The University of Birmingham

Doctoral Thesis

Vision-based trajectory control of unsensored robots to increase functionality, without robot hardware modification

Author:
Valerio Ortenzi

Supervisors:
Dr. Rustam Stolkin
Dr. Michael Mistry

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

School of Engineering, Department of Mechanical Engineering

March 19, 2017

Declaration of Authorship

I, Valerio Ortenzi, declare that this thesis titled, 'Vision-based trajectory control of unsensored robots to increase functionality, without robot hardware modification' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: ____________________________

Date: ____________________________

UNIVERSITY OF BIRMINGHAM

Abstract

School of Engineering, Department of Mechanical Engineering

Doctor of Philosophy

Vision-based trajectory control of unsensored robots to increase functionality, without robot hardware modification

by Valerio Ortenzi

In nuclear decommissioning operations, very rugged remote manipulators are used, which lack proprioceptive joint angle sensors. Hence these machines are simply tele-operated, where a human operator controls each joint of the robot individually using a teach pendant or a set of switches. Moreover, decommissioning tasks often involve forceful interactions between the environment and powerful tools at the robot's end-effector. Such interactions can result in complex dynamics, large torques at the robot's joints, and can also lead to erratic movements of a mobile manipulator's base frame with respect to the task space. This Thesis seeks to address these problems by, firstly, showing how the configuration of such robots can be tracked in real-time by a vision system and fed back into a trajectory control scheme. Secondly, the Thesis investigates the dynamics of robot-environment contacts, and proposes several control schemes for detecting, coping with, and also exploiting such contacts. Several contributions are advanced in this Thesis. Specifically a control framework is presented which exploits the constraints arising at contact points to effectively reduce commanded torques to perform tasks; methods are advanced to estimate the constraints arising from contacts in a number of situations, using only kinematic quantities; a framework is proposed to estimate the configuration of a manipulator using a single monocular camera; and finally, a general control framework is described which uses all of the above contributions to servo a manipulator. The results of a number of experiments are presented which demonstrate the feasibility of the proposed methods.

Acknowledgements

A doctorate is a long journey, full of good days and of bad days. I am lucky to have people around me giving me constant support and encouragement.

I start thanking my beloved parents. They have been rock solid, giving me strength and support all the way through this journey. They had to stand by me when I was very irritable. Not to mention angry and disheartened. They have always been there for me, no matter what. I love you.

I want to thank a very special person. An uncommonly good man, a man I have the pleasure to call my brother. Mark, you have been very kind to me. You are so exceptional and outstanding. I am elated to have found a brother in you. We are family now. I love you.

To continue thanking my family, I want to mention Graziano and Anna and their Leo and Matteo, my aunt Gina and my cousin Damiano.

A distinguished mention for John and Wilson, two people very dear to me. A big thank you to all the Flamengo lads.

I want to thank Laura, Miren, Elisa, Manolis, Chie and Vijay for having been good company during this adventure.

Last but not the least, I want to thank Rustam and Mike for believing in me all the way. Also, many thanks to Naresh, Morteza and Hsiu-Chin for their help and supervision.

Finally, huge thanks to Yasemin, a colleague, a friend, and most of all a very special person.

This work was funded by a bursary from the UK Nuclear Decommissioning Authority, and was administered and co-supervised by Lorraine McDermott, Jeff Kuo and Jim Harken of UK National Nuclear Laboratory. I am also grateful to Sellafield Ltd for enabling me to visit the Sellafield nuclear site, and providing inspiration and encouragement for this project.

Contents

List of Figures

List of Tables

Symbols

$\mathbf{q}$	Robot configuration coordinates, joint position, $\boldsymbol{q} \in \mathcal{C} \subseteq \mathcal{R}^n$
$\dot{\mathbf{q}}$	Joint velocity
$\ddot{\mathbf{q}}$	Joint acceleration
$\mathbf{y}$	Robot task coordinates, $\mathbf{y} \in \mathcal{W} \subseteq \mathcal{R}^m$
$\dot{\mathbf{y}}$	Task coordinates velocity
$\ddot{\mathbf{y}}$	Task coordinates acceleration
$\mathbf{J}(\mathbf{q})$	Task Jacobian
$\dot{\mathbf{J}}(\mathbf{q})$	Task Jacobian first derivative
$\mathbf{J}^T(\mathbf{q})$	Task Jacobian transpose
$\mathbf{J}^\dagger(\mathbf{q})$	Task Jacobian Moore-Penrose pseudoinverse
$\mathbf{M}(\mathbf{q})$	Inertia matrix
$\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}})$	Centrifugal, Coriolis, gravity and friction components in robot dynamics
$\boldsymbol{\tau}$	Torque input, torque commands to the joint motors
$\mathbf{J}_C(\mathbf{q})$	Jacobian of the constraints
$\dot{\mathbf{J}}_C(\mathbf{q})$	First derivative of the Jacobian of the constraints
$\mathbf{J}_C(\mathbf{q})^T\boldsymbol{\lambda}$	Torques at the joint level which are the effect of external forces
$\mathbf{F}$	Cartesian external forces/torques at the end effector
$\boldsymbol{\Lambda}(\mathbf{y})$	Operational space inertia matrix
$\boldsymbol{\mu}(\mathbf{y}, \dot{\mathbf{y}})$	End effector centrifugal and Coriolis forces
$\mathbf{p}(\mathbf{y})$	End effector gravity forces
$\tilde{\boldsymbol{\mu}}(\mathbf{y}, \dot{\mathbf{y}})$	End effector centrifugal, Coriolis, gravity and friction forces
$\mathbf{M}_C(\mathbf{q})$	Projected dynamics "inertia" matrix
$\boldsymbol{\Lambda}_C(\mathbf{y})$	Operational space projected dynamics inertia matrix
$\mathbf{K}$	General feedback control gain
$\mathbf{K}_A$	Admittance control force feedback gain
$\mathbf{K}_S$	Joint stiffness matrix
$\boldsymbol{\delta}$	Small displacement
$\mathbf{S}$	Selection matrix, indicating directions where free motion is not allowed
$(\mathbf{I} - \mathbf{S})$	Selection matrix, indicating directions where free motion is allowed

$\mathbf{P}$	Projector into the space of permissible motion
$(\mathbf{I}-\mathbf{P})$	Projector into the space of permissible forces

Dedicated to my parents.

Chapter 1

Introduction

1.1 Overview

In several nuclear sites in the UK, as well as important nuclear sites world-wide, such as ongoing work at the Fukushima Daiichi nuclear disaster site, very rugged remote manipulators are used, which lack proprioceptive joint angle sensors. It is not considered feasible to retrofit proprioceptive sensors to such robots: firstly, electronics are vulnerable to gamma and beta radiation; secondly, for nuclear applications, the installation of new sensors on trusted machinery would compromise long-standing certification; thirdly, such robots are predominantly deployed on a mobile base platform and typically use powerful hydraulic drilling and cutting tools at the end-effector. Even if the robot had proprioceptive sensors, such tools cause large and frequent perturbations to the base frame, so that proprioceptive sensors would still be unable to obtain the robot's pose with respect to a task frame set in the robot's surroundings. For these reasons, the adoption of external sensors, such as cameras, offers a means of closing the control loop with quantitative feedback, enabling advanced trajectory control and increased autonomy which are currently not possible.

At present, these kinds of remote manipulator machines used for decommissioning tasks on nuclear sites are simply tele-operated, where a human operator controls each joint of the robot individually using a teach pendant or a set of switches, as in Fig. 1.1. In some cases the human operator controls the robot via CCTV cameras or from behind thick lead glass windows, with very limited depth perception and situational awareness. In other cases, the operator must be positioned near to the robot, wearing protective clothing and breathing apparatus. This not only means that task performances are sub-optimal, but also that humans are being exposed to risk in hazardous environments. This work contributes a step towards increased autonomy of such tasks, including precise automatic

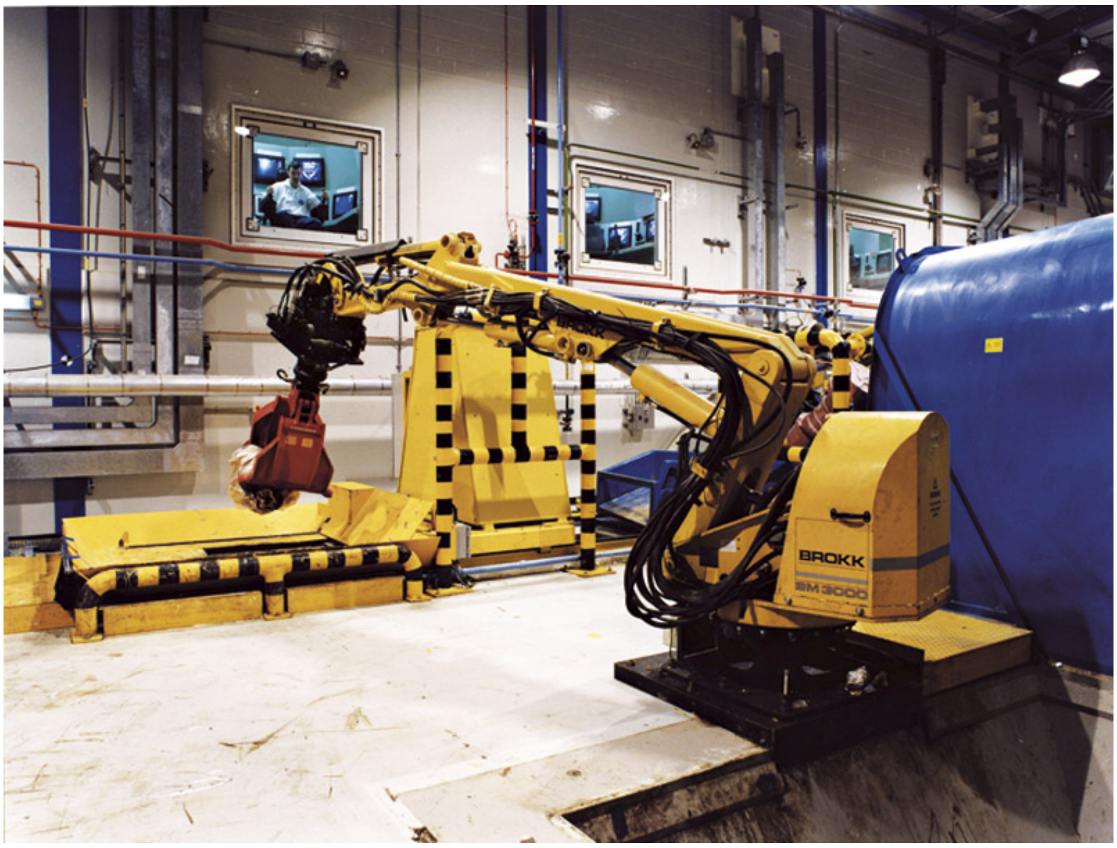


FIGURE 1.1: A BROKK robot, equipped with a gripper, being used for a pick and place task at the Sellafield nuclear site in UK. This robot has no sensors for measuring joint angles, and so must be directly controlled by a human operator, who pushes a separate switch for each joint and judges the robot's pose by eye. The human operator can be seen controlling the robot from behind a 1.6m thick lead glass window which shields against radiation. For more examples, refer to www.sellafieldsites.com/solution/decommissioning/.

control which is currently almost completely lacking in such industries. Removal of the human factor from such environments could improve safety, and also improve the speed and precision of task performance.

Specifically, this work aims at providing reliable quantitative feedback on the robot configuration by tracking several parts of the robot in monocular camera images. In this context, we present a framework whose main component estimates the robot configuration by enforcing the equality between a set of transformation matrices relating frames set in the camera, world and the tracked robot parts. The joint configuration is estimated by combining this tracked information along with the robot's kinematic model. To solve this estimation problem, we present and compare two different types of non-linear optimisation schemes. In addition to estimating the robot state, we also show how these estimations can be successfully used as quantitative feedback to a classical kinematic controller, in order to make the robot achieve a desired end-effector position.

The vast majority of robots in world-wide research laboratories possess joint encoders, which is why the larger part of previous robotics literature (including the visual servoing literature) assumes knowledge of joint angles. In contrast, we believe our approach of estimating the robot configuration by using only monocular camera images, represents both novelty and substantial usefulness for robotics applications in harsh environments. Additionally, the methods proposed in this Thesis may have wider applications to other problems, such as: human-robot or robot-robot interaction; articulated robot calibration; use of a remote camera for servoing of mobile manipulator platforms with respect to surrounding objects.

Robots used in the nuclear industry have powerful interactions with the environment in many tasks (*e.g.*, scabbling, where an arm must move a grinding tool across a wall or floor to a precise depth, in order to remove contaminated surface material). In general, as robotics becomes progressively pervasive in real-world applications, it is becoming increasingly necessary for robots to be aware of what is happening around them and to interact with their environments and other agents. Firstly, these interactions should be safe and cause no damage to the robot and to its surroundings. Secondly, and perhaps more interestingly, robots should ideally understand and exploit their environments to perform tasks more efficiently. Humans often exploit contacts to perform tasks more comfortably and more efficiently. For example, a person reaching to grasp an object further off on a table, usually leans on the table to support their weight during reaching and grasping. Alternatively, imagine an elderly person sweeping the floor, while also leaning on the broom handle to partially support their weight. In such cases, the human is exploiting contacts with the environment to reduce the required muscular effort. Note that, in the second example, the person is exploiting the same contact that is already required by the task itself (moving a broom so that its trajectory remains in contact with the floor surface).

For these reasons, this work also analyses such contact usage and is intended as an early step towards making better use of contacts in robotics, and enabling robots to mimic useful human strategies for exploiting such contacts. In order to replicate or at least understand this exploitation of the contacts, dynamics play a dominant role, not only because of the involvement of force and torque exchanges, but also because the robot might need to perform tasks with certain velocities and accelerations. Specifically, we use a controller based on projected dynamics and we formulate tasks both in the joint space and in the Cartesian space. Furthermore, we present methods to estimate the constraints arising from contacts analysing only kinematic quantities, *i.e.*, velocities. Thus, there is no need for either a force/torque sensor or tactile sensors. Also, it is possible to use the configuration estimations to compute these constraints, since they provide information on the kinematics of the robot.

1.2 Contributions of this Thesis

Hence several contributions are advanced in this Thesis. Specifically:

- a control framework is presented which exploits the constraints arising at contact points to effectively reduce commanded torques to perform tasks;
- methods are advanced to estimate the constraints arising from contacts in a number of situations, using only kinematic quantities;
- a framework is proposed to estimate the configuration of a manipulator using a single monocular camera;
- a general control framework is described which uses all of the above contributions to servo a manipulator.

1.3 Publications arising from this Thesis

The Chapters of this Thesis are extracted from a number of publications:

[1] V. Ortenzi, M. Adjigble, J. Kuo, R. Stolkin, and M. Mistry: An experimental study of robot control during environmental contacts based on projected operational space dynamics. In 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids), pages 407 - 412. IEEE, 2014 (Chapter 4);

[2] V. Ortenzi, R. Stolkin, J. A. Kuo, and M. Mistry: Projected inverse dynamics control and optimal control for robots in contact with the environment: A comparison. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015 (Chapter 4);

[3] V. Ortenzi, H-C. Lin, M. Azad, R. Stolkin, J.A. Kuo, and M Mistry: Kinematics-Based Estimation of Contact Constraints Using Only Proprioception. In Humanoids 2016: IEEE-RAS International Conference on Humanoid Robots, 2016 (Chapter 5);

[4] V. Ortenzi, N. Marturi, R. Stolkin, J. A. Kuo, and M. Mistry: Vision-guided state estimation and control of robotic manipulators which lack proprioceptive sensors. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016 (Chapter 6).

