequently, make unfeasible the prismatic movement. However, when the joint limit function (5.13) is utilized, the robot achieves the same control objectives and keep the prismatic joint within $[-20, 20]$, as seen in Figures 5.28a, 5.28b, 5.29a, 5.29b. Observe that the 3^{rd} joint was also kept inside its defined limits $[-4.18, 1.04]$ *rad*. Notice further that the robot passes over a singular region to attempt to conciliate the original control objectives with the additional objective.

During the simulations, we observed a small offset of both position and orientation errors, which were reduced if the control gain related to these variables

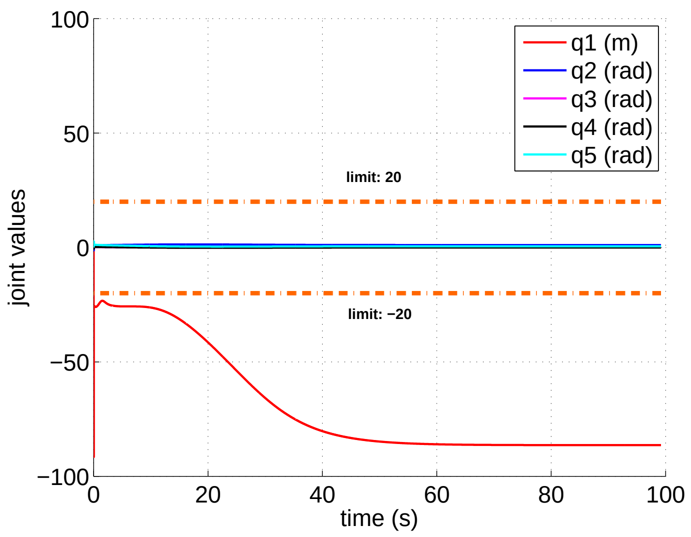
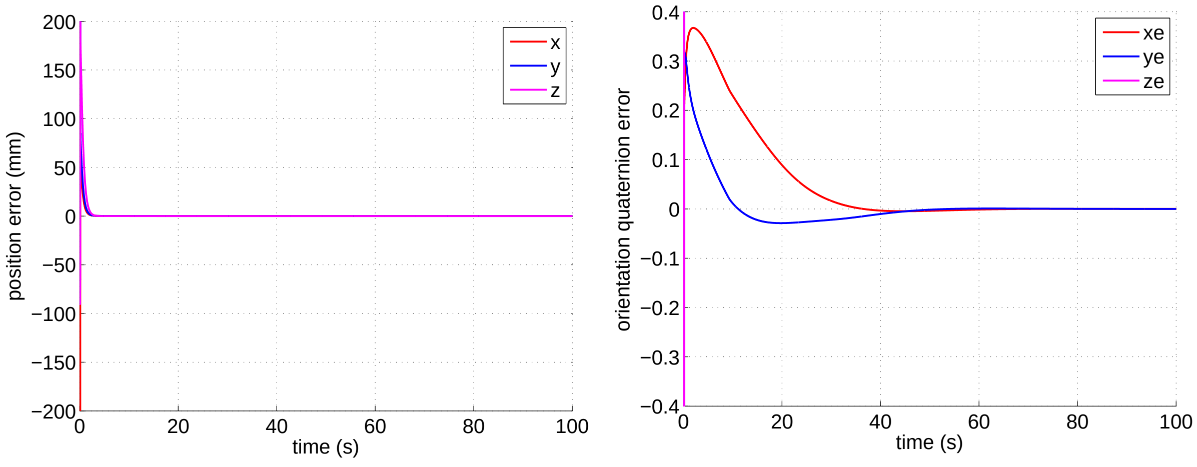


Figure 5.27: Simulation 7: joint values with no objective function.



(a) Position error (using obj. function)

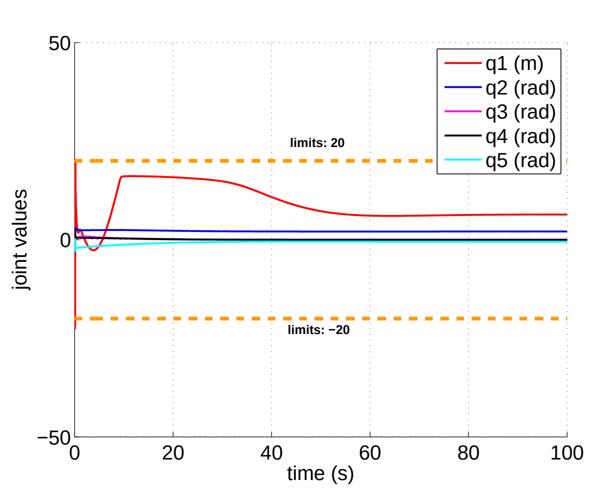
(b) Orient. error (using obj. function)

Figure 5.28: Simulation 7: Mechanical joint limit avoidance.

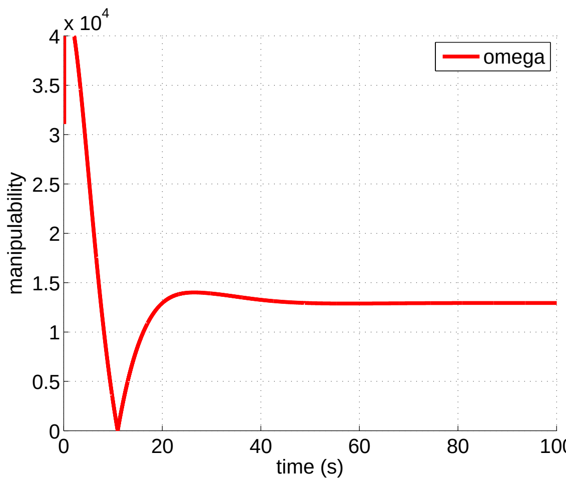
were increased or the α gains of the objective function 5.13 were reduced. This phenomenon was already observed by Vargas (2013).

5.5.2 Simulation 8: Reduce energy consumption

The goal of this simulation is to test the energy cost functions presented in Section 4.5.2, and compare themselves with the case in which no objective function is used. Two performance indexes related to the "consumed power" will be utilized: the robot manipulability ω and the norm-2 of the joint velocities ($\|\dot{q}\|$). Similarly to simulation 7, our goal is to regulate the position to $p_{be} \to p_d = (-40, 250, -100)$



(a) Joint values (using obj. function)



(b) Manipulability (using obj. function)

Figure 5.29: Simulation 7: Mechanical joint limit avoidance.

and $(\vec{z}_e)_b \rightarrow (0, 1, 0)$ departing from $q_{ready} = (0, 0, 0, 340°, 0)$.

5.5.2.1 Simulation 8A - Moving away high-torque configurations

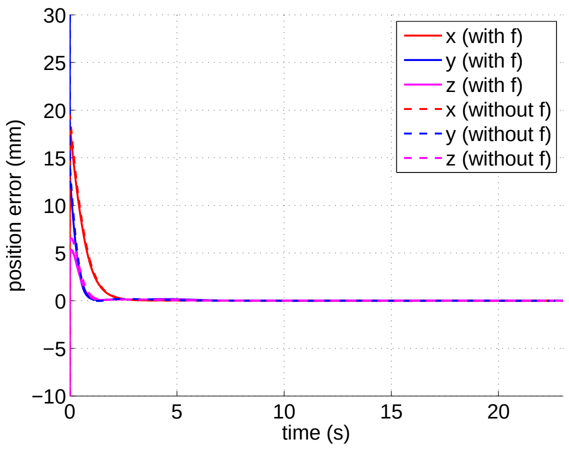
The first proposed energy cost function attempts to keep the robot away from some configurations that clearly demands more power from its joints to support the total torque, as explained in Section 4.5.2.1. By using $\alpha_i = 0.5$ and $k_i = 1$, the position error converged to zero (Figure 5.30a), and the error converged to a small offset (Figure 5.30b), which, after some attempts, was decreased when K_o was increased and/or α_i was decreased. This indicates that a tradeoff must be balanced between the main control goal and the objective function.

Observe that the norm-2 of the joint velocities were slightly reduced during the process (Figure 5.31a), being higher in some periods. The manipulability (Figure 5.31b) was not considerably affected. Different values of parameter α were tested for this cost function (4.26). Figure 5.32 shows that a reduction in energy consumption related to $||\dot{q}||$ may be achieved for higher α gains.

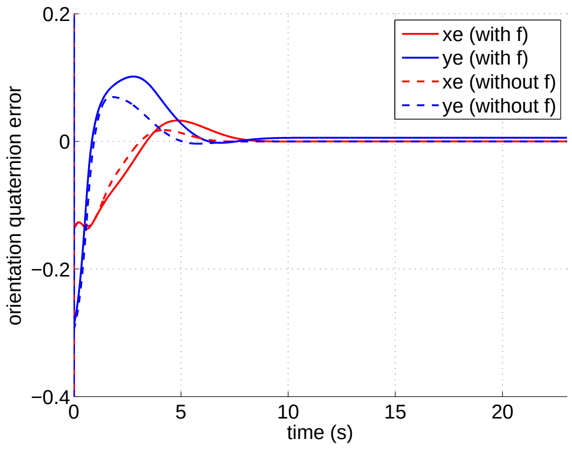
Using this function, the energy consumption could be also reduced if a limitation were imposed for q_1, as already approach by simulation 7.