1.4 Layout of this Thesis

This Thesis is structured as follows: Chapter 2 introduces theory and symbols used in this Thesis. A literature review on the topics most connected to this work is reported in Chapter 3. A special emphasis is put on the analysis of the decoupling of motion control and force control. Chapter 4 presents a control framework which accounts for the contacts and uses the constraints to lower the commands to perform tasks. Experiments are reported where a robot is asked to wipe a board and such control framework succeeds in performing the task while requesting less torques due to an exploitation of the constraints. Chapter 5 describes methods to estimate contact constraints using kinematics. Results of experiments both in simulation and on a robot show how these methods can be successfully applied to a variety of situations. Chapter 6 describes a framework to estimate the state of a manipulator using monocular vision. Experiments show that those estimates can be effectively used as feedback in a kinematic controller. Chapter 7 presents a general framework which uses all of the contributions described in the previous Chapters. And finally, Chapter 8 concludes this Thesis with final remarks and considerations.

Chapter 2

Background

Robots have to perform a large range of tasks, and most often to achieve quality performances those robots have to be reactive, quick and efficient. From such perspective, dynamics play an important role, especially when a robot has to perform tasks involving contacts and interactions with the environment, *i.e.*, external systems such as human beings, other robots and objects. Furthermore, when high speed is desired, the dynamics of the robot must be taken into account to achieve a sufficient degree of quality in task performance and completion. For this purpose, this Chapter reviews general concepts around the kinematics and the dynamics of a robot.

2.1 Kinematics

A robot is assumed with n degrees of freedom operating in an m-dimensional workspace $\mathcal{W}$. Let $\mathbf{q}$ be the robot configuration taking values in an n_q-dimensional configuration space $\mathcal{C}$.

A robot task is expressed in coordinates $\mathbf{y}$ and takes values in an m_y-dimensional space $\mathcal{Y}$. Task and robot configuration are related through the kinematic map

$$\mathbf{y} = \mathbf{f}(\mathbf{q})\,. \tag{2.1}$$

Let $\ddot{\mathbf{q}}$ and $\dot{\mathbf{q}}$ refer respectively to accelerations and velocities of the robot configuration. Then the differential kinematics relations can be expressed as

$$\dot{\mathbf{y}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{2.2}$$

and

$$\ddot{\mathbf{y}} = \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{J}}(\mathbf{q})\dot{\mathbf{q}}\,, \tag{2.3}$$

respectively for velocities and accelerations, where $\mathbf{J}(\mathbf{q}) = \frac{\partial \mathbf{y}}{\partial \mathbf{q}}$ is the task Jacobian.

2.2 Dynamics

Classically, robot dynamics are expressed in the joint space as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau} - \mathbf{J}_C(\mathbf{q})^T\boldsymbol{\lambda}\,, \tag{2.4}$$

$$\mathbf{J}_C(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0} \tag{2.5}$$

and

$$\mathbf{J}_C(\mathbf{q})\ddot{\mathbf{q}} = -\dot{\mathbf{J}}_C(\mathbf{q})\dot{\mathbf{q}}\,, \tag{2.6}$$

where $\mathbf{M}(\mathbf{q})$ is the inertia matrix, $\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}})$ contains centrifugal and Coriolis terms as well as gravity and friction components while $\boldsymbol{\tau}$ and $\mathbf{J}_C(\mathbf{q})^T\boldsymbol{\lambda}$ are respectively the torques exerted by the robot motors and the effect of external forces due to contacts and constraints. $\mathbf{J}_C(\mathbf{q})$ represents the relationship between the constraints and the change of configuration $\dot{\mathbf{q}}$, *i.e.*, it is the Jacobian of the constraints. And Eq. 2.6 provides the same relation at the acceleration level.

2.3 Operational Space Dynamics

In [5] and [6], Khatib proposed a formulation of robot dynamics in the operational space, and this formulation lends itself to a direct definition of motions and forces in the workspace of the robot. To express the robot dynamics in the operational space, multiply Eq. 2.4 by $\mathbf{J}(\mathbf{q})\mathbf{M}(\mathbf{q})^{-1}$, substitute $\mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} = \ddot{\mathbf{y}} - \dot{\mathbf{J}}(\mathbf{q})\dot{\mathbf{q}}$ from Eq. 2.3, and finally use

$$\boldsymbol{\tau} = \mathbf{J}(\mathbf{q})^T\mathbf{F}\,. \tag{2.7}$$

Then, the operational space formulation of dynamics is the following

$$\boldsymbol{\Lambda}(\mathbf{y})\ddot{\mathbf{y}} + \boldsymbol{\mu}(\mathbf{y}, \dot{\mathbf{y}}) + \mathbf{p}(\mathbf{y}) = \mathbf{F}\,, \tag{2.8}$$

where $\boldsymbol{\Lambda}(\mathbf{y}) = (\mathbf{J}(\mathbf{q})\mathbf{M}(\mathbf{q})^{-1}\mathbf{J}(\mathbf{q})^T)^{-1}$ is the operational space inertia matrix, $\boldsymbol{\mu}(\mathbf{y}, \dot{\mathbf{y}})$ the vector of end effector centrifugal and Coriolis forces, $\mathbf{p}(\mathbf{y})$ the vector of gravity forces, and $\mathbf{F}$ is the vector of external forces applied at the end effector. Simplified, and with a slight stretch of notation, operational space dynamics can be expressed as

$$\mathbf{\Lambda}(\mathbf{y})\ddot{\mathbf{y}} + \tilde{\boldsymbol{\mu}}(\mathbf{y}, \dot{\mathbf{y}}) = \mathbf{F}\,, \tag{2.9}$$

where $\tilde{\boldsymbol{\mu}}(\mathbf{y}, \dot{\mathbf{y}}) = \boldsymbol{\mu}(\mathbf{y}, \dot{\mathbf{y}}) + \mathbf{p}(\mathbf{y})$.

Alternatively, operational space dynamics can be expressed as in [7]

$$\mathbf{\Lambda}(\mathbf{y})\ddot{\mathbf{y}} + \mathbf{\Lambda}(\mathbf{y})(\mathbf{J}(\mathbf{q})\mathbf{M}(\mathbf{q})^{-1}\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) - \dot{\mathbf{J}}(\mathbf{q})\dot{\mathbf{q}}) = \mathbf{F}\,. \tag{2.10}$$

The formulations in [5], [6] and in [7] are equivalent.

In the following, dependencies on $\mathbf{y}$ and $\mathbf{q}$ will be dropped for the sake of notational simplicity. Notice also that hereinafter equations are taken from the cited papers but symbols are readapted to be consistent throughout all this work and to have a common general notation, thus enabling direct comparisons.

Khatib [6] expresses the control equation for redundant manipulators as

$$\boldsymbol{\tau} = \mathbf{J}^T\mathbf{F} + (\mathbf{I} - \mathbf{J}^T\mathbf{J}^{T\#})\boldsymbol{\tau}_0\,, \tag{2.11}$$

where $\mathbf{J}^{T\#} = (\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)^{-1}\mathbf{J}\mathbf{M}$ is intended to decouple the motion generated by $\boldsymbol{\tau}_0$ from having an impact on task dynamics.

2.4 Projected Dynamics

Projected dynamics in its basic formulation is found in [8]. A projector $\mathbf{P}$ is defined as

$$\mathbf{P} = \mathbf{I} - \mathbf{J}_C^{\dagger}\mathbf{J}_C\,. \tag{2.12}$$

$\mathbf{P}$ is an operator that projects into the null space of the constraints which limit the free motion space of the robot. Also, $\mathbf{P}$ has the property that

$$\mathbf{P}\mathbf{J}_C^T\boldsymbol{\lambda} = \mathbf{0}\,, \tag{2.13}$$

i.e., it is possible to remove the explicit presence of constraint forces from the dynamics equation (Eq. 2.4) by projecting the same dynamics equation with $\mathbf{P}$, which becomes

$$\mathbf{P}\mathbf{M}\ddot{\mathbf{q}} + \mathbf{P}\mathbf{h} = \mathbf{P}\boldsymbol{\tau}\,. \tag{2.14}$$

Since admissible velocities $\dot{\mathbf{q}}$ must belong to the null space of the Jacobian of the constraints, as in Eq. 2.5, then Eq. 2.5 can be also written using the projector $(\mathbf{I} - \mathbf{P})$ as

$$(\mathbf{I} - \mathbf{P})\dot{\mathbf{q}} = \mathbf{0}\,, \tag{2.15}$$

which differentiated over time holds

$$(\mathbf{I} - \mathbf{P})\ddot{\mathbf{q}} = \mathbf{C}\dot{\mathbf{q}}\,, \tag{2.16}$$

where $\mathbf{C} = \dot{\mathbf{P}} = -\mathbf{J}_C^{\dagger}\dot{\mathbf{J}}_C$.

Recognising that Eq. 2.14 and Eq. 2.16 are orthogonal and using a trick presented in [8], we multiply Eq. 2.16 by $\mathbf{M}$ and add it to Eq. 2.14, to obtain

$$\mathbf{M}_C\ddot{\mathbf{q}} = \mathbf{P}\boldsymbol{\tau} - \mathbf{P}\mathbf{h} + \mathbf{C}_C\dot{\mathbf{q}}\,, \tag{2.17}$$

where $\mathbf{M}_C = \mathbf{M} + \mathbf{P}\mathbf{M} - (\mathbf{P}\mathbf{M})^T$, $\mathbf{C}_C = \mathbf{M}\mathbf{C}$ and $\mathbf{C}\dot{\mathbf{q}} = -\mathbf{J}_C^{\dagger}\dot{\mathbf{J}}_C\dot{\mathbf{q}}$. However, the constraint inertia matrix $\mathbf{M}_C$ does not have a unique definition. Since $\mathbf{M}_C$ is invertible, it is possible to explicitly express accelerations and forces as

$$\ddot{\mathbf{q}} = \mathbf{M}_C^{-1}(\mathbf{P}\boldsymbol{\tau} - \mathbf{P}\mathbf{h} + \mathbf{C}_C\dot{\mathbf{q}}) \tag{2.18}$$

and

$$\mathbf{J}_C^T\boldsymbol{\lambda} = (\mathbf{I} - \mathbf{P})\boldsymbol{\tau} - (\mathbf{I} - \mathbf{P})\mathbf{h} - \boldsymbol{\mu}(\mathbf{P}\boldsymbol{\tau} - \mathbf{P}\mathbf{h} - \mathbf{C}_C\dot{\mathbf{q}})\,, \tag{2.19}$$

where $\boldsymbol{\mu} = (\mathbf{I} - \mathbf{P})\mathbf{M}\mathbf{M}_C^{-1}$. Specifically, the generalised torques $\mathbf{J}_C^T\boldsymbol{\lambda}$ can always be uniquely obtained, while this may not be true for $\boldsymbol{\lambda}$. It is true if and only if the Jacobian matrix is full rank.

Hence, by projecting the dynamics and so the commanded torques through $\mathbf{P}$ into the space orthogonal to the constraints, we can control the robot regardless of the external forces. For this reason this approach does not need a force sensor at contact points, in contrast to techniques such as hybrid force/torque control. This represents a step towards developing algorithms for contact control without force sensors which are critical for the control of sensor-deficient robots such as in nuclear industry, where it can be difficult to shield sensor electronics against radiation. Moreover, by exploiting the constraints, it is thus possible to reduce the commanded torques needed to execute the task. Since $\mathbf{P}$ is a projector, for all vectors $\mathbf{v}$

$$|\mathbf{P}\mathbf{v}| \leq |\mathbf{v}| \tag{2.20}$$

holds. This means that the commanded torques will be such that

$$|\mathbf{P}\boldsymbol{\tau}| \leq |\boldsymbol{\tau}| \,. \tag{2.21}$$

This means that $\mathbf{P}$ can be used to effectively reduce the commanded torques.

Using the projector $\boldsymbol{P}$, we can define a controller as in [7] and [1], in the form

$$\boldsymbol{\tau} = \mathbf{P}\boldsymbol{\tau}_m + (\mathbf{I} - \mathbf{P})\boldsymbol{\tau}_a + \mathbf{P}\boldsymbol{\tau}_n \,, \tag{2.22}$$

such that $\mathbf{P}\boldsymbol{\tau_m}$ is responsible for the motion part of the task, $(\mathbf{I} - \mathbf{P})\boldsymbol{\tau}_a$ is responsible for the active force, *i.e.*, the force exerted by the robot end effector on the contact point and does not generate accelerations in the free space, and finally $\mathbf{P}\boldsymbol{\tau}_n$ is a null-space component which can be used in order to achieve some desired behaviour, *e.g.*, to minimise joint drift from the initial configuration. This last component is available only if the robot is redundant with respect to the task, *i.e.*, $n > m$.

The feature of having separate control components which separately affect motion and force control, is called decoupling and has been studied extensively in literature. The reader is referred to Section 3.1 for more details.

In [7], the formulation of operational space dynamics with constraints is expressed as

$$\boldsymbol{\Lambda}_C\ddot{\mathbf{y}} + \boldsymbol{\Lambda}_C(\mathbf{J}\mathbf{M}_C^{-1}\mathbf{P}\mathbf{h} - (\dot{\mathbf{J}} + \mathbf{J}\mathbf{M}_C^{-1}\mathbf{C})\dot{\mathbf{q}}) = \mathbf{F} \,, \tag{2.23}$$

where $\boldsymbol{\Lambda}_C = (\mathbf{J}\mathbf{M}_C^{-1}\mathbf{P}\mathbf{J}^T)^{-1}$. The reader is referred to [7] for the derivation of this last equation.

Chapter 3

Literature Review

The goal of this work is to increase the functionality of under-sensored robots. Specifically, several contributions are made. A control framework is presented which exploits contacts. Also, methods to estimate kinematic constraints are advanced. Moreover, a framework relying on a single monocular camera is proposed to estimate the configuration of the robot. Finally, all these components are integrated in a general control framework.

These elements span a wide range of topics such as robot dynamics, projected dynamics, hybrid motion/force control, contact constraint estimation, visual tracking, and visual servoing. For the sake of clarity, this Chapter is divided into three Sections.

Sec. 3.1 reports related work on topics regarding robot dynamics and control. This Section also reviews hybrid motion/force control, a control scheme that enables robots to perform tasks involving both motion, in the free space, and interactive force, at the contacts. An emphasis is placed on the study of the decoupling of motion and force control. Empirical experiments with a simulated 3-degree-of-freedom manipulator compare the performance of a controller based on projected dynamics and a controller based on a selection matrix which defines directions of free motion and directions of force control in the Cartesian space, which are representative of two competing approaches that are prevalent in the literature. The results of these experiments on a pure motion task and on a task involving force control alongside motion confirm that it is indeed possible to achieve a complete decoupling, but we also show how this feature can be relaxed or sacrificed in order to reduce the robot's joint torques while still completing the task.

Differently, Sec. 3.2 concentrates on methods to estimate kinematic constraints due to contacts with the environment. Particularly, while we propose a method based on the analysis of a set of observed velocities which differ from the expected velocities during an explorative phase, most of the approaches in literature are based either on force sensors at the contact point or on vision.

Finally, Sec. 3.3 zooms in on visual tracking and visual servoing, with an emphasis also on the problem of pose estimation. Our framework to estimate the state of an under-sensored manipulator is finalised to produce quantitative feedback to be used in a controller, and this represents the main difference with classical visual servoing approaches. However, we make use of visual tracking as the first component of our framework, and this motivates a review of classical visual tracking and visual servoing techniques.

3.1 Robot Dynamics and Control

The utilisation of robots for tasks which involve forceful interactions with the environment is rapidly increasing, for example in machining and assembling with industrial robots; in tele-operation and tele-manipulation where robots have to act on the environment; in search and rescue tasks; in cooperative tasks where multiple agents are involved (robots, and/or humans). In manipulation, hands have to touch, grasp and place objects; humanoid robots touch the ground walking; in the nuclear industry, robots are used for decommissioning tasks, such as cutting or scabbling, which require large contact forces at precise positions.

Thus controlling the interactions robots have or might have with the environment is critical to executing a range of tasks, not limited only to the well-known peg-in-the-hole and surface/contour tracking problems. In these situations, contacts play a fundamental role for various reasons, not only limited to safety (often interpreted as safe interaction with the environment and the other agents), but also a desired exploitation of such contacts to minimise the effort needed to complete a task. Moreover, when robots have to act in cooperation with humans, it may be useful for them to mimic human behaviour as to have more readable actions in cooperative tasks.

To achieve high performance in task accomplishment and task quality, it is necessary to account for the dynamics of the robot. Robots have masses and inertias which are neither negligible nor unimportant when in motion. Additionally, reactive forces arising from contacts with the environment and active forces exerted on the environment need to be analysed and controlled in addition to the robot's position. Such specifications require a deep understanding of the role of dynamics and of the interactions of the dynamics of the robot with the environment. As an example, humans profit from contacts and perform tasks more efficiently exploiting such contacts, *e.g.*, a person reaching to grasp an object further off on a table, usually leans on the table surface to support their weight during reaching and grasping. A deeper understanding and a better exploitation of the dynamics of the contacts will make such abilities available for robots as well.

We introduced the basics of robot kinematics and dynamics in Sec. 2. See also [9] for a review of key robot dynamics concepts, and [10] and [11] for a general introduction to dynamics theory.

Compliance with the environment can be achieved with the introduction of mechanical devices at the robot end effector, *e.g.*, Remote Centre of Compliance devices as in [12]. This passive compliance can be very useful in peg-in-the-hole tasks but has limited use for more complex tasks where the robot is asked to interact actively with the environment, *e.g.*, pushing a cart or grasping an object. In contrast to passive solutions, an active robot compliance can be achieved with a control law so as to directly command the robot to act on the environment with a desired behaviour. Control methods such as active stiffness and impedance control fall into this category. In particular, in active stiffness control, contact forces depend on the position and the robot is controlled so that it behaves like a spring. Impedance control is another interaction control technique that regulates the interaction between robot and environment assuming the environment as an admittance (for example a mass) and outputting a force. The opposite approach is generally called admittance control.

Classically, there are several control techniques: explicit control of forces and torques, active stiffness or damping control, impedance control and finally hybrid force/motion control, just to name a few of them. Alongside these classical approaches, related work addresses modifications in the overall control architecture. As examples, Nicosia and Tomei [13] present a stability analysis of the use of a velocity observer in dynamic control for manipulators, while Hsu et al. [14] discuss the use of redundancy in dynamic control. Kröger et al. [15] combine further robot force control with compliant motion on Mason's Task Frame Formalism [16]. Luca and Manes [17] describe a modelisation of contact dynamics using a minimal parametrisation. Moreover, in addition to classical hybrid force/motion control, directions where either force or motion control may be executed are taken into account explicitly.

Motivated by the wide range of robotic applications with interactions with the environment, this Section focuses on a control technique formally defined as Hybrid Position/-Force or equivalently Motion/Force Control, a strategy that naturally defines a control for tasks involving both motion, in the free motion space, and force, at the contacts.

Books and manuals on robotics report on hybrid motion/force control, *e.g.*, [18] and [10]. Chiaverini et al. [19] present a survey on interaction schemes such as impedance control and parallel force/position control, but hybrid force/motion control is not taken directly into consideration. De Schutter et al. [20] review hybrid motion/force control, impedance control and parallel control, and propose high level statements of general value when applying force control, specifically when applying these three approaches. They highlight the differences among these approaches and suggest that there is a fundamental affinity among the schemes. We are not aware of any review article in the existing literature, which specifically and comprehensively covers the important area of hybrid motion/force control. For this reason, this Section focuses on hybrid control and aims at providing a review on the important contributions to hybrid control.

Numerous approaches for hybrid control have been proposed, over many years and in a variety of formats. We therefore propose a common, unifying mathematical notation, which enables a clearer analysis of the relationships and differences between various different methods. We use this analysis to help provide some insights into the general hybrid control framework.

Another contribution of this Section is the study of motion/force decoupling, a fundamental feature of hybrid control. Motion/force decoupling is an attribute of control schemes which enables independent controls for motion and force, without them interfering with each other. We report the results of empirical experiments, using a simulated 3-degree-of-freedom planar manipulator, to compare the performance of two representative approaches to hybrid control. Results suggest that a superior decoupling can be achieved at the cost of higher commanded torques at the joint level, while, if decoupling can be relaxed, an exploitation of the contacts can effectively reduce the commanded torques at the expense of having an interaction with the environment. We also describe an extension to this second approach that enables it to gain a complete decoupling, at the cost of losing the advantage of lowering the torque effort.

A comparative categorisation of various different hybrid control schemes is provided, according to their key attributes, such as if they need a priori knowledge or if they need any direct force feedback. Finally, we also discuss the connection between these classical control approaches and optimal control.

3.1.1 Introduction

As already stressed, when robots have to perform a task with a high speed or when there is an interaction between the robot and the environment, dynamics (of both robot and environment) play a key role for controlling the robot. Robots interacting with the environment typically have constraints that limit their motion along some directions. These constraints are due to the natural geometry of the environment and are classically called natural constraints. When robots are asked to perform a task, they are given additional artificial constraints to satisfy simultaneously to the natural constraints. It is desirable also to control the force exerted by the robot onto the environment in addition to controlling the robot's motion.

Hybrid control was conceived in an attempt to address this problem. The main principle of hybrid motion/force control is to control position in the free motion directions while controlling force in the directions that are not controlled in position, in other words, in the directions where the robot has its position constrained by contacts with the environment. In particular, the resulting controller uses all actuated joints to exert the desired active forces in the directions where free motion is not allowed because of geometrical constraints of the environment, and to realise the desired motion in the remaining directions of free motion. Several different schemes have been proposed for realising hybrid control, and in the following we will highlight the main features and differences between these approaches.

The late Seventies saw a proliferation of work in the areas of compliance and force control. This interest continued during the Eighties and consolidated in the Nineties and thereafter. We review the contribution thematically, presenting briefly compliance and admittance methods as an introduction, and then focusing only on hybrid control, categorising the methods for using either a selection matrix or a projector to realise the control. Moreover, a further discussion is presented as to whether or not methods based on the selection matrix are also in fact model-based. We define a method to be model-based if it uses a dynamic model in the controller, while the method is model-free if it uses only kinematics.

3.1.2 Compliance/Admittance methods

In 1976, to address the problem of making the robot compliant to the environment, Paul and Shimano [21] proposed the so called "free-joint" method for compliant control, where they select the joint most likely to produce compliance for each direction in the Cartesian space leaving the others for position control. In other words, a selection of directions is undertaken in the joint space.

Whitney [22] introduces force feedback control using an admittance matrix which converts force signals into desired joint velocities. This work also investigates the stability of such an approach, concluding that higher feedback gains can be applied with more compliant environments. In particular, an error is defined directly in the Cartesian space and the following control law is proposed

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}(\dot{\mathbf{y}}_d - \mathbf{K}_A\mathbf{F})\,, \tag{3.1}$$

where $\mathbf{F}$ are the forces measured with a force/torque sensor, $\mathbf{K}_A$ acts as a force feedback gain. The error $(\dot{\mathbf{y}}_d - \mathbf{K}_A\mathbf{F})$ is then converted to the joint space through the pseudo-inverse of the Jacobian (in Appendix A of the same paper). This scheme has also been referred to as admittance control. Along the lines of [22], Hogan [23] presents a three-part paper proposing the so-called impedance control, where the control reacts to disturbance to desired motion making the robot behave like an impedance instead.

In contrast to [22], Salisbury [24] investigates the possibility to devise the joint torque commands necessary to have the end effector of a manipulator behave as a spring in the Cartesian space. He introduces the concept of a joint stiffness matrix, which determines forces when the robot end effector is perturbed away from a nominal position. In directions where physical constraints are expected, it might be useful to specify low stiffness in order to minimise contact forces, while stiffness can be kept high when position references must be followed more exactly. This can be expressed as

$$\mathbf{F} = \mathbf{K}\boldsymbol{\delta}\mathbf{y}\,, \tag{3.2}$$

where $\boldsymbol{\delta}\mathbf{y}$ represents the displacement from a nominal position $\mathbf{y}_0$, where $\mathbf{K}$ is the stiffness matrix. Moreover, by defining

$$\boldsymbol{\delta}\mathbf{y} = \mathbf{J}\boldsymbol{\delta}\mathbf{q}\,, \tag{3.3}$$

it is possible to relate $\boldsymbol{\tau}$ to $\boldsymbol{\delta}\mathbf{q}$ as

$$\boldsymbol{\tau} = \mathbf{J}^T\mathbf{F} = \mathbf{J}^T\mathbf{K}\mathbf{J}\boldsymbol{\delta}\mathbf{q} = \mathbf{K}_Q\boldsymbol{\delta}\mathbf{q}\,, \tag{3.4}$$

where $\mathbf{K}_Q$ is the joint stiffness matrix and clearly depends on $\mathbf{K}$ and $\mathbf{J}$.

3.1.3 Selection Matrix: Model-free

Paul and Shimano [21], Whitney [22] and Salisbury [24] built a solid foundation for force control but still were proposing control schemes very different from hybrid control. Building on compliance control, hybrid motion/force control has its roots in the work of Mason [25] and [16] and Craig and Raibert [26] and [27].

Mason [25] and [16] introduces compliance and force control by identifying natural and artificial constraints, respectively geometric and trajectory-based constraints, by the means of selection matrices to keep the end effector within the constraint force and free motion subspaces. In this perspective, it is possible to define

$$\mathbf{S}\dot{\mathbf{y}} = 0 \tag{3.5}$$

and

$$(\mathbf{I} - \mathbf{S})\mathbf{F} = 0 \tag{3.6}$$

respectively, using $\mathbf{S}$ to declare directions where motion is not allowed by the physical constraints, and $(\mathbf{I} - \mathbf{S})$ the corresponding directions of free motion. $\mathbf{S}$ and $(\mathbf{I} - \mathbf{S})$ are formed of 1s and 0s. Mason thus stated that equations in such form limit velocities and forces to subspaces $\mathbf{S}_v$ and $\mathbf{S}_f$ which are orthogonal complements. There is a strong analogy with [21], where a selection of directions was instead proposed in the joint space.

Craig and Raibert [26] and [27] formally present hybrid control for position and force to achieve compliant motion for a manipulator. Specifically they map the error in the task related coordinate system to the joint space by means of the inverse Jacobian. This differs from the "free-joint" method in [21], because each joint is used for both position and force control, thus this approach involves all joints to devise position and force commands simultaneously. In other words, the selection of position and force control directions is done directly in the Cartesian space, leading to a more intuitive definition of the constraints. More specifically, they propose a controller whose two components are respectively responsible for motion control, $(\mathbf{I} - \mathbf{S})\boldsymbol{\Delta}\mathbf{y}$, and force control, $\mathbf{S}\boldsymbol{\Delta}\mathbf{F}$, where $\boldsymbol{\Delta}\mathbf{y}$ and $\boldsymbol{\Delta}\mathbf{F}$ are the errors on position and force measured on the end effector, giving

$$\boldsymbol{\tau} = \boldsymbol{\Gamma}(\mathbf{S}\boldsymbol{\Delta}\mathbf{F}) + \boldsymbol{\Psi}((\mathbf{I} - \mathbf{S})\boldsymbol{\Delta}\mathbf{y}), \tag{3.7}$$

where $\boldsymbol{\Gamma}$ and $\boldsymbol{\Psi}$ are force and position compensation functions used to map errors in the Cartesian space to commands in the joint space. Analogously to [25] and [16], $\mathbf{S} = diag(s_1 ... s_n)$ is defined as a compliance selection matrix, diagonal, with 1s on directions of constrained motion, thus enabling force control, and 0s in free motion directions. To map Cartesian errors, $\boldsymbol{\Delta}\mathbf{y}$, into the joint space, they suggest to use the inverse of the Jacobian, *i.e.*, $\mathbf{J}^{-1}$.

The contributions of [26] and [27] have had a great impact on the robotics community and inspired later efforts to analyse the stability of the scheme and to improve the initial formulation of hybrid control. In particular, the choice of using the inverse of the Jacobian, $\mathbf{J}^{-1}$, is especially critical since it does not scale to redundant manipulators and because of numerical issues arising from the inversion. Hence later work proposed alternative methods to avoid these problems.

Zhang and Paul [28] criticise: the approach in [24], claiming the method does not work near kinematic singularities; the approach in [26] and [27] because of the computational problems connected to the inverse of the Jacobian; and the approach in [21] since the proposed method achieves good performances only if the joints are aligned with each direction where compliance is required. Thus, they propose a modified hybrid control to overcome these difficulties in the form

$$\boldsymbol{\tau} = \mathbf{K}(\mathbf{I} - \mathbf{J}^{-1}\mathbf{S}\mathbf{J})\boldsymbol{\delta}\mathbf{q}\,. \tag{3.8}$$

This is different to the stiffness approach in [24], in that the latter uses $\boldsymbol{\tau} = \mathbf{J}^T\mathbf{K}\mathbf{J}\boldsymbol{\delta}\mathbf{q}$. It is also different from the formulation of hybrid control in [26] and [27], since it does not use only the $\mathbf{J}^{-1}$ to map the Cartesian error into the joint space, thus, according to the claim of the authors, enhancing performance since it reduces the computational load locally at the joints. Zhang and Paul call the selection matrix the Cartesian compliant matrix, but in our understanding it is the same as the selection matrix $\mathbf{S}$. Analogously to the $\mathbf{K}_Q$ in [24], they define the product $\mathbf{J}^{-1}\mathbf{S}\mathbf{J}$ as the joint compliance matrix $\mathbf{C}_\Omega$. The above analysis describes the position part of the control scheme. As for the active forces, they are transformed into joint torques through the Jacobian transpose $\mathbf{J}^T$ and added to the commanded torques in an open loop fashion.

The work of An and Hollerbach [29] proves kinematic instabilities of hybrid control by counterexamples, in particular showing that stability is dependent on the geometry of the manipulator. They report that hybrid control results in instability when used on a 2-degree-of-freedom manipulator with revolute joints, but proves stable on a polar manipulator. On the contrary, they show that stiffness control as proposed in [24] always results in a stable system, suggesting the use of the Jacobian transpose as the cause for this result.

On the same line of reasoning, Zhang [30] investigates kinematic stability of the linearised systems associated with hybrid control, [26] and [27], and active stiffness control, [24]. Zhang proves that hybrid control can become unstable when the joints are in certain configurations, while the stiffness control is always stable except in singular configurations, although constant gains do not guarantee the system be damped. Thus his conclusion reaffirms what [29] already proved by examples and counterexamples.