5.5.2.2 Simulation 8B - Minimum potential energy

The second proposed energy cost function attempts to minimize the net potential energy $f = \mathcal{U}$ stored in the robot structure, as explained in Section 4.5.3. In the

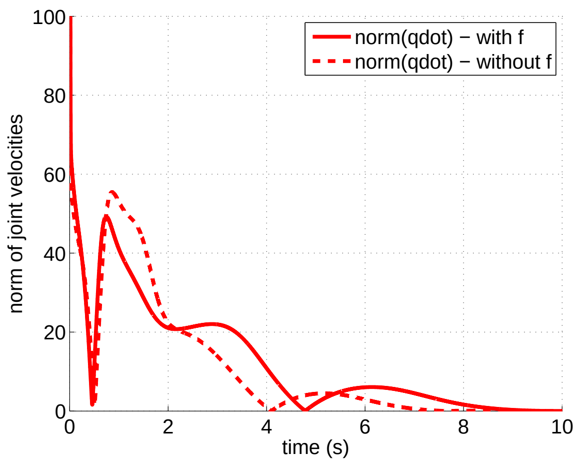


(a) Position error

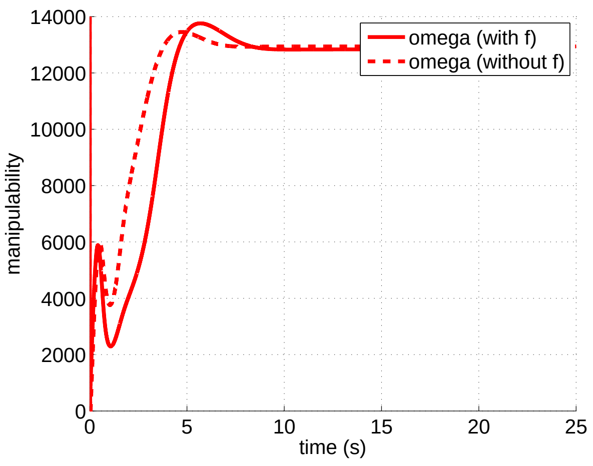


(b) Orient. error

Figure 5.30: Simulation 8A: Moving away from some configurations.



(a) Norm-2 of joint velocities



(b) Manipulability

Figure 5.31: Simulation 8A: Moving away from some configurations.

simulations, we observed that the position and orientation errors (Figures 5.33a and 5.33b) converged to zero only when a smaller cost function $f_{small} = \alpha f$ with $\alpha = 0.001$ was utilized and K_o was increased.

For higher α gains, a significant offset in both position and orientation errors are generated (as observed in Figures 5.34a and 5.34b for $\alpha = 0.01$). Notice further in Figures 5.35a ($\|\dot{q}\|$) and 5.35b (ω) that, for $\alpha = 0.01$, the joint velocities have high peaks at the simulation beginning, but then are taken quickly to zero, whereas the manipulability keeps at a lower value after $\dot{q} \to 0$.

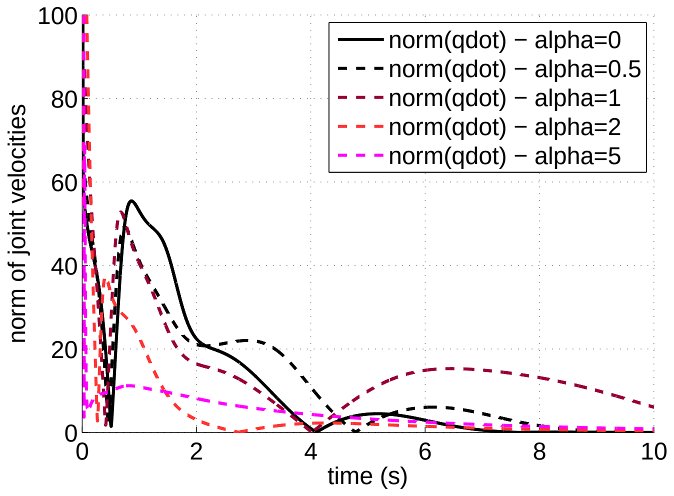
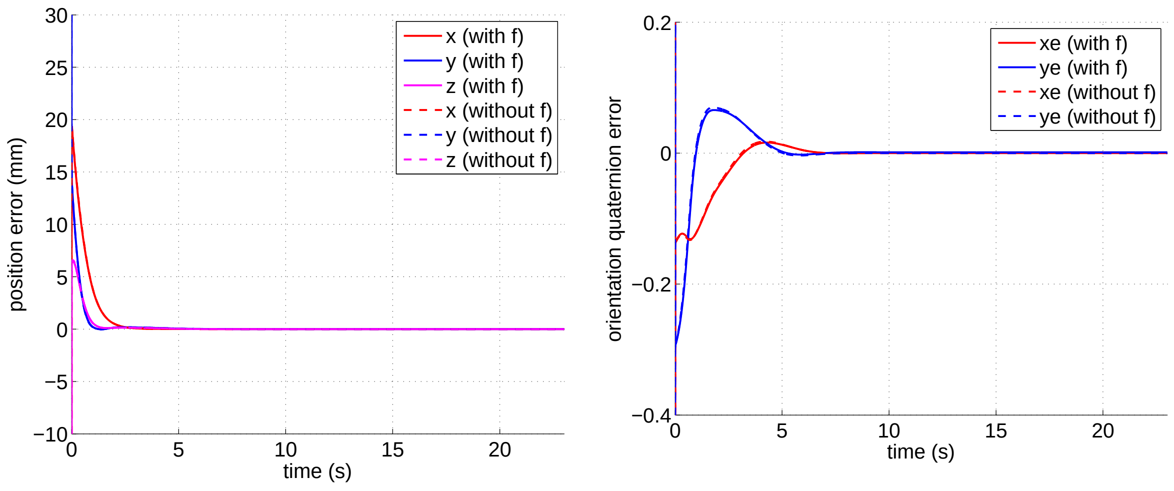


Figure 5.32: Simulation 8A: Norm-2 of joint velocities for different α gains.



(a) Position error $\alpha = 0.001$

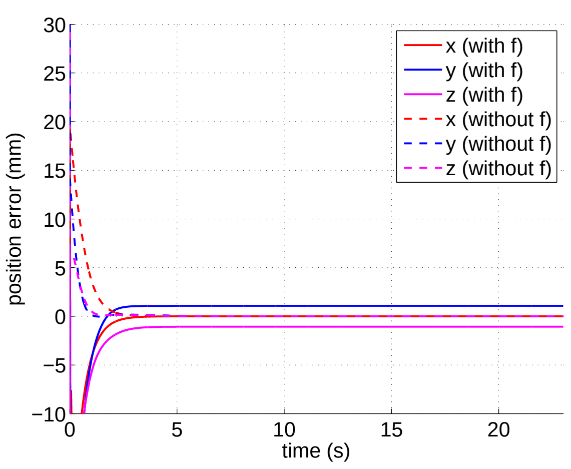
(b) Orient. error $\alpha = 0.001$

Figure 5.33: Simulation 8B: Minimum potential energy.

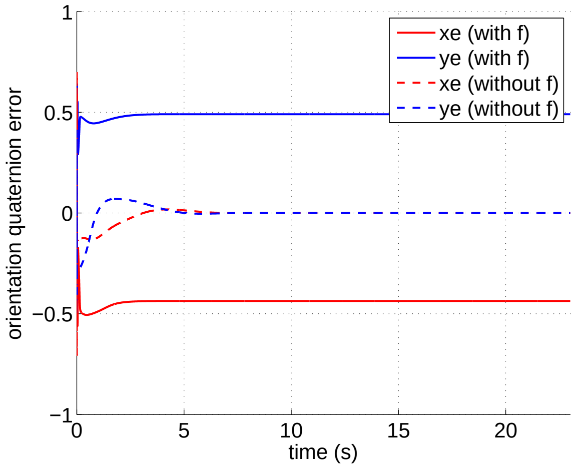
5.5.3 Conclusions about the augmented Jacobian method and the proposed cost functions

Despite the small tracking offset, the objective function for joint limits worked well for TETIS, as can be especially used to prevent large movement of DORIS system along the rail.