Fisher and Mujtaba [31] correct the formulations of [26] and [27], advocating that the original position solution, using the inverse of the Jacobian, $\mathbf{J}^{-1}$, not only gives rise to numerical problems but also is not a general and correct solution. Fisher and Mujtaba also criticise the approach in [28], since they propose an equivalent to $\mathbf{S}$ in the joint space, namely $\mathbf{J}^{-1}\mathbf{S}\mathbf{J}$. This is different from $\mathbf{S}$, which refers to the Cartesian space, while $\mathbf{J}^{-1}\mathbf{S}\mathbf{J}$ is a mapping in the joint space. They propose a revised formulation of the hybrid control scheme based on vector space analysis and advance a controller where the correct general position solution is in the form

$$\mathbf{q}_{es} = (\mathbf{S}^{\perp}\mathbf{J})^{\dagger}\mathbf{y}_e + (\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\mathbf{z}_q \tag{3.9}$$

and the general force control in the form

$$\boldsymbol{\tau}_{es} = (\mathbf{S}\mathbf{J})^{T}\mathbf{f}_e + (\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\mathbf{z}_{\tau}\,, \tag{3.10}$$

where $\mathbf{z}_q$ and $\mathbf{z}_{\tau}$ are arbitrary position and force vectors respectively, defined in the manipulator joint space. In this formulation, $\mathbf{q}_{es}$ represents the selected joint errors, while $\mathbf{y}_e$ is simply the position error in the Cartesian space. This means that by pre-multiplying $\mathbf{y}_e$ by $(\mathbf{S}^{\perp}\mathbf{J})^{\dagger}$, only the components of the error referring to the Cartesian directions where it is possible to control motion are taken into account and mapped into the joint space. The same reasoning applies to $\boldsymbol{\tau}_{es}$ and $\mathbf{f}_e$, where these quantities are used for force control in the directions indicated by $\mathbf{S}$. They consider that force control is orthogonal to position control, *i.e.*, $\mathbf{S}^{\perp} = \mathbf{I} - \mathbf{S}$. Furthermore, they show that the inverse of the Jacobian is an incorrect solution and they investigate how the null space of the Jacobian can influence the control of both positions and torques. Finally they derive two sufficient conditions for kinematic stability in addition to providing further insights on projection matrices. They provide the reader with a thorough section on linear algebra concepts related to projection matrices. In particular, they show that the projection matrix $\mathbf{S}$ is a projection matrix that selects a reduced Cartesian space, or equivalently a subspace of the entire Cartesian space, of interest. They also show that the mapping of this space into the joint space through the robot Jacobian may be problematic, hence their proposal of a more robust formulation of the hybrid control and their analysis of kinematic stability.

3.1.4 Selection Matrix: Model-based

Another essential contribution to the understanding of dynamics is the work of Khatib [5] and [6]. The operational space formulation of dynamics (see also Sec. 2) is defined and a controller directly in the operational space is proposed in the form

$$\boldsymbol{\tau} = \mathbf{J}^{T}(\mathbf{F}_m + \mathbf{F}_a + \mathbf{F}_{ccg})\,, \tag{3.11}$$

where $\mathbf{F}_m$ and $\mathbf{F}_a$ are respectively the components for motion and force control, while $\mathbf{F}_{ccg}$ is a compensation of centrifugal, Coriolis and gravity effects. From the hybrid control point of view, the definitions of $\mathbf{F}_m$ and $\mathbf{F}_a$ are of noteworthy importance. Khatib advances the following definitions

$$\mathbf{F}_m = \boldsymbol{\Lambda}(\mathbf{I} - \mathbf{S})\mathbf{F}_m^* \tag{3.12}$$

and

$$\mathbf{F}_a = \mathbf{S}\mathbf{F}_a^* + \boldsymbol{\Lambda}\mathbf{S}\mathbf{F}_s^* \,, \tag{3.13}$$

where $\mathbf{F}_m^*$ is the motion control part formed of a PD on position and velocity plus a feedforward on acceleration while $\mathbf{F}_a^*$ and $\mathbf{F}_s^*$ are respectively vectors of active force control and end effector velocity damping acting in the force control subspace. This work clearly follows the lines of Mason, and Raibert and Craig in the sense that it uses selection matrices to select certain directions in the Cartesian space, in order to separate motion and force control geometrically.

Additionally, McClamroch and Wang [32] present a generalisation of hybrid control, and studies conditions for closed-loop stabilisation including the constraints. A nonlinear controller, based on a modification of the computed torque method, is proposed and the authors claim that it is a generalisation of the hybrid control scheme presented in [26] and [27].

With respect to the analysis of stability for hybrid control, the main idea in Yabuta [33] is to study such stability by analysing the nonlinear dynamic system resulting from hybrid control using Lyapunov's theory and LaSalle's theorem. This analysis is a step forward in the sense that, contrary to previous work based on linearised models, it analyses the nonlinear dynamic system and factors such as additional phase lag and sampling delay. This work also offers stability conditions for hybrid control, even in the presence of friction, phase lag and discrete control. As a result, it provides new insights into how the inverse Jacobian in [27] adversely affects stability.

The work in Yoshikawa [34] is based on [27] but states that dynamics was not considered rigorously. Consequently Yoshikawa introduces dynamic hybrid control to account for it. He shows how a feedforward term $\mathbf{f}_{Fd}$ in the force control has a major desirable impact on the overall control, such that the final controller proposed is in the form

$$\boldsymbol{\tau} = \boldsymbol{\tau}_P + \boldsymbol{\tau}_F \,, \tag{3.14}$$

where:

$$\boldsymbol{\tau}_P = \mathbf{M}\ddot{\mathbf{q}}_d + \mathbf{h} \,, \tag{3.15}$$

$$\boldsymbol{\tau}_F = (\mathbf{S}\mathbf{J})^T \mathbf{f}_{Fd} \,, \tag{3.16}$$

where $\mathbf{S}$ is defined slightly differently as $\mathbf{S}\dot{\mathbf{y}} = 0$, originating from the derivation of the natural constraints expressed in the form $\mathbf{p}_i(\mathbf{y}) = 0, \;\;$ with $i = 1, 2, \ldots m$. Yoshikawa shows that desired position and force can both be realised if the robot is not in a kinematic singularity. Notice also that there is a substantial equivalence between $\mathbf{J}_C$ and $\mathbf{SJ}$. This is important because it relates controllers based on the selection matrix to controllers based on projectors, such as $\mathbf{P}$, in papers discussed later in this survey, such as West and Asada [35], Aghili [8] and Mistry and Righetti [7].

Yoshikawa et al. [36] reprise most of the concepts in the previous work [34], but adds the further contribution of a linearisation of the system by state feedback using a controller of the type

$$\boldsymbol{\tau} = \boldsymbol{\tau}_P + \boldsymbol{\tau}_F \,, \tag{3.17}$$

by setting

$$\boldsymbol{\tau}_P = \mathbf{M}\ddot{\mathbf{q}}_d(\mathbf{u}_1) + \mathbf{h} \tag{3.18}$$

and

$$\boldsymbol{\tau}_F = \mathbf{J}^T\mathbf{S}^T\mathbf{u}_2 \,. \tag{3.19}$$

The authors show that it is possible to choose an appropriate $\ddot{\mathbf{q}}$ such that when not in singularity, they achieve a linear decoupled system: $\ddot{\mathbf{y}}$ on the motion control directions and $\mathbf{F}$ in the force control directions, depending on the new control inputs $\mathbf{u}_1$ and $\mathbf{u}_2$

$$\ddot{\mathbf{y}} = \mathbf{u}_1 \tag{3.20}$$

and

$$\mathbf{F} = \mathbf{u}_2 \,, \tag{3.21}$$

where $\mathbf{u}_1$ and $\mathbf{u}_2$ are new inputs used to specify the desired motion trajectory, $\mathbf{u}_1$, and the desired force, $\mathbf{u}_2$. They also propose to use a PD feedback for the position control and I (integral) feedback on the force control to ensure zero error as time goes to infinity.

In later work, Yoshikawa and Sudou [37] propose a method for online estimation of unknown constraint surfaces, advocating that they can be estimated geometrically using the $\mathbf{y}$ displacements, $\boldsymbol{\delta}\mathbf{y}$, and the changes in contact force, $\boldsymbol{\delta}\mathbf{F}$, which need a force sensor on the end effector to be measured.

Chiaverini and Sciavicco [38] propose a different control architecture, which is rule-based, to overcome the flaws of model-based selection matrix hybrid control. This system entails the cost of a more complex design but enables a parallel force/position control, where force control is prevalent over position control so that the system is able to be compliant with the environment and its constraints. Chiaverini et al. [39] also study the stability of force/position control of a manipulator in elastic contact with the environment, exploiting Lyapunov analysis.

Predominantly, in previous work, the subspace of the force control is assumed to be orthogonal and complementary to the one of free motion. In contrast, De Luca and Manes [40] introduce a third subspace where it is possible either to control force, implicitly producing an acceleration, or to control motion, implicitly producing an active force. This is possible in directions where forces are not counterbalanced by a constraint reaction, thus producing active work, *i.e.*, directions where active forces and motion coexist. They model the environment as a dynamic system and introduce a framework for hybrid motion/force control. Following the trend of selection matrices and projectors, they show that along the directions of free motion, accelerations can be imposed freely (using the columns of a matrix defined as $\mathbf{T}_K$). Another matrix, $\mathbf{Y}_R$, defines directions where forces can be exerted. The authors suggest that it is possible either to apply forces in the directions spanned by $\mathbf{Y}_A$, thus resulting in accelerations in $\mathbf{T}_D$, or to impose accelerations in $\mathbf{T}_D$, resulting in forces in $\mathbf{Y}_A$. In other words, the last two matrices represent bases of the same subspace, the one where it is possible either to control force, implicitly producing an acceleration, or to control motion, implicitly producing an active force. For a better understanding of this third subset of directions, we report an example from the original paper. Consider a robot pushing a mass on a rail. In this example, the authors state that it is possible to either specify an active force on the mass to slide, thus implicitly forcing the mass to move, or specify a motion for the mass, thus implicitly imposing an active force on the mass. This reciprocity in the definition of such tasks is clearly represented in their model and control framework.

3.1.5 Projection Approach

West and Asada [35] model contacts between the robot and the environment, as virtual links with passive joints, stating that constraints arising from contacts generate a close kinematic chain. Their contribution lies mainly in the fact that their approach works seamlessly with multiple contacts and with contacts not at the end effector, which are situations likely to happen in realistic interactions with the environment. In particular, their idea takes root in the fact that it is possible to express the kinematics and dynamics of the robot and the constraints both in the joint space and as a closed kinematic chain. Specifically, they relate variations in the joint angles to variations in the position of the contact point(s) through the mapping

$$\boldsymbol{\delta}\mathbf{y}_C = \mathbf{J}_C\boldsymbol{\delta}\mathbf{q} = \mathbf{0}\,, \tag{3.22}$$

where the Jacobian $\mathbf{J}_C$ describes the relationship between variations in the joint space $\boldsymbol{\delta}\mathbf{q}$ and the position of the contact point, which is fixed, thus the equality to $\mathbf{0}$. They advocate that permissible motion and permissible forces of such closed kinematic chains are identified by the two projectors

$$\mathbf{P} = \mathbf{I} - \mathbf{J}_C^{\dagger}\mathbf{J}_C \tag{3.23}$$

and

$$\mathbf{I} - \mathbf{P} = (\mathbf{J}_C^{\dagger}\mathbf{J}_C)^T\,, \tag{3.24}$$

such that they are orthogonal complements, *i.e.*

$$\mathbf{P}(\mathbf{I} - \mathbf{P}) = \mathbf{0}\,. \tag{3.25}$$

They also define essential position and force variables, *i.e.*, those directions of motion and force in the Cartesian space and at the contact which must be controlled to perform the task, while leaving the other variables as arbitrary. As for the joint space, they also propose projectors to comply with those variable definitions. These projectors are used to map or filter the error signals in order to compute the torque commands to send to the robot, *i.e.*, for the force component

$$\boldsymbol{\tau} = (\mathbf{I} - \mathbf{P})\mathbf{J}^T\mathbf{F}\,. \tag{3.26}$$

Featherstone et al. [41] also extend the formulation of hybrid control proposed in [26] and [27], in that they introduce a more general contact model and a filtering of the error signals into force and motion components. They introduce two spaces $\mathcal{N}$ and $\mathcal{T}$ modelled by a pair of matrices $\mathbf{N}$ and $\mathbf{T}$ respectively, formed by independent force and velocity vectors, such that $\mathbf{N}^T\mathbf{T} = \mathbf{0}$. In other words, $\mathcal{N}$ is the space of normal vectors referring to the actual contact, while $\mathcal{T}$ refers to the space of those vectors which are tangential to the contact. For the examples provided in the paper, this representation overcomes the drawback of the classical selection matrix $\mathbf{S}$ since it does not rely solely on the six parameters used in $\mathbf{S}$ to select Cartesian directions, instead it scales well to more intricate situations, *e.g.*, multiple-point contacts with non-intersecting normals (see Fig. 1 of the same paper). Moreover, they define two more subspaces $\mathcal{N}'$ and $\mathcal{T}'$, complementary to $\mathcal{N}$ and $\mathcal{T}$, such that their direct sum is equal to the space of the force vectors and the motion vectors respectively. This is a representation analogous to the one in [40] but without taking into consideration the possible third subspace. They define the projectors $\boldsymbol{\Omega}_f$ and $\boldsymbol{\Omega}_m$ as

$$\boldsymbol{\Omega}_f(\mathbf{A}) = \mathbf{N}(\mathbf{N}^T\mathbf{A}^{-1}\mathbf{N})^{-1}\mathbf{N}^T\mathbf{A}^{-1}\,, \tag{3.27}$$

$$\bar{\boldsymbol{\Omega}}_f(\mathbf{A}) = 1 - \boldsymbol{\Omega}_f(\mathbf{A}) = \mathbf{A}\mathbf{T}(\mathbf{T}^T\mathbf{A}\mathbf{T})^{-1}\mathbf{T}^T \tag{3.28}$$

and

$$\mathbf{\Omega}_m(\mathbf{B}) = \mathbf{T}(\mathbf{T}^T\mathbf{B}^{-1}\mathbf{T})^{-1}\mathbf{T}^T\mathbf{B}^{-1}\,, \tag{3.29}$$

$$\bar{\mathbf{\Omega}}_m(\mathbf{B}) = 1 - \mathbf{\Omega}_m(\mathbf{B}) = \mathbf{B}\mathbf{N}(\mathbf{N}^T\mathbf{B}\mathbf{N})^{-1}\mathbf{N}^T\,, \tag{3.30}$$

where $\mathbf{A}$ and $\mathbf{B}$ are matrices mapping from motion space to force space and oppositely from force to motion space. They also claim that, with this proposed filtering ($\mathbf{\Omega}_m(\mathbf{B})$ and $\mathbf{\Omega}_f(\mathbf{A})$ respectively for motion and force components in the control) of the command inputs, the control is decoupled, stating that acceleration depends only on the filtered command of the motion control and force on the filtered command of the force control. Furthermore, it must be stressed that within this framework it is possible to include multiple contacts, in contrast to the formulation in [26] and [27].

We have previously stressed that in presence of constraints, the robot motion free space is a subspace of the entire workspace. Although not always the case, [40], it can often be assumed that the orthogonal subspace to the motion free subspace is the subspace where the robot can exert active force. Aghili [8] develops a method known as projected dynamics. Consider the constraint in Eqn. 2.5. It is possible to define a projector $\mathbf{P}$ into the null space of $\mathbf{J}_C$ in the same way already proposed in [35]

$$\mathbf{P} = \mathbf{I} - \mathbf{J}_C^{\dagger}\mathbf{J}_C\,. \tag{3.31}$$

The author goes further than [35] in the sense that he proposes projecting the entire dynamics of the robot with this projector, formally defining the projected dynamics. The range of the projector $\mathbf{P}$ is the null space of $\mathbf{J}_C$ and the range of the projector $(\mathbf{I} - \mathbf{P})$ is the orthogonal space to the null space of $\mathbf{J}_C$

$$\mathcal{R}(\mathbf{P}) = \mathcal{N}(\mathbf{J}_C) \tag{3.32}$$

and

$$\mathcal{R}(\mathbf{I} - \mathbf{P}) = \mathcal{N}(\mathbf{J}_C)^{\perp}\,. \tag{3.33}$$

Moreover, every vector $\mathbf{v}$ can be decomposed into

$$\mathbf{v} = \mathbf{v}_{\parallel} + \mathbf{v}_{\perp}\,, \tag{3.34}$$

where $\mathbf{v}_{\parallel} = \mathbf{P}\mathbf{v}$ and $\mathbf{v}_{\parallel} = (\mathbf{I} - \mathbf{P})\mathbf{v}$ such that $\mathbf{v}_{\parallel} \in \mathcal{N}(\mathbf{J}_C)$ and $\mathbf{v}_{\perp} \in \mathcal{N}(\mathbf{J}_C)^{\perp}$. Now, projecting the equation of the robot joint space dynamics (Eqn. 2.4) with $\mathbf{P}$ gives

$$\mathbf{P}\mathbf{M}\ddot{\mathbf{q}} = \mathbf{P}(\boldsymbol{\tau} - \mathbf{h})\,, \tag{3.35}$$

since $\mathbf{J}_C^T\boldsymbol{\lambda} \in \mathcal{N}(\mathbf{J}_C)^{\perp}$ so $\mathbf{P}\mathbf{J}_C^T\boldsymbol{\lambda} = \mathbf{0}$. Finally

$$\ddot{\mathbf{q}} = \mathbf{M}_C^{-1}(\mathbf{P}(\boldsymbol{\tau} - \mathbf{h}) + \mathbf{C}_C\dot{\mathbf{q}})\,, \tag{3.36}$$

or equivalently in the decomposed form

$$\ddot{\mathbf{q}} = \mathbf{M}_C^{-1}(\boldsymbol{\tau}_{\|} - \mathbf{h}_{\|} + \mathbf{C}_C\dot{\mathbf{q}})\,, \tag{3.37}$$

where $\mathbf{M}_C = \mathbf{M} + \mathbf{PM} - (\mathbf{PM})^T$ and $\mathbf{C}_C = \mathbf{MC}$. $\mathbf{M}_C$ is invertible thus $\ddot{\mathbf{q}}$ could be expressed explicitly, in contrast to the previous case where $\mathbf{PM}$ was not invertible and $\ddot{\mathbf{q}}$ could not be explicitly derived. The expression of $\mathbf{J}_C^T\boldsymbol{\lambda}$ can be obtained analogously to $\ddot{\mathbf{q}}$, by projecting Eq. 2.4 with $(\mathbf{I} - \mathbf{P})$

$$\mathbf{J}_C^T\boldsymbol{\lambda} = (\mathbf{I} - \mathbf{P})(\boldsymbol{\tau} - \mathbf{h}) - (\mathbf{I} - \mathbf{P})\mathbf{M}\mathbf{M}_C^{-1}(\mathbf{P}(\boldsymbol{\tau} - \mathbf{h}) + \mathbf{C}_C\dot{\mathbf{q}})\,, \tag{3.38}$$

or equivalently

$$\mathbf{J}_C^T\boldsymbol{\lambda} = (\boldsymbol{\tau}_{\perp} - \mathbf{h}_{\perp}) - \boldsymbol{\mu}(\boldsymbol{\tau}_{\|} - \mathbf{h}_{\|}) - \boldsymbol{\mu}\mathbf{C}_C\dot{\mathbf{q}}\,, \tag{3.39}$$

where

$$\boldsymbol{\mu} = (\mathbf{I} - \mathbf{P})\boldsymbol{\alpha} \tag{3.40}$$

and

$$\boldsymbol{\alpha} = \mathbf{M}\mathbf{M}_C^{-1}\,. \tag{3.41}$$

Aghili claims that within this framework a complete decoupling is then achieved only when

$$\forall \mathbf{v} \in \mathcal{N}(\mathbf{J}_C) \;:\; \mathbf{M}\mathbf{v} \in \mathcal{N}(\mathbf{J}_C)\,, \tag{3.42}$$

such that the expression of the contact forces would reduce to

$$\mathbf{J}_C^T\boldsymbol{\lambda} = (\boldsymbol{\tau}_{\perp} - \mathbf{h}_{\perp}) - \boldsymbol{\mu}\mathbf{C}_C\dot{\mathbf{q}}\,. \tag{3.43}$$

Moreover, if $\boldsymbol{\tau}_{\perp}$ is defined as

$$\boldsymbol{\tau}_{\perp} = \mathbf{h}_{\perp} + \boldsymbol{\mu}(\boldsymbol{\tau}_{\|} - \mathbf{h}_{\|} + \mathbf{C}_C\dot{\mathbf{q}}) + \mathbf{u}_F\,, \tag{3.44}$$

then the decoupling can be restored because

$$\mathbf{J}_C^T\boldsymbol{\lambda} = \mathbf{u}_F\,. \tag{3.45}$$

This means that a compensation for an imperfect decoupling by controlling the force is proposed. In other words, the author proposes to control the force by compensating for the coupling and at the same time imposing the desired behaviour $\mathbf{u}_F$. Aghili and Su

[42] extend the formulation of projected dynamics proposing to use optimisation to take into account unilateral constraints in the definition of force control.

Mistry and Righetti [7] fuse the formulations of the operational space dynamics and projective dynamics to express operational space dynamics as

$$\mathbf{\Lambda}_C\ddot{\mathbf{y}} + \mathbf{\Lambda}_C(\mathbf{J}\mathbf{M}_C^{-1}\mathbf{P}\mathbf{h} - (\dot{\mathbf{J}} + \mathbf{J}\mathbf{M}_C^{-1}\mathbf{C})\dot{\mathbf{q}}) = \mathbf{F}\,, \tag{3.46}$$

where $\mathbf{M}_C = \mathbf{P}\mathbf{M} + \mathbf{I} - \mathbf{P}$, $\mathbf{\Lambda}_C = (\mathbf{J}\mathbf{M}_C^{-1}\mathbf{P}\mathbf{J}^T)^{-1}$, and $\mathbf{C} = -\mathbf{J}_C^{\dagger}\dot{\mathbf{J}}_C$. This approach, based on projective dynamics, has been used in [7] and further in [1] in order to devise a controller based on operational space, whose advantage is that it is formed of three separate components respectively in charge of motion control, active force control and possibly also a compensation in the task null space. This additional term is responsible for maximising or minimising some criterion. For example, in [1] the null space compensation is used as to avoid a drift of the elbow while the manipulator wipes a board, and has been defined as a PD controller whose desired joint values are the initial configuration of the robot, thus effectively minimising the drift while performing the task. Such a controller can be formulated as

$$\boldsymbol{\tau} = \mathbf{P}\mathbf{J}^T\tilde{\mathbf{F}} + (\mathbf{I} - \mathbf{P})\boldsymbol{\tau}_a + \mathbf{P}\mathbf{N}\boldsymbol{\tau}_{null}\,, \tag{3.47}$$

where $\mathbf{P}\mathbf{J}^T\tilde{\mathbf{F}}$ is the motion control part; $(\mathbf{I}-\mathbf{P})\boldsymbol{\tau}_a$ is a reference in active force, containing a feedforward term specifying the desired force interaction; and $\mathbf{P}\mathbf{N}\boldsymbol{\tau}_{null}$ is the motion term in the null space of the task, where $\mathbf{N}$ is any projection into the null space of the task Jacobian, dynamically consistent. An analysis of the case of underactuated systems is presented in [7]. This is a very interesting topic but not the main focus of this review.

3.1.6 Other Applications

Hybrid control has also been exploited for multiple robot manipulators. Hayati [43] extends hybrid position/force control to cooperating robots, assigning to each robot a part of the object mass. Yoshikawa and Zheng [44] build on dynamic hybrid control and devise a control scheme for single object motion, which handles both constraint and internal force for multiple manipulators while accounting for both object and robot dynamics. In particular, they propose to linearise the system through nonlinear state feedback and a servo controller is used to achieve higher robustness.

3.1.7 Decoupling

In summary, hybrid motion/force control is a powerful control framework for tasks involving not only motion but also active forces. Over the years it has been extended to generate better performance, to be more flexible with respect to the geometry of the contacts and to ensure stability of the overall system. There is one particular feature, known as decoupling, which merits a deeper analysis in its own right.

Decoupling is the capability of a system to provide both motion and force control which do not interact with each other. In other words, it is possible to change the exerted force without altering or disturbing the robot trajectory, and vice versa. This is a very fundamental and powerful feature, especially when devising distinct desired evolutions for motion and force, which otherwise could not be realised. It is a topic arising prominently when using a control scheme such as hybrid control, or in any task where both motion and active force need to be precisely controlled. An important feature necessary to achieve decoupling is either the knowledge of the dynamic model of the robot, in order to compensate for the dynamics of the robot, or a feedback on force at the contact point, so that this feedback can be used to modulate force control. Specifically, we now discuss several different ways of achieving decoupling, which have been proposed in the literature.

A selection matrix $\mathbf{S}$ is introduced in Mason [25] and [16], which provides the means to define and select directions in the Cartesian space frame of interest, where there is the possibility of free motion and conversely the possibility to exert forces on a contact surface. This concept is found also in Craig and Raibert [26] and [27] with few fundamental differences. The main difference is that the latter use the selection matrix to select only the error signals for each of the two subspaces of free motion and contact force. However the main idea proposed by these works is similar, namely to obtain decoupling between motion and force through an appropriate selection of directions in the Cartesian space, *i.e.*, by defining directly the commands which are feasible in the Cartesian space. In other words, these control schemes devise position and/or velocity commands in the directions of free motion, and force references in the directions where the robot is in contact and thus can exert active forces on the contact point or surface. Mathematically

$$\mathbf{S}\dot{\mathbf{y}} = 0 \tag{3.48}$$

and

$$(\mathbf{I} - \mathbf{S})\mathbf{F} = 0\,, \tag{3.49}$$

meaning that in the directions of Cartesian space indicated by $\mathbf{S}$, motion is not feasible, while, on the contrary, in the orthogonal subspace, described by $(\mathbf{I} - \mathbf{S})$, it is possible to specify references in position or velocity.

In contrast to the above ideas, Yoshikawa et al. [36] propose to use a feedback linearisation scheme such that, when not in singularity, the system is linearised as

$$\ddot{\mathbf{y}} = \mathbf{u}_1 \tag{3.50}$$

and

$$\mathbf{F} = \mathbf{u}_2 \,, \tag{3.51}$$

with the two new inputs $\mathbf{u}_1$ and $\mathbf{u}_2$ to specify respectively accelerations and forces. While defining the control

$$\boldsymbol{\tau} = \boldsymbol{\tau}_P + \boldsymbol{\tau}_F \,, \tag{3.52}$$

$$\boldsymbol{\tau}_P = \mathbf{M}\ddot{\mathbf{q}}_d + \mathbf{h} \tag{3.53}$$

and

$$\boldsymbol{\tau}_F = \mathbf{J}^T\mathbf{S}^T\mathbf{u}_2 \,, \tag{3.54}$$

where the selection matrix $\mathbf{S}$ appears in order to define the suitable directions for force control in the Cartesian space. The selection matrix, as proposed by [25], [16], [26] and [27], has the intrinsic limitation of not scaling easily to multiple contacts on the same robot or to situations where it is not easy or possible to know a priori or define orthogonal contact directions explicitly in the Cartesian space.

In [45], a model is derived for a robot in contact with the environment. A control architecture is proposed which leads to a complete decoupling of motion and force control. Reducing the dimensionality of the equation of motion, pseudo-velocities $\mathbf{v}$ are defined as

$$\mathbf{v} = \mathbf{B}\dot{\mathbf{q}} \,, \tag{3.55}$$

such that the matrix $\begin{bmatrix}\mathbf{J}_C^T & \mathbf{B}^T\end{bmatrix}^T$ is non singular. A control law in the form

$$\boldsymbol{\tau} = \mathbf{J}_C^T\mathbf{u}_1 + \mathbf{M}\mathbf{E}\mathbf{u}_2 + \mathbf{h} - \mathbf{M}(\mathbf{D}\dot{\mathbf{J}}_C + \mathbf{E}\dot{\mathbf{B}})\mathbf{E}\mathbf{v} \,, \tag{3.56}$$

where $\begin{bmatrix}\mathbf{J}_C \\ \mathbf{B}\end{bmatrix}^{-1} = \begin{bmatrix}\mathbf{D} & \mathbf{E}\end{bmatrix}$, is verified to lead to

$$\boldsymbol{\lambda} = \mathbf{u}_1 \tag{3.57}$$

and

$$\dot{\mathbf{v}} = \mathbf{u}_2 \,, \tag{3.58}$$

which is a result very similar to [36]. Notice that in this formulation, $\mathbf{J}_C\mathbf{E} = \mathbf{0}$ so that pre-multiplying (2.4) by $\mathbf{E}^T$ yields

$$\mathbf{E}^T\mathbf{M}\ddot{\mathbf{q}} = \mathbf{E}^T(\boldsymbol{\tau} - \mathbf{h})\,, \tag{3.59}$$

which is closely related to (3.35).

Featherstone et al. [41] propose to use projectors which are arguably a more flexible solution than the selection matrix (capable of additionally handling non-perpendicular contact directions), and to filter the commands in order to map those same commands into the subspaces of free motion and active force.

West and Asada [35] and Aghili [8] propose to use a projector based on the Jacobian of the constraints, $\mathbf{J}_C$, defined as

$$\mathbf{P} = \mathbf{I} - \mathbf{J}_C^\dagger\mathbf{J}_C\,. \tag{3.60}$$

Aghili in particular analyses the projected dynamics and concludes that a perfect decoupling is achieved when

$$\forall \mathbf{w} \in \mathcal{N}(\mathbf{J}_C) \;:\; \mathbf{M}\mathbf{w} \in \mathcal{N}(\mathbf{J}_C)\,. \tag{3.61}$$

This condition is not satisfied in general and he therefore devises an additional component in the force control part of the controller in order to compensate for the coupling effect. Refer to the Subsec. 3.1.5 for more details. The same $\mathbf{P}$ projector is also used in [7] and [1] when dealing with control using operational space projected dynamics.

3.1.8 Simulative Results

Hybrid control builds its foundations on the selection of directions where it is possible to control motion and directions where it is possible to control force. We described different ways to compute such control actions, but they all share the same goal, *i.e.*, such discrimination of the space where the robot acts. We pick here two techniques to achieve decoupling and we analyse how they behave in two tasks: a pure positional task and a motion/force task. We believe that the selection matrix $\mathbf{S}$ and the projector $\mathbf{P}$ are two representative choices to analyse decoupling quantitatively. As already explained in other sections, the other variations are originated mainly by these two techniques, thus it does not appear to be restrictive to show results only with these two techniques.

To highlight the differences between the approaches based respectively on the selection matrix $\mathbf{S}$ and the projector $\mathbf{P}$, we report the results of a simulated 3-degree-of-freedom planar manipulator with links of unitary mass and length, *i.e.*, $\mathrm{l} = 1\mathrm{m}$ and $\mathrm{m} = 1\mathrm{kg}$, constrained to move parallel to the x axis of the workspace plane as in Fig. 3.1. The manipulator is asked to slide its end effector along a rigid wall (shown as thick black line), from the starting point (shown as a red dot), to the goal point (shown as a green dot).