The first proposed energy function is similar to the joint-limit function, once its strategy is to keep the robot joints far from some established configurations. The preliminary simulations showed a significant reduction of the overall joint velocities, in spite of the small tracking error. A more comprehensive function of this kind

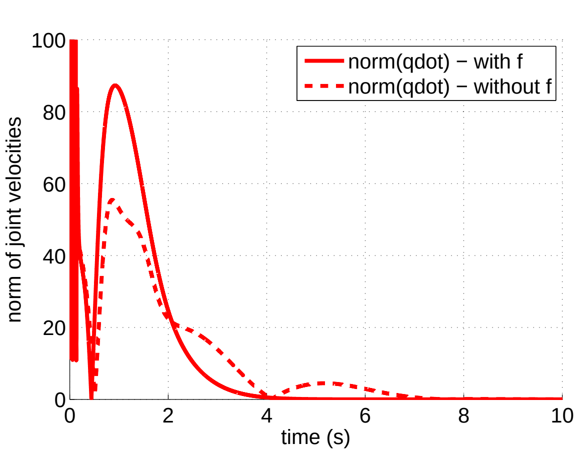


(a) Position error $\alpha = 0.01$

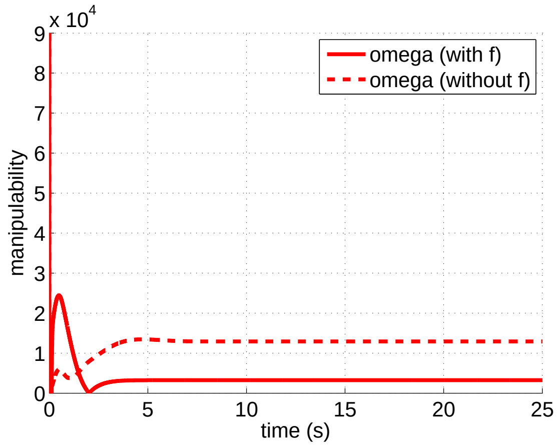


(b) Orient. error $\alpha = 0.01$

Figure 5.34: Simulation 8B: Minimum potential energy.



(a) Norm-2 of joint velocities $\alpha = 0.01$



(b) Manipulability $\alpha = 0.01$

Figure 5.35: Simulation 8B: Minimum potential energy.

should include all the configurations that the robot should avoid to save energy. The strategy of the second function was to minimize the robot potential energy. The simulation results proved this is not a good approach, and the tracking offset was to large.

The tracking offsets are a phenomenon already observed by Vargas (2013) in his simulation about joint limits of a redundant manipulator. This suggests that, regardless of the manipulator DoF, the utilization augmented Jacobian method described by Vargas (2013) may cause a conflict between the control objectives (position and orientation) and the extra goal, which may generate conflicting control signals. We illustrated by simulations 7 and 8 that some objective functions (such as those related to joint limits) are less problematic than others. Also, the priority of the conflicting goals are determined by the control gains (in case of position and orientation tracking) or by the objective function f parameters.

This conflict is not present in the classic approach presented in Section 2.6.5, since the redundant DoF related to the additional control objective are projected in the Jacobian null-space, i.e., it does not affect the control signal that acts to achieve the main control objective. On the other hand, in Vargas (2013) approach, the Jacobian of the additional objective J_F is simply pilled together with the robot Jacobian J. Therefore, the control signals for both main and additional objectives would be conflicting even if the robot has redundant DoF.

Chapter 6

Conclusions and Future Works

6.1 Final conclusions

- This work presented in details the design, modeling and preliminary simulations of manipulators for specific tasks, in which TETIS, the manipulator of DORIS system, was used as a case study. Actually, the studies approached in this work can be extended to the design of any manipulator, but the synthesis of the mechanical structure would hardly depends on the project constraints. In case of TETIS, we stated that the elbow structure formed by the 3^{rd} and 4^{th} joints provides a good range extension and compact retraction pose capacity. Yet, in the example of a manipulator for operation in very constrained spaces, a strongly redundant structure with numerous DoFs but made by short links would be required, so that the robot could reach several positions and orientations within a small dexterous workspace.
- Together with TETIS mechanical design, which required specifications demanded several studies concerning its functionalities and project constraints, the integration of the manipulator with DORIS electric system was also developed. As detailed in Chapter 3, DORIS electric system (Galassi et al. 2014, Freitas et al. 2015) was designed to provide computational support for the desired robot tasks, internal and remote communications, and for supplying and control the robot equipment. The manipulator is controlled by a CAN bus and powered by four power drivers compatible with this network.
- Harmonic Drive Servo-Actuators are the greatest motivation for the construction of an *ad hoc* manipulator, instead of purchasing a commercial solution.

In addition, depending on the required task, no commercially available manipulator may satisfy the specifications. In some cases, they may even fulfill them, but also have additional features (extra DoFs, high power consumption, several communication interfaces, etc.) that are not needed and may turn the model little economically attractive.

- In this work, as studied and discussed in Chapter 3, an standard profile of the relationships between link lengths, robot range, weight, payload and joint velocities was found in several commercially available industrial and high-load manipulators. It was observed that the same relationships are different for lightweight arm, since the material composition of their links - which shows high mechanical resistance - permits the utilization of high-power actuators not only for the joints next to the base, but also for the joints responsible for the tool orientation. Generally, high performance materials are more expensive, but their utilization in lightweight arms may be economically feasible, since the needed material amount in such robot types are smaller than those in industrial models.
- Interaction tasks that require only regulation over unknown flat environments can be combined with a simple technique that places the tool tip at three non-collinear points to estimate the orientation of the contact surface, which uses only the measurements of the robot forward kinematics (interpreted from the actuators' encoders). Hence, this method is simpler and more accurate than others that utilize the force sensor measurements, which are generally noisier.
- The use of FI method (firstly presented by Vargas (2013)) for manipulator control dispenses a sophisticated planning of the trajectories that keep themselves far from the singular configurations. As demonstrated in this work, this occurs because the Jacobian pseudo-inverse matrix is dynamically computed in a kind of first order filter. Therefore, the estimation of $\Theta \approx J^{\dagger}$ is more computationally efficient when the robot is passing near or even over a singularity, since this estimation also depends on previous states. On the other hand, if $J^{\dagger}$ is computed only in function of the joint space q, the robot runs away singular regions that lead the Jacobian to be ill-conditioned. If the initial joint configuration is very near a singularity, the control system may even collapse, as shown in simulation 5.
- FI performance hardly depends on its parameters and the utilized step size,

since it is composed of stiff differential equations. It was stated that the γ gain is intrinsically tied to the system period. Vargas (2013) proposed the utilization of a single gain $\Gamma = \gamma I$ to tune the algorithm. However, we concluded that the adaptation effort of Θ i^{th} row depends on the singular value σ_i associated to it. In TETIS case, the varying range of $\sigma_{1,2,3}$ is almost 10^3 greater than $\sigma_{4,5}$ in order of magnitude. In preliminary simulations with $\Gamma = \gamma I$, it was shown that the orientation error never converged to zero, since the 4^{th} and 5^{th} of Θ were being cancelled. When different gains for each row were set $\Gamma = diag(\gamma_1, \gamma_2, \cdots, \gamma_m)$, both position and orientation errors converged. Empirically, we concluded that the relationship between $\frac{\sigma_i}{\sigma_j}$ and $\frac{\gamma_i}{\gamma_j}$ are inversely proportional.

- One of FI properties is to extend the system Jacobian to include an additional objective function to be optimized together with the control goals. However, this method turns the control goals and the additional function conflicting objectives, whose priority are defined by weighting the control gains or by the function parameters. In the classic method presented in Siciliano et al. (2009) for redundant manipulators, the extra DoFs are projected over the Jacobian null-space, so that the main control objectives are not affected by additional control goals.

- Vargas (2013) proposed two cost functions, which one of then (keeping joints within defined limits) were tested in this work. In TETIS case, keeping the motion of the first prismatic joint within a small length ensures that DORIS never reach a curved rail section.

- Energy consumption in autonomous mobile robots is a complicated issue. A proposed solution was to investigate a cost function $f(q)$ to be optimized with the augmented Jacobian approach. Unfortunately, the estimation of the system power depends on several variables (such as joint velocities, torques, control signal), whereas the cost function depends only of q. Even though, two cost functions strategies were suggested: (a) keep joint away certain configurations that demand more power to support high torques; (b) minimize potential energy. The first strategy showed to be the best approach among them, since the overall joint velocities were reduced, and the cost function shows to be less conflicting with the main control objectives.

- In this work, we stated that the proposed structure for DORIS manipulator presented some characteristics that were not welcomed in the preliminary simulations: non-redundant DoFs, ill-conditioned Jacobian for all poses, and dependence of a prismatic movement (DORIS wagon along rail), whose reliability and repeatability is not mechanically guaranteed. Based on this analysis, some modifications of the original structure were proposed.

- This work presents a self-contained set of essential tools for the study of manipulators, which are extremely important to synthesise an good structure to comply with a given task or functionality, preliminarily analyze its main features (workspace, manipulability, redundant DoF) and problems (singular configurations), simulate its operation in a safe computing environment, and propose modifications of the original design based on the results, as seen in Chapter 3.

6.2 Future works

In order to improve the developed research, the following subjects are proposed to be approached as future works:

- Inclusion of a more complete force sensor model in the simulations, which should include: mass-spring-damping model of both sensor and environment (as approached by Leite (2005)), kinetic friction, spring deformation in three-axes, torque measurement, different behaviours within and without the cone-of-friction.
- More complete analysis of the robot workspace.
- Test of the suggested modifications in the manipulator original structure.
- Matlab/Simulink are suitable and easy-to-use environments for academic purposes. However, it is highly recommended the utilization of proper 3D simulation softwares for better modeling, testing and visualization of the robot geometric structure, motion planning, environment and control algorithms. OpenRAVE (Diankov 2010) and MoveIt! (Chitta et al. 2012) are two of the most used state-of-art environments, being the latter highly integrated with ROS (Robot Operating System), which is an established framework for robot software development (and also used in DORIS project).

- Simulation of infinitesimal displacement method for re-estimation of surfaces with unknown geometry, as presented in Section 2.7.9.1.
- Simulation of more realistic situations - such as manipulator parametric uncertainties, joint encoder resolution, and transmitted vibration effect on the force loop bandwidth - aiming to analyze not only the performance of the utilized control techniques, but also the manipulator performance and robustness during the control operation. In addition, it can be included the uncertainty of the robot motion along the rail which was already estimated in $20mm$, according to field tests.
- Theoretical analysis about the effect of choosing a weighted gain $\Gamma = diag(\gamma_1, \gamma_2, \cdots, \gamma_m)$ for FI method. In our simulation results, we concluded that the gain γ_i should be relatively higher for smaller σ_i.
- Definition of a more generic objective function for reduction of the energy consumption. This should also consider external forces/torques and different directions of the gravity vector, since DORIS may be posed at a leaned or vertical rail section.
- Modifications in the augmented Jacobian method presented by Vargas (2013) aiming to eliminate the conflict between the objective of tracking the robot pose and the additional system constraint. A possible solution would be to project J_f in the null-space of the robot Jacobian J. The classic method (Siciliano et al. 2009) uses the projection operator $(I - J^{\dagger}J)$, which depends on the Jacobian pseudo-inverse, which is not accessible in the Filtered Inverse FI method. However, other projection operators can be investigated.
- Definition of path planning techniques based on obstacle avoidance, such as *artificial potential fields* (Siciliano et al. 2009) and cost function treated by the *augmented Jacobian* (Section 2.8.6.2).
- A more thorough design of the *retraction* phase, including: (i) definition of the *nest* pose (see 4.4.3.2); (ii) definition of the procedures for calibration of the motors' initial positions; (iii) mechanical design of hooks that will support the manipulator at the nest pose.
- Define techniques for task M1 (camera drive) or calibration using robot visual-servoing.

Bibliography

Ackerman, E. (2015), 'Robots with smooth moves are up to 40% more efficient', IEEE Spectrum. http://spectrum.ieee.org/automaton/robotics/industrial-robots/robots-with-smooth-moves-are-more-efficient Accessed on January 16th, 2016.

Anderson, R. J. & Spong, M. W. (1988), 'Hybrid impedance control of robotic manipulators', *Robotics and Automation, IEEE Journal of* **4**(5), 549–556.

Anisi, D., Gunnar, J., Lillehagen, T. & Skourup, C. (2010), Robot automation in oil and gas facilities: Indoor and onsite demonstrations, *in* 'Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)', pp. 4729–4734.

Bar-Itzhack, I. Y. (2000), 'New method for extracting the quaternion from a rotation matrix', *Journal of guidance, control, and dynamics* **23**(6), 1085–1087.

Bechlioulis, C. P., Doulgeri, Z. & Rovithakis, G. A. (2010), 'Neuro-adaptive force/position control with prescribed performance and guaranteed contact maintenance', *Neural Networks, IEEE Transactions on* **21**(12), 1857–1868.

Bengel, M. & Pfeiffer, K. (2007), Mimroex mobile maintenance and inspection robot for process plants. *Fraunhofer Institute for Manufacturing Engineering and Automation IPA*, pp. 1–2.

Bengel, M., Pfeiffer, K., Graf, B., Bubeck, A. & Verl, A. (2009), Mobile robots for offshore inspection and manipulation, *in* 'Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS).', pp. 3317–3322.

Bibalan, P. T. & Featherstone, R. (2009), A study of soft contact models in simulink, *in* 'Proceedings of the Australasian Conference on Robotics and Automation (ACRA)', Citeseer, pp. 2–4.

Bonfadini, A. (2001), Controle coordenado hıbrido de força e posiçao de um manipulador móvel, PhD thesis, Master's thesis, PEE/COPPE/UFRJ.

Borgerink, D., Stegenga, J., Brouwer, D., Wortche, H. & Stramigioli, S. (2014), Rail-guided robotic end-effector position error due to rail compliance and ship motion, *in* 'Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on', IEEE, pp. 3463–3468.

Carneiro, R. F., Leite, A. C., Peixoto, A. J., Goulart, C., Costa, R. R., Lizarralde, F. & Hsu, L. (2006), 'Underwater robot for tunnel inspection: design and control', *Proc 12th Latin-American Congr on Automatic Control, Salvador, Brazil* .

Carvalho, G., Freitas, G., Costa, R., Carvalho, G., Oliveira, J., Netto, S., Silva, E., Xaud, M., Hsu, L., Motta-Ribeiro, G., Neves, A., Lizarralde, F., Marcovistz, I., Peixoto, A., Nunes, E., From, P., Galassi, M. & Røyrøy, A. (2013), 'Doris - monitoring robot for offshore facilities', *Offshore Technology Conference (OTC) Brasil* .

Carvalho, G. P. S. d. (2016), Localization of an autonomous rail-guided robot based on particle filter, Master's thesis, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil.

Chen, H., Stavinoha, S., Walker, M., Zhang, B. & Fuhlbrigge, T. (2014), 'Opportunities and challenges of robotics and automation in offshore oil & gas industry', *Intelligent Control and Automation* .

Chiaverini, S., Siciliano, B. & Egeland, O. (1994), 'Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator', *Control Systems Technology, IEEE Transactions on* **2**(2), 123–134.

Chitta, S., Sucan, I. & Cousins, S. (2012), 'Moveit![ros topics]', *Robotics & Automation Magazine, IEEE* **19**(1), 18–19.

Chou, J. C. (1992), 'Quaternion kinematic and dynamic differential equations', *Robotics and Automation, IEEE Transactions on* **8**(1), 53–64.

Christensen, L., Fischer, N., Kroffke, S., Lemburg, J. & Ahlers, R. (2011), 'Cost-effective autonomous robots for ballast water tank inspection', *Journal of ship production and design* **27**(3), 127–136.

Christensen, L., Kirchner, F., Fischer, N., Ahlers, R., Psarros, G. & Etzold, L. (2011), Tank inspection by cost effective rail based robots, *in* 'Proceedings of International Conference on Computer Applications in Shipbuilding (ICCAS)'.

Corke, P. I. (2011), *Robotics, Vision & Control: Fundamental Algorithms in Matlab*, Springer.

Corrigan, S. (2008), 'Introduction to the controller area network (CAN)', Texas Instrument Application Report SLOA101A. http://www.ti.com/lit/an/sloa101a /sloa101a.pdf Accessed on October 20th, 2014.

Coutinho, F. d. G. (2015), Controle de manipulador redundante com restrições cinemáticas aplicado a cirurgias robóticas assistidas, Master's thesis, Tese de Mestrado, COPPE, UFRJ, Rio de Janeiro.

Craig, J. J. (2005), *Introduction to robotics: mechanics and control*, Vol. 3, Pearson Prentice Hall Upper Saddle River.

Craig, J. J. & Raibert, M. H. (1979), A systematic method of hybrid position/force control of a manipulator, *in* 'Computer Software and Applications Conference, 1979. Proceedings. COMPSAC 79. The IEEE Computer Society's Third International', IEEE, pp. 446–451.

de Lima, A., de M Prego, T., Netto, S. L., da Silva, E. A., Gutierrez, R. H., Monteiro, U. A., Troyman, A. C., da C, S., Francisco, J., Vaz, L. et al. (2013), On fault classification in rotating machines using fourier domain features and neural networks, *in* 'Circuits and Systems (LASCAS), 2013 IEEE Fourth Latin American Symposium on', IEEE, pp. 1–4.

De Luca, A. (2003), 'Robots with flexible links: Modeling and control', *outlined, Roma University, Department of informatics and systematic (DIS), Roma* .

de Oliveira, A. S., De Pieri, E. R., Moreno, U. F. & Martins, D. (2014), 'A new approach to singularity-free inverse kinematics using dual-quaternionic error chains in the davies method', *Robotica* pp. 1–15.

Diankov, R. (2010), Automated Construction of Robotic Manipulation Programs, PhD thesis, Carnegie Mellon University, Robotics Institute.

Diebel, J. (2006), 'Representing attitude: Euler angles, unit quaternions, and rotation vectors', *Matrix* **58**(15-16), 1–35.

Donelan, P. (2010), *Kinematic singularities of robot manipulators*, INTECH Open Access Publisher.

dos Santos Scofano, F., Meggiolaro, M. A. & Sujan, V. A. (2005), 'Inverse kinematics of a binary flexible manipulator using genetic algorithms'.

Emerson (2014), 'Measurement types in machinery monitoring'.