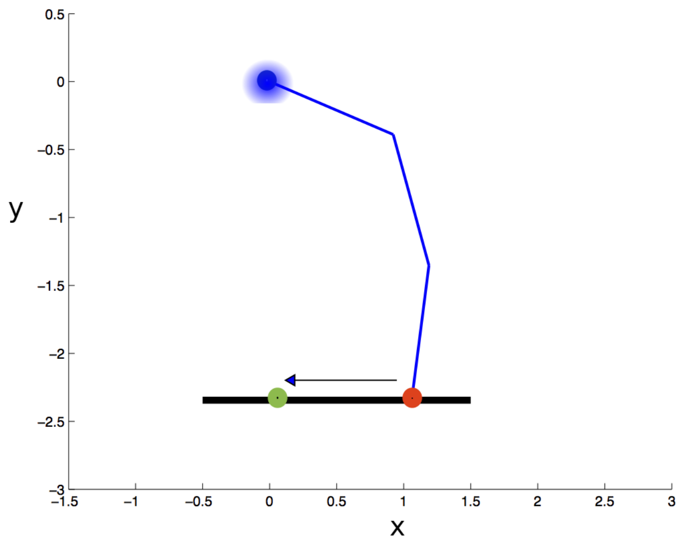


FIGURE 3.1: 3-degree-of-freedom planar manipulator asked to slide its end effector along a rigid wall (thick black line) from the starting point o to the goal point o. The fixed base of the robot is set in the origin of the plane o.

We implemented two controllers in Matlab[1]. The controller using the selection matrix **S** is the one originally proposed by Khatib [5] and [6], namely

$$\boldsymbol{\tau} = \mathbf{J}^T(\mathbf{F}_m + \mathbf{F}_a + \mathbf{F}_{ccg}) \,, \tag{3.62}$$

with

$$\mathbf{F}_m = \boldsymbol{\Lambda}(\mathbf{I} - \mathbf{S})(\ddot{\mathbf{y}}_d + \mathbf{K}_d(\dot{\mathbf{y}}_d - \dot{\mathbf{y}}) + \mathbf{K}_p(\mathbf{y}_d - \mathbf{y})) \,, \tag{3.63}$$

$$\mathbf{F}_a = \mathbf{S}\mathbf{F}_d \tag{3.64}$$

and

$$\mathbf{F}_{ccg} = (\mathbf{M}^{-1}\mathbf{J}^T\boldsymbol{\Lambda})^T\mathbf{h} - \boldsymbol{\Lambda}\dot{\mathbf{J}}\dot{\mathbf{q}} \,. \tag{3.65}$$

The controller based on projected operational space dynamics, *i.e.*, using **P**, is taken from Mistry and Righetti [7] and Ortenzi et al. [1]

$$\boldsymbol{\tau} = \mathbf{P}\boldsymbol{\tau}_m + (\mathbf{I} - \mathbf{P})\boldsymbol{\tau}_a \,, \tag{3.66}$$

[1]MATLAB, version 8.3.0.532 (R2014a): The MathWorks Inc., 2014

where

$$\begin{aligned}\boldsymbol{\tau}_m = \mathbf{J}^T\boldsymbol{\Lambda}_C(\ddot{\mathbf{y}}_d + \mathbf{K}_d(\dot{\mathbf{y}}_d - \dot{\mathbf{y}}) + \mathbf{K}_p(\mathbf{y}_d - \mathbf{y}) \\ + \mathbf{J}\mathbf{M}_C^{-1}(\mathbf{Ph} - \mathbf{C}_C\dot{\mathbf{q}}) - \dot{\mathbf{J}}\dot{\mathbf{q}})\end{aligned} \tag{3.67}$$

and

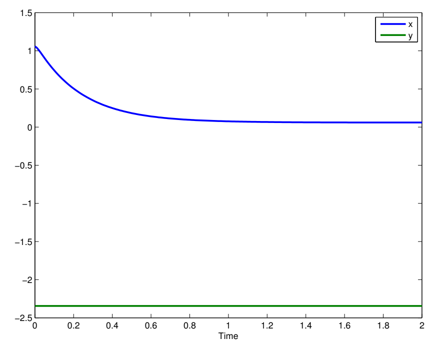
$$\boldsymbol{\tau}_a = (\mathbf{I} - \mathbf{P})\mathbf{h} + (\mathbf{I} - \mathbf{P})\mathbf{M}\mathbf{M}_C^{-1}(\mathbf{P}\boldsymbol{\tau}_m - \mathbf{Ph} + \mathbf{C}_C\dot{\mathbf{q}}) + \mathbf{J}_C^T\boldsymbol{\lambda}_d\,. \tag{3.68}$$

We set 200 simulation steps with $\delta t = 0.01$s for a total simulated time of 2s. First we compare these two controllers on a purely positional task, *i.e.*, respectively only $\mathbf{F}_m$ and $\boldsymbol{\tau}_m$ components are used. Fig. 3.2 reports the evolution of the robot on the plane, and its end effector trajectory in the x-y plane. We also report the torques commanded by each controller and the forces exerted by the robot at the contact point in Fig. 3.3. These figures clearly show that both controllers succeed in the positional task, and the robot shows indistinguishable evolution. However, the controller using the projector $\mathbf{P}$ significantly reduces torques as compared to the controller using the selection matrix $\mathbf{S}$, at the cost of having non-zero active forces at the contact point. This was expected from the analysis already presented.

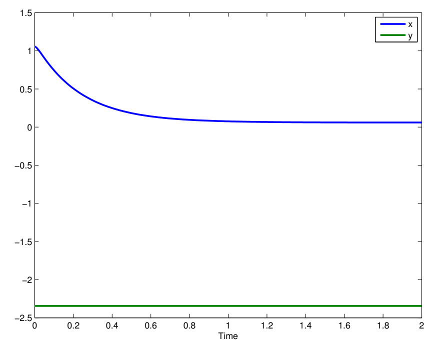
Secondly, we compare these two controllers when given a task which involves also a desired active force. The motion task remains as in the previous experiment, but the robot is also asked to exert a force $\lambda_{des} = 5$N at the contact point. We report the evolutions in Fig. 3.4. Robot motion is unchanged, but this is easily explained since the components of force control in both controllers do not affect motion. In contrast to the previous experiment and as expected, the exerted force accurately matches the desired active force. We report also the commanded torques generated by both controllers, Fig. 3.5. Interestingly but not surprisingly, the torques generated by the controllers are substantially the same. This can be explained by the fact that by specifying a desired active force, the controller based on projected dynamics (on $\mathbf{P}$) has to compensate for the robot dynamics, thus being denaturalised in the sense that it cannot exploit the contact anymore.

These results suggest that:

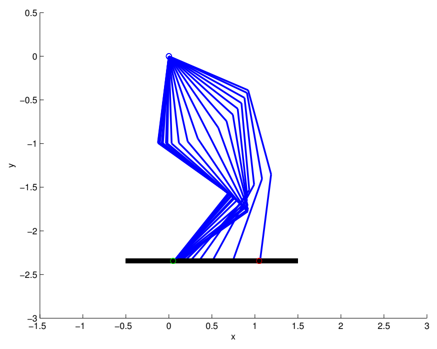
1. $\mathbf{S}$ decouples motion from active force but it needs an entire compensation of the robot dynamics even just for a pure motion task. On the contrary, for a motion task, $\mathbf{P}$ allows the controller to command lower torques with respect to an approach based on $\mathbf{S}$, but the motion part of the controller affects also the exerted force at the contact point. This feature can be explained by considering that the approach based on $\mathbf{P}$ exploits the constraint, in the sense that it exploits the reaction force at the contact point in order to compensate for the robot dynamics. To do this



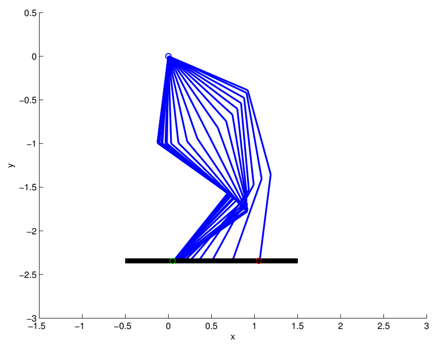
(a) End effector position - Operational Space Dynamics with Selection Matrix (**S**)



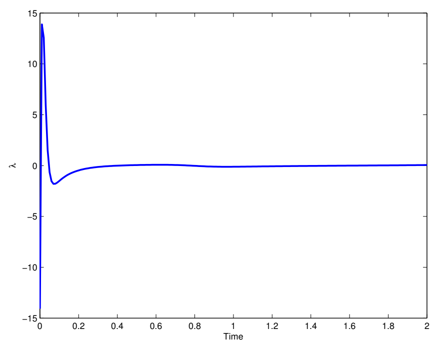
(b) End effector position - Projected Operational Space Dynamics (**P**)



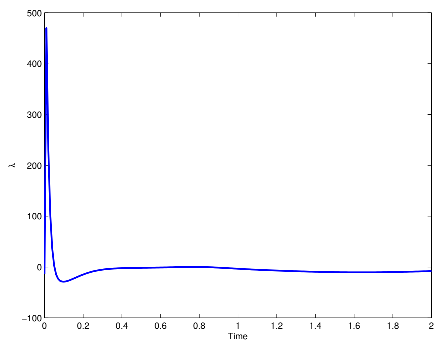
(c) Robot evolution - Operational Space Dynamics with Selection Matrix (**S**)



(d) Robot evolution - Projected Operational Space Dynamics (**P**)



(e) λ - Operational Space Dynamics with Selection Matrix (**S**)



(f) λ - Projected Operational Space Dynamics (**P**)

FIGURE 3.2: First line: end effector position on the workspace plane. Second line: robot evolution. Third line: reconstructed exerted interactive force between the end effector and the contact surface.

it is indeed necessary to interact with the environment, and so the non-null active forces are explained. On the other hand, the approach based on **S** devises higher commanded torques because it does not rely on the contact to compensate for the robot dynamics. In other words, this approach compensates for the whole robot dynamics while controlling motion, and this explains higher torque commands;

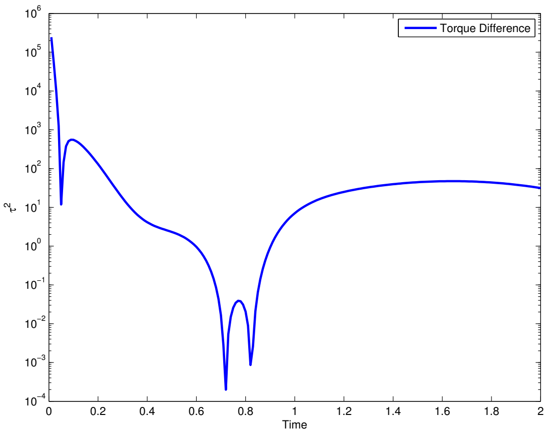


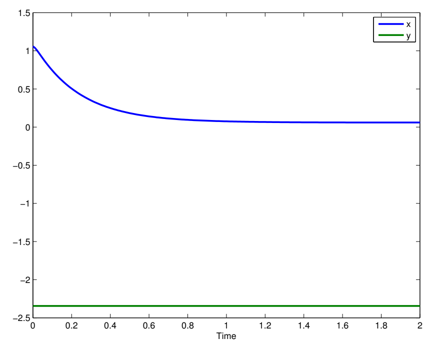
FIGURE 3.3: Torque comparison. This figure reports the difference (*i.e.*, $T_S - T_P$ with an abuse of notation) of the (sum of squared) torques commanded to the joint by the controller using the selection matrix and the one using the projector. The ones computed by projected operational space dynamics controller, *i.e.*, based on $\mathbf{P}$, are constistently lower than the ones by operational space dynamics controller using $\mathbf{S}$. The difference is reported in a semilogarithmic plot.

it can be concluded that the method using the projector $\mathbf{P}$ sacrifices decoupling for torque efficiency (lowering the commands), while the method using $\mathbf{S}$ provides superior decoupling at the cost of higher torques;

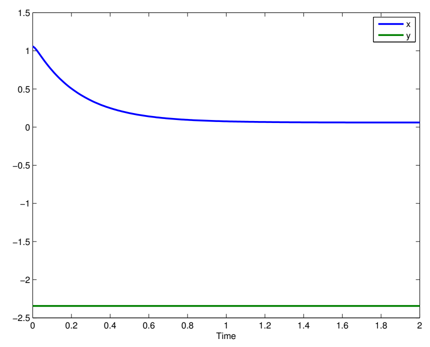
2. a force component in the $\mathbf{P}$ approach denaturalises the whole approach in the sense that the advantage of exploiting the contact is overridden by the imposition of a desired active force and an entire compensation of the robot dynamics may be necessary in addition to a compensation of the cross-coupling effect brought by the motion control part. On the other hand, this is the natural way to achieve a decoupling for this approach, in the sense that it is thus possible to impose a desired motion and a desired active force. As a matter of fact, adding a force component into the $\mathbf{P}$ approach makes this become equivalent to the selection matrix $\mathbf{S}$ approach.

Clearly the correct choice of approach is dependent on the application. If the task is in motion and there is no need to care about the force interaction or we can sacrifice decoupling to reduce the torque effort (*e.g.*, the robot leans on a table and we are sure the table is not going to break under the forces exerted on it by the robot) then the approach with $\mathbf{P}$ is convenient because it commands lower torques. When force is also an issue, then both approaches are successful.

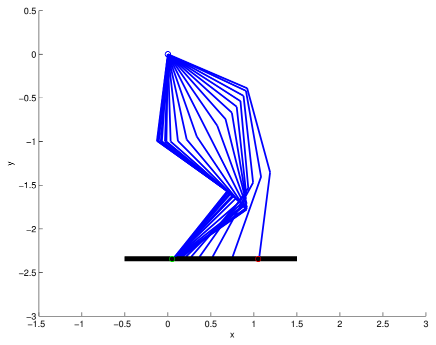
A final consideration is the fact that both methods require a precise knowledge of the model of the robot, otherwise a decoupling will become compromised due to fallacious compensations.



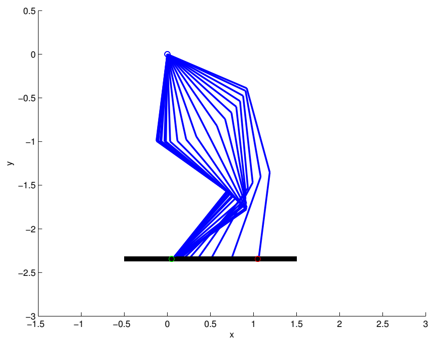
(a) End effector position - Operational Space Dynamics with Selection Matrix (**S**)



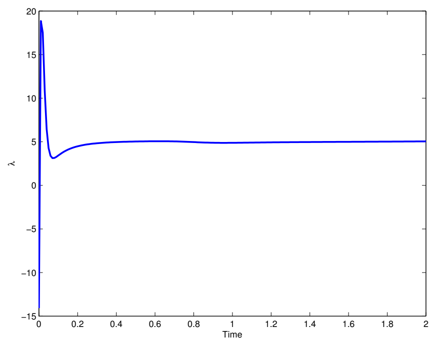
(b) End effector position - Projected Operational Space Dynamics (**P**)



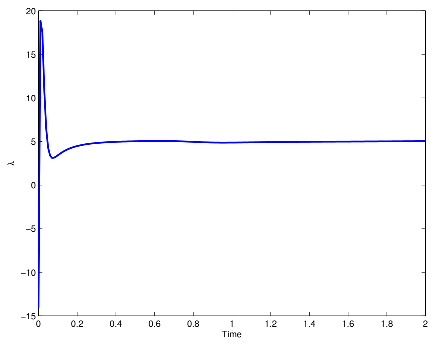
(c) Robot evolution - Operational Space Dynamics with Selection Matrix (**S**)



(d) Robot evolution - Projected Operational Space Dynamics (**P**)



(e) λ - Operational Space Dynamics with Selection Matrix (**S**)



(f) λ - Projected Operational Space Dynamics (**P**)

Figure 3.4: First line: end effector position on the workspace plane. Second line: robot evolution. Third line: reconstructed exerted interactive force between the end effector and the contact surface.