Eppinger, S. D. & Seering, W. P. (1987), Understanding bandwidth limitations in robot force control, *in* 'Robotics and Automation. Proceedings. 1987 IEEE International Conference on', Vol. 4, IEEE, pp. 904–909.

Faber Archila, J. & Becker, M. (2013), Study of robots to pipelines, mathematical models and simulation, *in* 'Robotics Symposium and Competition (LARS/LARC), 2013 Latin American', IEEE, pp. 18–23.

Farsi, M., Ratcliff, K. & Barbosa, M. (1999), 'An overview of controller area network', *Computing & Control Engineering Journal* **10**(3), 113–120.

Fässler, M. (2010), 'Force sensing technologies'.

Ferreira, C. V. & Romano, V. F. (2007), A semi-passive arm concept to support auvs activities, *in* 'Proceedings of the 19th International Congress of Mechanical Engineering, in CD, Brasília, Brazil'.

Ferro, F. & Teixeira, P. (2009*a*), 'Os desafios do pré-sal', Câmara dos Deputados, Edições Câmara. http://www2.camara.leg.br/a-camara/altosestudos/ pdf/Livro-pre-sal.pdf Accessed on October 20^{th} 2014.

Ferro, F. & Teixeira, P. (2009*b*), 'Os desafios do pré-sal. http://www2.camara.leg.br/a-camara/ altosestudos/pdf/livro-pre-sal.pdf. Accessed on October 20^{th}, 2014', Câmara dos Deputados, Edições Câmara, 2009.

Freitas, R. S. (2016), Arquitetura híbrida e controle de missão de robôs autônomos, Master's thesis, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil.

Freitas, R. S., Xaud, M. F. S., Marcovistz, I., Neves, A. F., Faria, R. O., Carvalho, G. P. S., Liu Hsu, E. V. L. N., Peixoto, A. J., Lizarralde, F., Freitas, G., Costa,

R. R., Pål From, M. G., Derks, P. W. J. & Røyrøy, A. (2015), 'The embedded electronics and software of doris offshore robot', *2nd IFAC Workshop on Automatic Control in Offshore Oil and Gas Production (Florianopolis, Brazil)* .

From, P. J. (2010), Off-Shore Robotics: Robust and Optimal Solutions for Autonomous Operation, PhD thesis, Norwegian University of Science and Technology, Trondheim.

From, P. J., Gravdahl, J. T. & Pettersen, K. Y. (2014), *Vehicle-Manipulator Systems*, Springer.

Galassi, M., Røyrøy, A., Carvalho, G., Freitas, G., From, P. J., Costa, R. R., Lizarralde, F., Hsu, L., de Carvalho, G. H., de Oliveira, J. F. et al. (2014), 'Doris - a mobile robot for inspection and monitoring of offshore facilities', *Anais do XX Congresso Brasileiro de Automática* .

GmbH, D. (2011), 'Artis - autonomous railguided tank inspection system. http://robotik.dfki-bremen.de/en/research/robot-systems/artis-1.html. Accessed on January 18th, 2016'.

Graf, B. & Pfeiffer, K. (2007), Mobile robots for offshore inspection and manipulation, *in* 'Proc. Int. Petroleum Technology Conf. (IPTC)'.

Hamilton, W. R. (1844), On a new species of imaginary quantities connected with a theory of quaternions, *in* 'Proceedings of the Royal Irish Academy', Vol. 2, pp. 424–434.

Hamilton, W. R. (1853), *Lectures on Quaternions: Containing a Systematic Statement of a New Mathematical Method; of which the Principles Were Communicated in 1843 to the Royal Irish Academy; and which Has Since Formed the Subject of Successive Courses of Lectures, Delivered in 1848 and Sub Sequent Years, in the Halls of Trinity College, Dublin: Withnumerous Illustrative Diagrams, and with Some Geometrical and Physical Applications*, University Press by MH.

HBM (2015), 'Spoilt for choice: piezoelectric or strain gauge based force transducers. http://www.hbm.com/pt/3719/piezoelectric-or-strain-gauge-based-force-transducers/ . Accessed on February 01st, 2016'.

Hogan, N. (1985), 'Impedance control: An approach to manipulation: Part ii-implementation', *Journal of dynamic systems, measurement, and control* **107**(1), 8–16.

Homayounzade, M. & Keshmiri, M. (2014), Adaptive position/force control of robot manipulators with force estimation, *in* 'Robotics and Mechatronics (ICRoM), 2014 Second RSI/ISM International Conference on', IEEE, pp. 736–741.

Kahan, W. (1983), 'Lectures on computational aspects of geometry. department of electrical engineering and computer sciences', *University of California, Berkeley. Unpublished* .

Klumpp, A. R. (1976), 'Singularity-free extraction of a quaternion from a direction-cosine matrix', *Journal of spacecraft and rockets* **13**(12), 754–755.

Kyrkjebø, E., Liljebäck, P. & Transeth, A. (2009), A robotic concept for remote inspection and maintenance on oil platforms, *in* 'Proc. Int. Conf. on Ocean, Offshore and Arctic Engineering (OMAE)'.

Leite, A. C. (2005), Controle híbrido de força e visão de um manipulador robótico sobre superfícies desconhecidas, Master's thesis, Universidade Federal do Rio de Janeiro.

Leite, A. C. (2011), Servovisão Adaptativa e Controle de Força Para Robôs Manipuladores Com Cinemática e Dinâmica Incertas Interagindo Com Ambientes Não-Estruturados, PhD thesis, Universidade Federal do Rio de Janeiro.

Leite, A. C., Lizarralde, F., From, P. J., Costa, R. R., Hsu, L. et al. (2012), Remote calibration and trajectory replanning for robot manipulators operating in unstructured environments, *in* 'Automatic Control in Offshore Oil and Gas Production', Vol. 1, pp. 59–65.

Leite, A. C., Lizarralde, F. & Hsu, L. (2009), 'Hybrid adaptive vision-force control for robot manipulators interacting with unknown surfaces', *The International Journal of Robotics Research* **28**(7), 911–926.

Lin, G. C. & Lu, T.-F. (1997), 'Vision and force/torque sensing for calibration of industrial robots', *Industrial Robot: An International Journal* **24**(6), 440–445.

Lizarralde, F. & Wen, J. T. (1996), ‘Attitude control without angular velocity measurement: A passivity approach’, *Automatic Control, IEEE Transactions on* **41**(3), 468–472.

Luo, M., Huang, Q. & Zhang, T. (2007), A mathematical model of six-axis force/moment sensor and its applied control method for robot finger, *in* ‘Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on’, IEEE, pp. 1960–1965.

Mason, M. T. (1981), ‘Compliance and force control for computer controlled manipulators’, *Systems, Man and Cybernetics, IEEE Transactions on* **11**(6), 418–432.

Mathas, C. (2012), ‘What you need to know about vibration sensors’, Digikey. http://www.digikey.com/en/articles/techzone/2012/oct/what-you-need-to-know-about-vibration-sensors Accessed on January 10^{th}, 2016.

Mayorga, R. V., Milano, N. & Wong, A. K. (1993), ‘A fast procedure for manipulator inverse kinematics computation and singularities prevention’, *Journal of robotic systems* **10**(1), 45–72.

Meggiolaro, M. A., Jaffe, P. C., Iagnemma, K. & Dubowsky, S. (1999), A force-updated kinematics virtual viewing system with application to nuclear power plant maintenance, *in* ‘Proceedings of the 10th World Congress on the Theory of Machine and Mechanisms’, Citeseer.

Meggiolaro, M. A., Scriffignano, G. & Dubowsky, S. (2000), Manipulator calibration using a single endpoint contact constraint, *in* ‘Proceedings of ASME Design Engineering Technical Conference, Baltimore, USA’.

Meng, J., Liu, G., Li, Z. et al. (2007), ‘A geometric theory for analysis and synthesis of sub-6 dof parallel manipulators’, *IEEE Transactions on Robotics* **23**(4), 625–649.

Murray, R. M., Li, Z., Sastry, S. S. & Sastry, S. S. (1994), *A mathematical introduction to robotic manipulation*, CRC press.

Nakamura, Y. & Hanafusa, H. (1986), ‘Inverse kinematic solutions with singularity robustness for robot manipulator control’, *Journal of dynamic systems, measurement, and control* **108**(3), 163–171.

NPL (2010), 'How many different types of force transducer are there (faq - force). http://www.npl.co.uk/reference/faqs/how-many-different-types-of-force-transducer-are-there-(faq-force) . Accessed on February 01st, 2016'.

NREC/CMU (2012), Sensabot: A safe and cost-effective inspection solution. *Jour. of Petroleum Technology*, pp. 32–34.

Oliveira, P., Pascoal, A., Silva, V. & Silvestre, C. (1998), 'Mission control of the marius autonomous underwater vehicle: system design, implementation and sea trials', *International journal of systems science* **29**(10), 1065–1080.

P. Oliveira, A. Pascoal, V. S. C. S. (1996), Mission control of the MARIUS AUV: System design, implementation, and sea trials.

Paden, B. E. (1985), 'Kinematics and control of robot manipulators'.

Paes, K., Dewulf, W., Vander Elst, K., Kellens, K. & Slaets, P. (2014), 'Energy efficient trajectories for an industrial abb robot', *Procedia CIRP* **15**, 105–110.

Pechev, A. N. (2008), Inverse kinematics without matrix inversion, *in* 'Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on', IEEE, pp. 2005–2012.

Peiper, D. L. (1968), The kinematics of manipulators under computer control, Technical report, DTIC Document.

Piezotronics, P. (2016), 'Introduction to piezoelectric accelerometers', PCB Piezotronics. http://www.pcb.com/techsupport/tech_accel Accessed on February 25th, 2016.

Qt-Project (2014), 'Qt online documentation. http://qt-project.org/ Accessed on October 20th, 2014'.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R. & Ng, A. (2009), ROS: an open-source robot operating system, *in* 'Proc. Int. Conf. on Robotics and Automation (ICRA), ser. Open-Source Software Workshop'.

Raibert, M. H. & Craig, J. J. (1981), 'Hybrid position/force control of manipulators', *Journal of Dynamic Systems, Measurement, and Control* **103**(2), 126–133.

Ribeiro, G. C. d. M. (2013), Teleoperação bilateral de múltiplos robôs aplicada ao transporte de carga, Master's thesis, Tese de Mestrado, COPPE, UFRJ, Rio de Janeiro.

Salisbury, J. K. (1980), Active stiffness control of a manipulator in cartesian coordinates, *in* 'Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on', IEEE, pp. 95–100.

Shukla, A. & Karki, H. (2013), A review of robotics in onshore oil-gas industry, *in* 'Mechatronics and Automation (ICMA), 2013 IEEE International Conference on', IEEE, pp. 1153–1160.

Siciliano, B. & Khatib, O. (2008), *Springer handbook of robotics*, Springer Science & Business Media.

Siciliano, B., Sciavicco, L., Villani, L. & Oriolo, G. (2009), *Robotics: modelling, planning and control*, Springer Science & Business Media.

Skourup, C. & Pretlove, J. (2009), 'The robotized field operator', *ABB Review* (1), 68–73.

Slotine, J.-J. E., Li, W. et al. (1991), *Applied nonlinear control*, Vol. 199, Prentice-hall Englewood Cliffs, NJ.

Spong, M. W., Hutchinson, S. & Vidyasagar, M. (2006), *Robot modeling and control*, Vol. 3, Wiley New York.

Sujan, V. A. & Meggiolaro, M. A. (2005), 'Intelligent and efficient strategy for unstructured environment sensing using mobile robot agents', *Journal of Intelligent and Robotic Systems* **43**(2-4), 217–253.

Tecnadyne (2006*a*), *General power system wiring practices applied to Tecnadyne DC brushless motors.*

Tecnadyne (2006*b*), 'General power system wiring practices applied to tecnadyne dc brushless motors', Tecnadyne Application Note AN601. http://www.tecnadyne.com/cms/images/products/pdf /AN601_Back_EMF_-Ground_Loops.pdf Accessed on October 20th, 2014.

Tekscan (2012), 'Tekscan ebook - force sensor study guide'.

Tikhonov, A. N. & Arsenin, V. I. (1977), *Solutions of ill-posed problems*, Vh Winston.

Vargas, L. V. (2013), Inversa filtrada: Uma solucao alternativa para a cinematica inversa de manipuladores roboticos, Master's thesis, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil.

Vergnano, A., Thorstensson, C., Lennartson, B., Falkman, P., Pellicciari, M., Leali, F. & Biller, S. (2012), 'Modeling and optimization of energy consumption in cooperative multi-robot systems', *Automation Science and Engineering, IEEE Transactions on* **9**(2), 423–428.

Verma, V., Gordon, G., Simmons, R. & Thrun, S. (2004), 'Real-time fault diagnosis [robot fault diagnosis]', *Robotics & Automation Magazine, IEEE* **11**(2), 56–66.

Wampler, C. W. et al. (1986), 'Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods', *Systems, Man and Cybernetics, IEEE Transactions on* **16**(1), 93–101.

Wilson, J. (1999*a*), 'Acceleration/vibration - a practical approach to vibration detection and measurement part 1: Physical principles and detection techniques', Sensors Online. http://www.sensorsmag.com/sensors/acceleration-vibration/a-practical-approach-vibration-detection-and-measurement-par-951 Accessed on January 10th, 2016.

Wilson, J. (1999*b*), 'Acceleration/vibration - a practical approach to vibration detection and measurement, part 2: Dynamic and environmental effects on performance', Sensors Online. http://www.sensorsmag.com/sensors/acceleration-vibration/a-practical-approach-vibration-detection-and-measurement-par-6370 Accessed on January 10th, 2016.

Wilson, J. (1999*c*), 'Acceleration/vibration - a practical approach to vibration detection and measurement, part 3: Installation, recalibration, and application', Sensors Online. http://www.sensorsmag.com/sensors/acceleration-vibration/a-practical-approach-vibration-detection-and-measurement-par-6371 Accessed on January 10th, 2016.

World-nuclear (2012), 'Uranium, electricity and climate change', World Nuclear Association website. http://www.world-nuclear.org/info/Energy-and-

Environment/Uranium-Electricity-and-Climate-Change Accessed on October 20th, 2014.

World-nuclear (2014), ‘Uranium, electricity and climate change. http://www.world-nuclear.org/info/energy-and-environment/uranium-electricity-and-climate-change . Accessed on October 20th, 2014’.

Yuan, J. S. (1988), ‘Closed-loop manipulator control using quaternion feedback’, *Robotics and Automation, IEEE Journal of* **4**(4), 434–440.

Appendix A

Selection criteria of electromechanical parts

This appendix presents in details the commercial selection of actuators, controllers and sensors that composẽmanip structure.

A.1 Selection of adequate actuators based on torque calculation

Given three possible choices among FHA Mini Servomotors (FHA-8C-100-D200-EKMI, FHA-11C-100-D200-EKMI and FHA-14C-100-D200-EKMI), it was made an estimation on how much necessary torque each joint should provide to support the required payload ($300g$, as defined in Section 3.2.1). Initially, there were considered the gravity acceleration as $g = 9.810 m/s^2$ and the specifications of each motor type (Table A.1).

A.1.1 Links' mass

The next step was to found the effective mass of each link i, so that one could compute the necessary torque that joint i should provide to drive the mass of the subsequent chain of the links $i, i+1, \cdots, n$. As will be discussed further in Section 3.2.7.2, the links will be composed of hollow carbon fiber tubes. In this project, only $0.995 in$ and $1.120 in$ OD (outer diameter) tubes were used, each one with $0.11 lbs/ft$ and $0.13 lbd/ft$ of linear density, respectively. Hence, it was possible to estimate the mass of each link (as shown in Table A.2).

Table A.1: Harmonic Drive AG Servo Actuators - Specifications

Item/Model	FHA-8C*	FHA-11C*	FHA-14C*
Gear ratio	100:1	100:1	100:1
$1min$ torque ($N \cdot m$)	3.3	7	13
Rated torque ($N \cdot m$)	2	4.2	6.8
Torque constant ($N \cdot m/A$)	2.7	2.6	2.9
Max. output speed (rpm)	60	60	60
Rated speed (rpm)	35	35	30
Max. current (A)	2.4	5.6	12.3
Rated voltage (VDC)	24	24	24
Maximum voltage (VDC)	48	48	48
Mass (kg)	0.4	0.62	1.2
Max. moment load ($N \cdot m$)	15	40	75
Encoder resolution at output ($p/rev.$)	200000	200000	200000
Encoder resolution at output ($qc/rev.$)	800000	800000	800000
Number of pole pairs	5	5	5
Thermal time constant (min)	15	17	14

$*$ -100-D200-EMKI

Table A.2: Links mass - Estimation

Item/Link	Link 5	Link 4	Link 3	Link 2
Tube OD (in)	0.995	0.995	1.120	$*$
Est. tube length** (mm)	205	225	320	$*$
Tube mass*** (g)	25	37	62	$*$
Accessories mass**** (g)	50	120	140	75
Center-of-mass ($\overline{CM}$) dist. to joint***** (mm)	52	113	160	$*$
Total mass (g)	75	157	202	75

$*$ This calculation does not apply to link 2, since it is not a tube (see Section 3.2.7.1), and the rotation axis of joint 2 crosses with the axis of joint 3 (i.e., its center-of-mass distance to the joint is negligible). $**$ Estimation from May 2015. $***$ Calculated based on the carbon fiber density. $****$ Accessories: coupled joints (see Section 3.2.7.2), cables, connectors, etc. $*****$ Estimated.