3.1.9 Discussion

There are three other key concepts that need to be discussed further to better understand the implications of using hybrid control: 1) the need for a precise model to compensate the dynamics when devising the control and a priori knowledge about the environment,

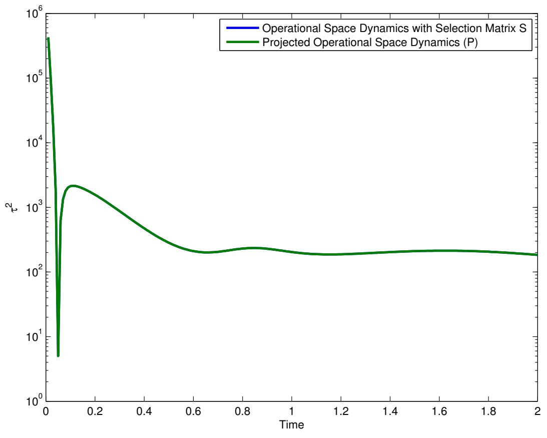


FIGURE 3.5: Comparison of the sum of squared torques commanded by the controllers. There is no difference between the commands produced by the two controllers. The plot is semilogarithmic.

2) the need to use a force sensor when a force feedback is necessary to define a force error, and finally 3) the relationship between hybrid control and optimal control.

3.1.9.1 A priori knowledge

A critical point when dealing with compensations based on dynamics, and also kinematics in general, is the assumption of a perfect knowledge of the model and parameters of the robot and also knowledge about the environment, such as the geometry of the constraints and the timing of the contacts. When this knowledge is not known exactly, stability of the system can be jeopardised. Hybrid control compensates for the dynamics of the robot and expresses the desired behaviour at the same time. Thus, with uncertainty on parameter values, the compensation might not be perfect and the behaviours obtained might be different from the desired ones specified through the control framework.

There are variants to the scheme which attempt to overcome this issue, and other studies proposing conditions on the stability of the system even in the presence of model uncertainties. An example to potentially solve the uncertainty problem is presented in Lozano and Brogliato [46], where the authors propose adaptive hybrid motion/force control for redundant manipulators. Results are reported which suggest that all quantities remain bounded and position and force errors converge to zero. Adaptive control was already used for robot control as reviewed in Hsia [47] and later illustrated in Slotine and Li [48]. The work in [49] extends model-based adaptive control to contact situations and proposes to use a projector in the form $\mathbf{P} = \mathbf{I} - \mathbf{J}_C^T \mathbf{J}_C$ to achieve a so-called joint space orthogonalisation, where motion signals are orthogonal to force vectors. This feature is stated to be a generalisation of hybrid motion/force control. The focus of this review is not on adaptive control, but it is worthwhile to mention this alternative approach.

To explore the impact of potential mismatches between models and real systems, Cheah et al. [50] study the stability of hybrid control in the presence of model uncertainties, namely on the constraint Jacobian. They derive sufficient conditions on the choice of the feedback gains and the estimation of the Jacobian in order to retain stability.

3.1.9.2 Force Feedback vs No Force Feedback

Most of the hybrid control schemes depend on a feedback of force to define an error in force directly. This implies the need of a force/torque sensor mounted at the contact point. If the contact may happen only at the end effector, then the problem might be solved with the use of such a sensor. Otherwise a tactile skin could be used, but with an increase in the complexity of the system and an increase of the overall cost. An alternative approach is the one proposed in Magrini et al. [51] where, by means of a vision system capable of recognising contacts on the whole surface of the robot, they propose a virtual force sensor using the so-called method of the residuals, and they are able to estimate contact forces. These quantities can then be used to close the force feedback loop.

It is also possible to avoid the need for a force sensor and an example is the approach in [1], which does not need a force/torque sensor. The force controller first compensates for the dynamics of the robot using a model of the robot and measurements from the proprioceptive sensors (joint angles) and then provides a desired value for the active force in the form of a feedforward force component. Possibly joint torque readings might also be used in order to recover contact forces, in the same fashion as in [51].

3.1.9.3 Relationship between Hybrid Control and Optimal Control

The use of optimisation methods in robotics is becoming increasingly popular. Optimisation algorithms become available once the problem can be formulated as a set of linear and nonlinear equations with a cost function to be minimised. The means to achieve this is to define the problem of trajectory finding as an optimisation problem. The work in [52] poses a discrete model of dynamics of contacts as a Linear Complementarity Problem accounting for friction as well. In addition, the work in [53] presents improvements to a Model Predictive Control approach applicable to high-degree-of-freedom systems. On the same line of reasoning, inelastic impacts and friction during contacts are accounted for in [54] and [55], where the optimisation is used to find a trajectory and to resolve contact forces while intrinsically satisfying the constraints. Contact forces are resolved as additional variables in the optimisation. A benefit of this method is that it does not require the definition of modes to handle contacts, as in [56]. Also, an analysis of the stability of systems undergoing impacts in the presence of friction is presented in [57].

Optimisation-based control techniques focus on control and motion planning encompassing transitions from free motion to contact and vice versa, which hybrid motion/force control does not include. Generally in optimisation based methods, the definition of constraints represents the means to specify general features of the system and to take into account the model of the system, *i.e.*, imposing constraints on dynamic consistency is the connection between optimisation techniques and the classical approaches. In the case of hybrid motion/force control, this is done by using the model in the controller.

3.1.10 Summary

Motivated by the fast increasing use of robotic platforms in interactive tasks involving contacts between the robots and the environment, we reviewed the literature around hybrid control and advanced a unitary notation in order to ease the comparison between the great variety of control schemes proposed over the previous several decades. From the initial formulation dating back to [25], [16], [26] and [27], until now, several eminent researchers have addressed and tackled the problem. We analysed fundamental features of each scheme and Table 3.1 concisely lists the key papers surveyed in this work, highlighting their major contribution.

A critical contribution of this Section is the analysis of the decoupling. We report simulative results and show that a perfect decoupling can be achieved using the selection matrix $\mathbf{S}$. Using projected dynamics and the projector $\mathbf{P}$, it is indeed possible to lower significantly the torques needed to perform the task at the cost of sacrificing decoupling. We have shown how decoupling can be regained using an extension to the scheme, but at the cost of sacrificing the advantage of lowering the torques, so that there is a substantial equivalence of this scheme to the one using the selection matrix.

As final consideration and remark, we have not claimed and we are not stating now that hybrid control is the only correct way to control a robot in such situations. We are advocating, though, that this is a very natural control framework to be used for tasks involving both motion and force control and for this reason, we chose to devote a review to it.

Table 3.1: List of contributions

Main Contribution	**Paper**
Admittance control	[22], Whitney, 1977
Selection matrix	[25], [16], Mason, 1979, 1981
Hybrid control	[26], Craig and Raibert, 1979
Joint stiffness matrix	[24], Salisbury, 1980
Hybrid control, use of inverse Jacobian	[27], Raibert and Craig, 1981
Modified hybrid control	[28], Zhang and Paul, 1985
P projector	[35], West and Asada, 1985
Operational space dynamics	[5], [6], Khatib, 1986, 1987
Dynamic hybrid control	[34], Yoshikawa, 1987
Instab. by counterexamples	[29], An and Hollerbach, 1987
Stability closed-loop	[32], McClamroch and Wang, 1988
Feedback linearisation	[36], Yoshikawa et al., 1988
Instability of classical hybrid control	[30], Zhang, 1989
Correct general position formulation	[31], Fisher and Mujtaba, 1992
Stability with Lyapunov (using dyn)	[33], Yabuta, 1992
Online estimation of selection matrix	[37], Yoshikawa and Sudou, 1993
Parallel force/position control	[38], Chiaverini and Sciavicco, 1993
Third subspace	[40], De Luca and Manes, 1994
Projectors overcoming selection matrix	[41], Featherstone et al., 1998
Projected dynamics	[8], Aghili, 2005
Operational space projected dynamics	[7], Mistry and Righetti, 2011
Applications of op. space projected dyn	[1], Ortenzi et al., 2014

3.2 Contact Constraint Estimation

The robotic revolution in manufacturing industries, over the past four decades, predominantly relied on robots which move payloads through unobstructed trajectories in free-space. In contrast, a new generation of robots is now needed which must cope with complex tasks, in uncertain environments, which frequently will involve forceful contacts between parts of the robot and surrounding objects or surfaces.

Many behaviours can be described as performing tasks under a set of contact constraints. For example, when a robot interacts with a horizontal surface (Fig 3.6), the end effector cannot penetrate the table. Some examples of tasks involving contacts include: robotic grasping, active perception and manipulation; foot contacts in legged walking robots; grinding and polishing of cast parts in manufacturing; and tasks needed for nuclear decommissioning, such as “disruption” (cutting) or “scabbling” (grinding off the contaminated surface of a concrete room or “cave” to make it safe).

In Chapter 5, a method is presented for achieving reliable estimations of the constraints arising from contacts which limit the free motion space of the robot. We propose to use only kinematic observations derived from basic proprioception (rotation encoders at

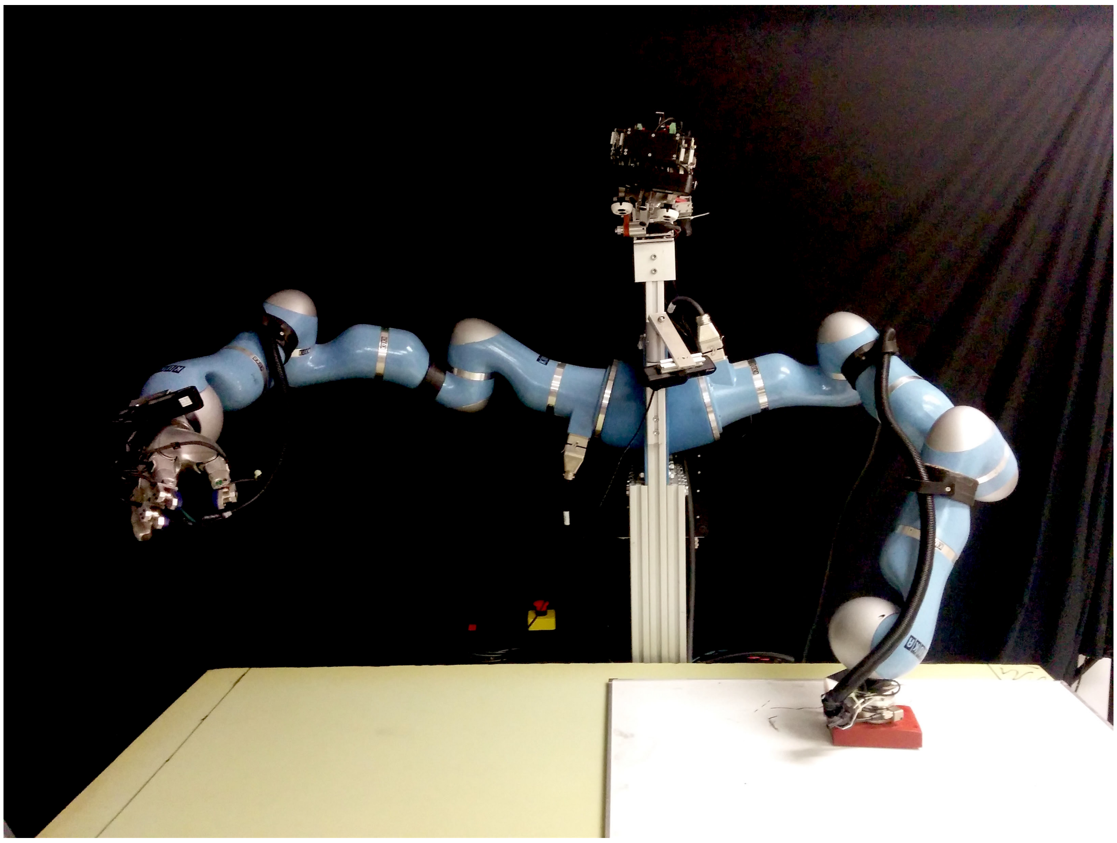


Figure 3.6: Our bi-manual half-humanoid platform, Boris, with one of its end effectors in contact with a horizontal surface. This contact prevents the end effector from moving in the negative z direction of the task space.

joints). Therefore, in contrast to much of the related literature, no added sensors are needed such as force-torque sensors or tactile sensors at the contact points, or feedback from a vision system.

Our proposed method would therefore be particularly useful for highly underactuated robotic arms, fingers or legs which contain passive (*e.g.*, spring-loaded) joints, provided that basic position/rotation sensing is available at each joint. Such passive/underactuated robots are attracting increasing attention from the research community, *e.g.*, [58, 59]. Additionally, in extreme environments such as nuclear decommissioning, beta and gamma radiation can destroy the delicate electronics needed for force, torque or tactile sensing. While intense gamma radiation can also destroy conventional proprioceptive rotation encoders (by causing the glass of optical encoders to become opaque), these can be replaced by electro-magnetic resolvers so that proprioception of each joint's rotation can still be reliably sensed in such environments. Our method could therefore be used to help such robots estimate the position and direction of constraint surfaces, *e.g.*, for tasks such as scabbling (described above).

There is an extensive body of literature on the use of computer vision to infer object shapes and features, [60–63]. There is also rapidly growing interest in estimation of object shapes and contacts using force-torque and tactile sensors, [64–70]. Recent work, [71, 72] has also begun exploring the fusion of visual and tactile data. In [73], a quadrupedal robot estimates the inclination of a planar surface on which it is trotting, by fusing data from IMU accelerometers with optical force sensors at each foot and the kinematics of each leg. In the context of hybrid motion/force control, [74] proposed a method for estimating the local shape of a constraint surface by combining position and end-effector force measurements. Another method for estimating constraints is proposed in [75], however this method learns the null space projector of an unknown task constraint from human demonstration.

In contrast to the above approaches, our method locally estimates the kinematic constraints due to contacts, without using any additional sensors (force, torque, tactile, vision) other than basic proprioception (rotation encoders at each joint). Specifically, we propose to perform a set of explorative actions and then estimate the kinematic constraints by observing the resulting motions. We present two variants, based either on a Cartesian space analysis or on a joint space analysis, and we also show how to discern unilateral constraints (*e.g.*, contact with a rigid surface which only constrains motion in one direction). We first demonstrate our method with a simulated redundant 3 DOF planar robot, and show how it can detect and estimate various kinds of constraints. Next we analyse the effect of different levels of observation noise on the accuracy with which such constraints can be estimated. Finally we show the results of experiments carried out using a KUKA LWR IV robot arm, which is tasked with estimating the environmental constraints when contacting surfaces of different inclinations.

Our lab currently lacks passively compliant, underactuated robots. Therefore, for proof of principle we have instead used an actively-compliant KUKA LWR IV robot to demonstrate our method. We only make use of this robot's proprioceptive rotation encoders at each joint, and we do not explicitly make any use of joint torque information in our experiments. However, this robot does use torque sensing internally for low level control, to achieve compliant behaviours when in contact with environmental constraint surfaces.

3.3 Vision-based State Estimation

The main goal of our framework is to estimate the state of an under-sensored manipulator thus producing quantitative feedback to be used in a controller. Although deeply influenced by the visual servoing (VS) and articulated body tracking literatures, we are neither interested in controlling the robot directly using visual features, as in a VS paradigm, nor solely in visual tracking of the body of the robot. The VS literature predominantly relies on accurately knowing robot states derived from joint encoders, which are not available in our case. Furthermore, this work endeavours to find a balance between computational speed, performance accuracy, robustness to real-world conditions, and monetary cost. The choice of a single monocular camera represents an efficient and robust solution in terms of cost and reliability in nuclear or other extreme environments, where variable range, strong variations in lighting, reflective surfaces, outdoor sunlight conditions, or dust can cause the performance of many kinds of depth sensors to deteriorate.