A.1.2 Calculations for the 5th joint

The subsequent step is to find the necessary torques for each joint and the supported payloads by testing each motor type. Firstly, let us consider only link 5 (which applied forces are represented in Figure A.1). The required torque for joint 5 is given by $\tau_{5,min} = g(m_5\overline{CM}_5 + m_p l_5) = 0.64Nm$. As we can see from Table A.1, this torque can be provided by all the three motor options. Thereby, we can calculate the supported payload of Joint 5 by $m_{p5} = \frac{\tau_5 - m_5 \cdot \overline{CM}_5 \cdot g}{l_5 \cdot g}$. For the situation of the arm fully horizontally extended, we consider the $1min$ torque (Table A.1), which can be safely provided during 1 minute without the risk of overheating. For the retracted arm situation, we consider the *rated* torque as an input. It is worth stating that, presumably, no manipulator task will last longer than $1min$ and the arm will be rarely fully extended, which is why we can consider a higher torque than the *rated* for this situation. Table A.5 shows the estimated payload for joint 5 considering all the motor options. Notice that all the options provide the needed payload in both situations, but FHA-8C seemed the most suitable choice for joint 5, since it is the lightest model.

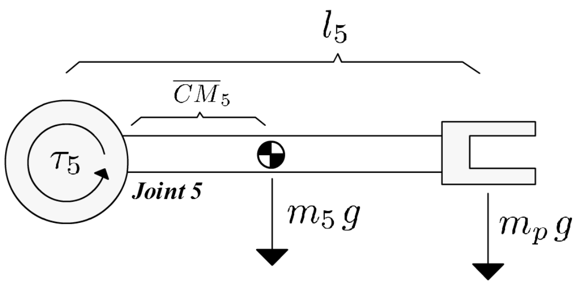


Figure A.1: Link 5, and joint 5 - Forces.

Table A.3: Joint 5 - Payloads

Situation/Model	FHA-8C	FHA-11C	FHA-14C
Payload - extended arm (g)	1622	3462	6445
Payload - retracted arm (g)	976	2069	3362

A.1.3 Calculations for the 4th joint

Once the joint 5 motor is defined, the same calculations had to be performed for joint 4. According to Figure A.2 that represents both links 4 and 5 fully horizontally extended, the minimum torque that joint 4 must provide to support the required payload is given by $\tau_{4,min}^{ext} = g\left[m_4\overline{CM}_4 + \bar{m}_5 l_4 + m_5(l_4 + \overline{CM}_5) + m_p(l_4 + l_5)\right] = 2.53Nm$. However, when the arm is fully retracted, link 5 stays rotate 180° around joint 5, and the necessary torque for joint 4 to support the system when the arm is retracted is given by $\tau_{4,min}^{ret} = g\left[m_4\overline{CM}_4 + \bar{m}_5 l_4 + m_5(l_4 - \overline{CM}_5) + m_p(l_4 - l_5)\right] = 1.24Nm$. Similarly, looking at the $1min$ values (for the extended arm) and *rated* values (for the retracted arm) at Table A.1, we can notice that all the three motor options can provide the required torque at joint 4. We can though estimate the payload that joint 4 could support (Table A.4) if all the three motor options were used, being that:

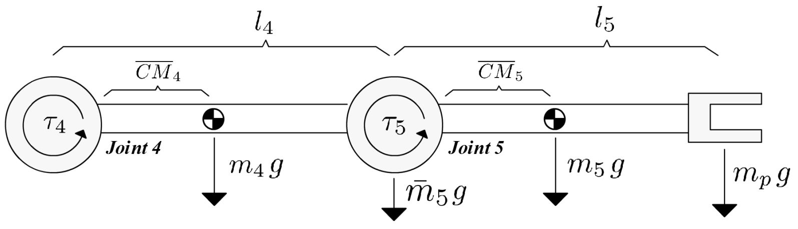


Figure A.2: Links 4 and 5, and joints 4 and 5 - Forces.

$$m_{p4}^{ext} = \frac{\tau_4^{ext} - \left[m_4\overline{CM}_4 + \bar{m}_5 l_4 + m_5(l_4+\overline{CM}_5)\right]g}{(l_4+l_5)g} \text{ (extended)}$$

$$m_{p4}^{ret} = \frac{\tau_4^{ret} - \left[m_4\overline{CM}_4 + \bar{m}_5 l_4 + m_5(l_4-\overline{CM}_5)\right]g}{(l_4-l_5)g} \text{ (retracted).}$$

Once again, FHA-8C is the best choice for joint 4 among the three available models,

Table A.4: Joint 4 - Payloads

Situation/Model	FHA-8C	FHA-11C	FHA-14C
Payload - extended arm (g)	483	1361	2783
Payload - retracted arm (g)	4158	15371	28623

since it is the lightest.

A.1.4 Calculations for the 2^{nd} and 3^{rd} joints

Analogously for joint 3, we should consider the three links both extended and retracted. For the extended arm (Figure A.3), we state that joint 3 must support provide a torque of:

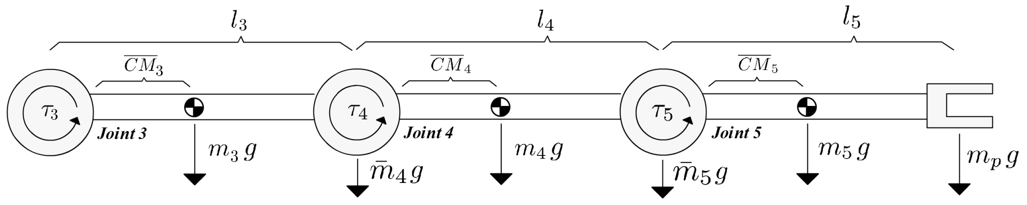


Figure A.3: Links 3, 4 and 5, and joints 3, 4 and 5 - Forces.

$$\tau_{3,min}^{ext} = g[m_3\overline{CM}_3 + \bar{m}_4 l_3 + m_4(l_3 + \overline{CM}_4) + \bar{m}_5(l_3 + l_4) + m_5(l_3 + l_4 + \overline{CM}_5) + m_p(l_3 + l_4 + l_5)] = 7.03Nm.$$

On the other hand, the retraction situation causes joint 3 to require a torque of:

$$\tau_{3,min}^{ret} = g[m_3\overline{CM}_3 + \bar{m}_4 l_3 + m_4(l_3 - \overline{CM}_4) + \bar{m}_5(l_3 - l_4) + m_5(l_3 - l_4 + \overline{CM}_5) + m_p(l_3 - l_4 + l_5)] = 3.26Nm.$$

As we can see from Table A.1, only motors FHA-11C and FHA-14C can provide the needed torque. We can estimate the payload that joint 3 could support (Table A.5) considering using these two models, being that:

$$m_{p3}^{ext} = \frac{\tau_3^{ext} - \left[m_3\overline{CM}_3 + \bar{m}_4 l_3 + m_4(l_3+\overline{CM}_4) + \bar{m}_5(l_3+l_4) + m_5(l_3+l_4+\overline{CM}_5)\right]g}{(l_3+l_4+l_5)g} \text{ (extended)}$$

$$m_{p3}^{ret} = \frac{\tau_4^{ret} - \left[m_3\overline{CM}_3 + \bar{m}_4 l_3 + m_4(l_3-\overline{CM}_4) + \bar{m}_5(l_3-l_4) + m_5(l_3-l_4+\overline{CM}_5)\right]g}{(l_3-l_4+l_5)g} \text{ (retracted).}$$

Notice that the motor FHA-11C supports $3g$ less than the stipulated of $300g$.

Table A.5: Joint 5 - Payloads

Situation/Model	FHA-11C	FHA-14C
Payload - extended arm (g)	297	1112
Payload - retracted arm (g)	621	1504

However, it is worth to highlight that: (i) $300g$ payload already considers a safe gap (as seen in Section 3.2.1); (ii) FHA-11C weighs almost half of what FHA-14C does, being that the manipulator total weight is a critical point of the project; (iii) FHA-14C may require a maximum of 12.3 supply current, which is not supported by any selected driver for the project (see Section 3.2.5); (iv) FHA-14C is much more expensive than FHA-11C. Hence, the most suitable choice for joint 3 is the motor FHA-11C.

The calculations for joint 2 are simple that the above, since its rotation axis crosses joint 3 axis. Therefore, it is effortless to notice that joint 2 should provide the same torque as joint 3, hence, FHA-11C motor is also the best choice for it.

A.2 Selection of controller/power drivers

According to Harmonic Drive, all Maxon Motor EPOS2 sub-models are compatible to its FHA Mini Servo Actuators. We should highlight that DORIS *actuation power bus* provides a maximum of 33.6V, while FHA Mini actuators are supplied by a nominal voltage of 24VDC and maximum of 48VDC (see Table A.1). A priori, the best solution requires a driver able to provide 24VDC power output.

It is worth to remind that these driver models type take an input voltage V_{in} and output a three-phase power signal that commands the EC (Electronically commuted) motor, like our case. In Maxon Motor EPOS2 drivers, $V_{out} = 0.9V_{in}$ (see model datasheets). Thereby, the driver EPOS2 24/5 (that supports a maximum of 24VDC and 5A) could be utilized in the 33.6V DORIS bus. Yet, it would require a DC-DC converter for each joint to avoid burn-out. Unfortunately, the use of these four extra DC-DC would bring several disadvantages to the system, such as: (i) heat generation; (ii) increase of occupied space and weight; (iii) unforeseen instabilities in DC-DC operation.

A.2.1 Solution

The other two EPOS2 models dispense the use of DC-DCs, since they support a maximum of 50VDC/5A (EPOS2 50/5) and 70VDC/10A (EPOS2 70/10). Although the motor requires a 24VDC nominal supply, Harmonic Drive guarantees that it is possible to operate it using a higher power supply (such as 33.6VDC), as long as the Maxon driver is correctly configured so as to ensure that the motor performance features (speed, current, torque, etc.) are not exceeded from the given values at the rating Table A.1.

Aiming to permit the highest possible current, four drivers EPOS2 70/10 P/N 375711 (Figure 3.18b) were selected to actuate the four joints each. Table A.6 summarizes the driver main specifications, and Figure-b depicts its main connections.

A.3 Market search of force sensor

For both TETIS tasks, the end-effector pose should be controlled so that it remains always in contact with the surface and perpendicular to it. This requires the mea-

Table A.6: Maxon Motor EPOS2 70/10 Driver - Specifications

Item/Model	Maxon Motor EPOS2 70/10
TETIS joints	All
Power supply V_{in}	$11 - 70VDC$
Output voltage to motor	$0.9V_{in}$
Nominal output current	$10A$
Peak output current	$25A$
Efficiency	94%

surement of the forces in x, y, z-axes, without the needing of torque measurement. Interaction tasks may not exceed $1kgf$, hence, we require a minimum of $10N$ measuring capacity. In addition, the force sensor should be as lightweight as possible so as to not add too much weight to the manipulator, since it is limited to only 5 kg.

A.3.1 Market search

An extensive market search was performed to assist the selection of a suitable model to be included in TETIS. A summary of the most highlighted commercially available models is shown in Table A.7.

A.3.2 Market analysis

All FlexiForce models (Figure A.4a) are extremely small, lightweight, flexible and inexpensive items, which would make then mechanically perfect for this application. However, each one is a piezoelectric sensor that measure only force in 1 axis. This brings out two problems: (i) we should use three of them arranged into a complex structure to allow the measurement of three axes perpendicular to each other (x,y,z); (ii) piezoelectric sensors are not indicated for high-precision applications. FUTEK MAU300 Stick and Robotiq FT 150 (Figure A.4b) are also not suitable, since they are heavy and only measures 2 axes (in case of FUTEK).

JR3 sensors are capable of measuring not only forces in the 3 axes, but also torques. Yet, they are also very expensive and heavy to this application. In average, ATI-IA sensors (Figure A.4c) are lightweight, small, capable of measuring 6-axes (force/torque), and offer several interesting features (such as many communication interfaces as certified IP protection). However, they are extremely expensive, and our application does not require torque measuring, which means that their purchase

Table A.7: Force sensors - Market search (May 2015)

Model/Item	Range	Dim. (mm)	Weight	Type	Cost	Comments
Tekscan FlexiForce A201	F_z : 111N	0.2 × 191 × 14	Negligible	Piezoelectric 1 axis	\$65.00	∗ Also in various lengths ∗ Also in 4 and 445N
Optoforce OMD-10-SE-10N	F_z : 10N $F_{x,y}$: 5N	10ϕ	3g	Infrared 3 axes (force)	750.00 €	∗ Also in 20mm/60N, 30mm/100N/600N ∗ Rounded surface ∗ CAN, UART, USB, ∗ Ethernet, EtherCAT
Optoforce OMD-20-FE-200N	F_z : 200N $F_{x,y}$: 30N	25 × 25 × 16	23g	Infrared 3 axes (force)	1100.00 €	∗ Also in 600 and 1600N ∗ Flat surface ∗ CAN, UART, USB, ∗ Ethernet, EtherCAT
Optoforce HEX-80-RE-1200	F_z : 1200N $F_{x,y}$: 300N $T_{x,y,z}$: 5Nm	80 × 80 × 35	N/A	Infrared 6 axes (force/tq.)	2000.00 €	∗ Also in 350/3200N ∗ Flat surface ∗ CAN, UART, USB, ∗ Ethernet, EtherCAT
Robotiq FT 150	$F_{x,y,z}$: ±150N $T_{x,y,z}$: 15Nm	120ϕ (outer) 45ϕ (inner) 37.5 (thick.)	650g	Load cell 6 axes (force/tq.)	N/A (but high)	∗ Ubuntu/Linux ROS compatible ∗ RS232/RS485/USB
ATI-IA Nano17	$F_{x,y}$: ±50N F_z : ±70N $T_{x,y}$: ±0.5Nm	17ϕ 14.5 (height)	9.07g	Strain gauge 6 axes (force/tq.)	\$6790.00 +\$2000 (interface)	∗ Optional: 16-bit DAQ (PCI, PCIe, PXI, USB, RS232, Ethernet, CAN Bus, DeviceNet
ATI-IA Nano17 Titanium	$F_{x,y}$: ±32N F_z : ±56.4N $T_{x,y}$: ±0.2Nm	17ϕ 14.5 (height)	10.1g	Strain gauge 6 axes (force/tq.)	\$4300.00 +\$2000 (interface)	∗ Same as above
ATI-IA Nano17 IP Protection	$F_{x,y}$: ±50N F_z : ±70N $T_{x,y}$: ±0.5Nm	20.1ϕ 22.2 (height)	40.8g	Strain gauge 6 axes (force/tq.)	\$5350.00 (IP65) \$6400.00 (IP68) +\$2000 (interface)	∗ Same as above ∗ IP protection
ATI-IA Nano25	$F_{x,y}$: ±250N F_z : ±1000N $T_{x,y}$: ±6Nm	25ϕ 21.6 (height)	63.4g	Strain gauge 6 axes (force/tq.)	\$4250.00 +\$2000 (interface)	∗ Same as above
ATI-IA Mini27 Titanium	$F_{x,y}$: ±80N F_z : ±160N $T_{x,y}$: ±4Nm	27ϕ 18.2 (height)	33.4g	Strain gauge 6 axes (force/tq.)	N/A (but high)	∗ Same as above
JR3 30E15A4-I40-EF	$F_{x,y}$: ±40N F_z : ±80N $T_{x,y,z}$: ±3.1Nm	76ϕ 38.1 (thick.)	280g	Load cell 6 axes (force/tq.)	\$7184.00 (dig.) \$6589.10 (ana.) +\$2000	∗ Available in: 40/100/200N
JR3 67M25A3-I40-DH	$F_{x,y}$: ±100N F_z : ±200N $T_{x,y,z}$: ±6.3Nm	67ϕ 25 (thick.)	175g	Load cell 6 axes (force/tq.)	\$5671.50 (dig.) \$5130.60 (ana.) +\$2000	∗ Available in: 100/200N
FUTEK MAU300 Stick Shift Sensor (Automotive)	$F_{x,y}$: ±44.5N	38.1ϕ 76.2 (thick.)	280g	Load cell 2 axes (force)	\$2100.00	-

would include unnecessary cost.

Optoforce sensors (Figure A.4d) seemed to be the best overall choice, since they are, in average, small, very lightweight, CAN compatible, and their measuring principle is based on infrared light. Among the Optoforce models, there are 3-axes and 6-axes and round and flat surfaces.

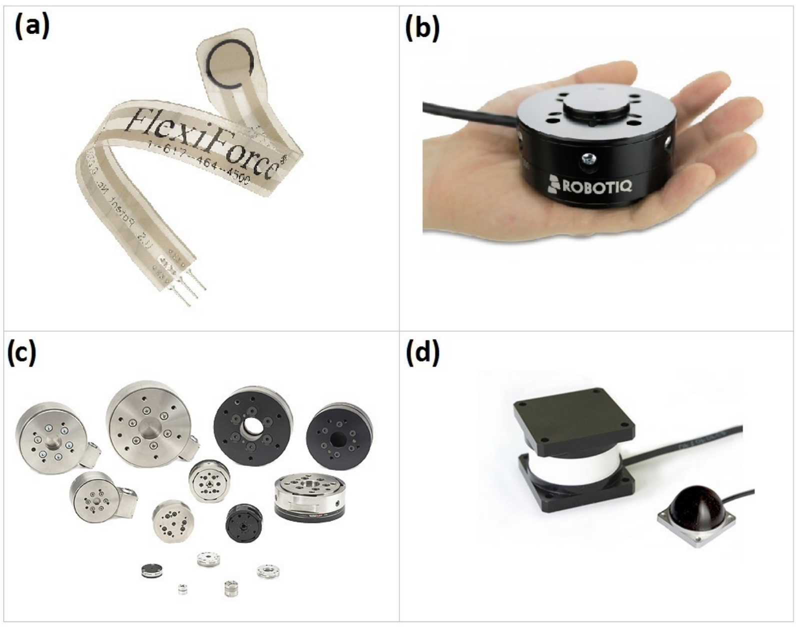


Figure A.4: Commercial force sensors: (a) Tekscan FlexiForce; (b) Robotiq FT 150; (c) ATI-IA; (d) Optoforce.