In order to use any information gathered from an image, it is most important to understand where the visual sensor is with respect to the world reference frame or to the robot reference frame or to any reference frame of importance. This process is called camera calibration.

In [76] Tsai explains that:

> camera calibration in the context of three-dimensional (3D) machine vision is the process of determining the internal camera geometric and optical characteristics (intrinsic parameters) and/or the 3D position and orientation of the camera frame relative to a certain world coordinate system (extrinsic parameters), for the following purposes: 1. inferring 3D information from computer image coordinates . . . and 2. inferring 2D computer image coordinates from 3D information.

Also known as camera resectioning, camera calibration is thus the process to estimate intrinsic and extrinsic parameters of the camera matrix bonding 3D Cartesian points and 2D camera points correspondences.

The work of this Thesis is related also to visual servoing. Visual servoing is a technique used to servo a robot using information coming from a visual sensor to close the control loop. Visual Servoing (VS) methods are classified into three broad categories: Image-Based VS, where the error is defined between current and desired visual features; Position-Based VS, where the visual features are used to estimate the 3D information and the error is defined between these and the desired 3D information; and finally hybrid approaches.

More details can be found in Ch. 24, Visual Servoing and Visual Tracking, Chaumette, Hutchinson, [10] and in Ch. 10, Robotics: Modeling, Planning and Control, Siciliano, Sciavicco, Villani, Oriolo, [11]. A survey on IBVS, PBVS and hybrid VS is reported also in the work of Kragic and Christensen, [77].

There is a very large literature on VS. For example, Weiss et al. [78] formalise analytically robot visual control and focus on an adaptive IBVS scheme. The work of Espiau et al. [79] covers the basics of VS and pays particular attention on the computation of the interaction matrix. Hutchinson et al. presented a tutorial on Visual Servo Control in [80], analysing the basics, the two major classes, PB and IB, among other things. Chaumette and Hutchinson then released a two-part tutorial, [81] and [82], where the first article revises the classical theory of Visual Servoing while the second one deals with newer techniques, taking into account issues regarding robot dynamics and/or nonholonomic constraints and data fusion to name only a few. Vincze [83] studies the dynamic performance of Visual Servoing. Wilson et al. [84] describe a PBVS technique featuring an extended Kalman filter to use a redundant number of known points on the target object.

Along with the classical PBVS and IBVS, there are other techniques proposed in literature. Castaño and Hutchinson [85] propose a hybrid structure to achieve visual compliance especially for grasping tasks. Malis et al. [86] introduce the so-called 2-1/2 D Visual Servoing, highlighting the drawbacks of classical IBVS and PBVS.

Drummond and Cipolla [87] propose a camera-in-hand setting and the use of an active contour whose deformations are limited to the Lie group of transformations. Fox and Hutchinson [88] introduce visual constraints surfaces for motion plans, in particular to devise critical velocity orientations. Chaumette et al. [89] describe a Visual Servoing scheme to put the robot in a determined position with reference to a target object to track and simultaneously estimate it's velocity. Dixon et al. [90] put forward a system with two cameras, one in-hand, the other fixed and use Lyapunov analysis to devise a control law to ensure a uniformly ultimately bounded tracking under uncertainties in camera's parameters.

Visual Servoing is not used only in open-chain robots. Dallej et al. [91] propose a Visual Servoing scheme to control an I4R parallel robot using edges extracted from the images. The work in [92] instead focuses on servoing the legs of a Gough-Stewart parallel robot.

As in many other problems, Visual Servoing has also been investigated from a human point of view. Hu et al. [93] analyse the human grasp with respect to the geometry of the target, claiming these parameters can be used to improve robot control.

Also, visual tracking can be defined in the Visual Servoing paradigm as having certain visual features, possibly changing over time, in a suitable position on the image plane. As examples, Papanikolopoulos et al. [94] propose visual tracking of moving targets using cross-correlation and an adaptive control to compensate for errors and parameter uncertainties while Comport et al. [95] analyse another real-time visual tracking framework.

Previously, Marchand et al. [96] demonstrated an eye-to-hand visual servoing scheme to control a robot with no proprioceptive sensors. In order to compute the Jacobian of the manipulator, they need to estimate the robot configuration. Thus, they feed the end effector position to an inverse kinematics algorithm for the non-redundant manipulator. Our work is related to this approach, but overcomes the redundancy problem by simultaneously tracking multiple parts of the robot, consequently having more relationships constraining the configuration, making our method potentially applicable to high DOF robots. In [97], a model-based tracker was presented to track and estimate the configuration as well as the pose of an articulated object. In this work, an extended Kalman filter was used to update the object configuration using tracked feature locations. Our approach is marginally related to this work. However, the major differences are that we simultaneously track entire 3D models of various parts of an articulated object and separately define an optimisation problem to estimate the joint values using the tracked poses.

Within the visual servoing literature, virtual visual servoing is an approach which has profoundly influenced this work. Virtual visual servoing has been introduced in [98] and [99] as a new framework for augmented reality. Its principle is:

> to define the pose computation problem as the dual problem of (2D) Visual Servoing [99].

Gratal et al. [100] use VVS in pose estimation and tracking problems to align a rendered graphical model with the current image from the actual camera. They use a 3D model of the robot (a robot hand) and render virtual images of it. They extract visual features and define a difference between the desired features, *i.e.*, the ones extracted from the image from the actual camera, and the ones coming from the synthetic view. This difference guides a VS control to change the pose of the model till the difference vector is smaller than a certain threshold. At this point the pose is recovered by the model. The authors propose to use distances between point features as features, *i.e.*, the Chamfer distance between points and their closest match on the real image features. The set of points are edges computed with Sobel or Canny operators. Gratal et al. focus on pose estimation but considered the robot hand as a rigid object, *i.e.*, the robot hand has constant joint values. In [101] they use a humanoid robot but either they assume a known joint configuration or they introduce some error in the initial estimate of the joint configuration, but they assume to have the transformation with respect to the base of the manipulator. It is not clear how they reconstruct the joint configuration if any.

VVS has also been used in Augmented Reality applications like in [102] and [103]. Here the VVS approach is deployed together with an M-estimator for the displacement estimation between two consecutive images of a video using only 2D information.

As previously stated, our main objective is to estimate the robot joint configuration, a problem which is also related to pose estimation. Pose estimation is classically defined as the task to identify a single-body rigid object and estimate its 3D position and orientation (6 degrees of freedom) with respect to a world reference frame or to some other suitable reference frame.

Lowe [104] combines multiple images of an object into a model using SIFT features and a matching based on the Hough transform. Glasner et al. [105] present a technique that takes into account shape and appearance. A voting method and view-specific Support Vector Machines finalise the estimation from single 2D images. Savarese and Fei-Fei [106] propose to divide the objects into regions and connect these through homographic transformations. Sun et al. in [107] describe the so called Depth-Encoded Hough Voting scheme which uses depth information to learn the distributions of the features of the object. Leibe et al. present Implicit Shape Model to address simultaneously segmentation and categorisation in [108]. Hel-Or and Werman in [109] propose a way to combine 2D and 3D measurements to solve pose estimation. Mei et al. in [110] put forward a statistical method using spatiotemporal parts and joint distributions of parts. Payet and Todorovic [111] use features called bag of boundaries placed on a grid and then match the image under analysis with an optimisation providing a continuous pose estimation. Li et al. [112] instead deploy Gaussian Mixtures and the global optimisation algorithm Particle Swarm Optimisation to solve global shape alignment. The work in [113] uses Kernel Principal Component Analysis to first reduce the input space into a tractable feature space and then compute PCs in this feature space.

Usually, pose estimation is defined for single-body rigid objects, but there is work dealing with multi-body articulated objects as well. Hel-Or and Werman [114] propose a model-based pose estimation framework which takes into account physical constraints of articulated objects during the estimation, defined as strong constraints. The actual measurements are treated as soft constraints in an Extended Kalman Filter. Mulliganet al. [115] use edge-based Chamfer matching between the real image and a synthetic image and gradient descent algorithms to minimise this difference by adjusting joint angles of the synthetic models. In particular, besides what they call location and orientation constraints, they lower the dimensionality of this minimisation by constraining each link position to be dependent on the previous link position, using so-called quasi-separability constraint, *i.e.*, minimising the error for each link individually modifying only one parameter.

Then there are contributions that go into different directions but that give very interesting perspectives on the problem. Shahrokni et al. [116] propose a probabilistic classification to find lines when the perceived textures change. Wagner et al. [117] present modified versions of SIFT and Ferns features plus a template-matching tracker for real-time mobile phones applications.

Another slightly different 360-degree application of vision to robotics is object recognition for manipulation. An example for industrial purposes is reported in [118].

As always, the human being and in this context, the human body, has been studied in order to better understand its movements. Besides, this lets computer vision researchers understand better the problem of pose estimation for articulated objects. A survey of human motion analysis can be found in [119]. Shotton et al. in [120] propose a method to predict body joints positions on each frame using Randomized Decision Forests trained on very large datasets of real and synthetic images. Jenkins et al. [121] present a method that uses a range of learned motion primitives that builds the action space so that a particle filter algorithm infers the pose.

Besides, human-robot interaction is a thriving field. Goodrich et al. [122] highlight seven principles to have a working interaction. Kulić and Croft [123] study human signals to estimate human intent. In [124] they present two planning criteria for the interaction while in [125] they propose a Hidden Markov Model to estimate human state for human-robot interaction. Kanda et al. instead analyse such interactions through a study of the body movements in [126]. In [127] the authors address the problem of a human controlling a robot with free hand gestures.

So, articulated objects are composed of multiple rigid bodies and possess higher DOF (often redundant). There are also a number of kinematic (and potentially dynamic) constraints that bind together the bodies belonging to kinematic chains. Further, these constraints can also be used to locate and track the chain of robot parts. In this work, for the sake of modularity, the kinematic constraints are used only when estimating the joint values, and not for visual tracking of robot parts, *i.e.*, each part of the robot is tracked individually, and then a separate stage of our architecture performs best-fitting of the robot kinematics to the tracked part positions.

A variety of ways to track articulated bodies can be found in [97, 128–131]. In contrast to our work, these authors mainly focused on localising parts of the articulated bodies in each image frame, and not on the estimation of joint angles between the connected parts. Additionally, much of this work focussed on tracking parts of robots, but made use of information from the robot's joint encoders to do so, in contrast to the problem posed in our paper.

A real-time system to track multiple articulated objects using RGB-D and joint encoder information is presented in [132]. A similar approach was used in [133] to track and estimate the pose of a robot manipulator. SimTrack [134] is also a framework for real-time robot tracking using cameras, a Kinect and the joint angles. In [135], a marker-tracking method was used to identify the joint origins of robots. Other notable examples can be found in [136] and [137], where the authors propose to use depth information for better tracking of objects. Recently, an approach based on regression forests has been proposed to directly estimate joint angles using single depth images in [138]. Additionally, in the context of our work, it is worth mentioning some of the human hand pose tracking methods presented in, *e.g.*, [120, 139, 140]. However, most of these methods require either posterior information (*e.g.*, post-processing of entire image sequences offline to best-fit a set of object poses), or require depth images, or must be implemented on a GPU to achieve online tracking. In contrast, our approach does not make use of depth information, does not require visual tracking of the entire robot, and does not require special markers to be attached to the robot. Instead, a small set of the robot's parts are tracked to estimate the joint configuration. We detail the choice of these robot parts in Sec. 6.1.2.

In summary, the use of depth information alongside standard RGB images can improve the tracking performances. However, it also increases the computational burden and decreases robustness in many real-world applications. Our choice of using only a simple, monocular 2D camera is motivated by cost, robustness to real-world conditions, and also in an attempt to be as computationally fast as possible.

Chapter 4

Robot Control based on Projected Inverse Dynamics

As robotics becomes progressively pervasive in real-world applications, it is becoming increasingly necessary for robots to be aware of what is happening around them and to interact with their environments and other agents. Firstly, these interactions should be safe and not cause damage to the robot or its surroundings. Secondly, and perhaps more interestingly, robots should ideally understand and exploit their environments in order to perform tasks more efficiently. Humans often exploit contacts with their environments to perform tasks more comfortably and more efficiently. For example, a person reaching to grasp an object further off on a table, usually leans on the table to support their weight during reaching and grasping. Alternatively, a person sweeping the floor can also lean on the broom handle to partially support their weight, thereby reducing the required joint torques and resulting muscle tensions.

This Chapter addresses the problem of constrained motion for a manipulator performing a task while in contact with the environment, and proposes a solution based on projected operational space dynamics. The main advantages of this control technique are: 1) it exploits the environment contact constraint, so as to minimise the joint torques needed to perform the task; 2) it enables decoupling of motion and force control; 3) force feedback from a force sensor mounted at the end effector or other contact points is not needed.

This Chapter is motivated by the analysis of such contact usage and is intended as an early step towards making better use of contacts in robotics, and enabling robots to mimic useful human strategies for exploiting such contacts. For proof of principle, we demonstrate this idea with an example implementation, in which a robot uses an eraser to wipe a whiteboard. Effectively, the robot is also “leaning” on the whiteboard while executing the wiping motion, so as to reduce the motor torques required to maintain the desired arm positions. We have implemented this experiment on our bi-manual humanoid torso robot “Boris”, see fig. 4.1.

In Sec. 4.1, we present an experimental implementation of the control method in which a KUKA LWR IV manipulator uses an eraser to wipe a whiteboard, and we show that this controller can effectively exploit contact with the whiteboard in order to reduce joint torques while still performing the desired wiping motion.

In Sec. 4.2, we propose a comparison between two force control frameworks, one based on projected inverse dynamics, and one based on optimal control. Firstly, we propose a control method based on projected inverse dynamics, which directly exploits the contact constraints to minimise the instantaneous joint torques needed to perform a task, as in Sec. 4.1. Secondly, we propose an optimal control strategy which provides a tool to minimise the joint torques over an interval of time. We show how contact constraints can be used as optimisation constraints in the definition of the problem, and how to formulate the optimal control problem directly using projected dynamics. Initially we explore a positional control problem, where the robot is required to follow a desired path, and show that both of the proposed methods can satisfy the positional task while significantly reducing the joint torques as compared to simple kinematic control and also classical inverse dynamics control. We also show that the proposed optimal control method outperforms the pure projected inverse dynamics method in terms of minimising the required joint torques. We then show how each method can be extended to follow a desired path while also exerting a desired contact force. Again, the method incorporating optimal control is shown to satisfy the task requirements with significantly smaller commanded torques than the pure projected inverse dynamics method. To confirm the analysis, and demonstrate proof of concept, we present the results of empirical experiments with a simulated 3-degree-of-freedom planar manipulator which is constrained to move while in contact with a rigid surface.

In contrast to reviewed works, the main contribution of this Chapter is to formulate the optimal control problem directly using projected inverse dynamics. We then compare this control technique with one based on purely projected inverse dynamics. Specifically, we focus on the commanded torques, showing that it is indeed possible to reduce torques while also performing the desired task. We show that the control method based on projected inverse dynamics outperforms both a pure kinematic controller and a classical inverse dynamics controller in the sense that it sends lower commanded torques to the robot. Furthermore, the commanded torques produced by the optimal control framework are substantially reduced with respect to the ones required by projected inverse dynamics control. We also show how both controllers can be extended to impose a desired interactive force, in addition to performing the desired positional task. These

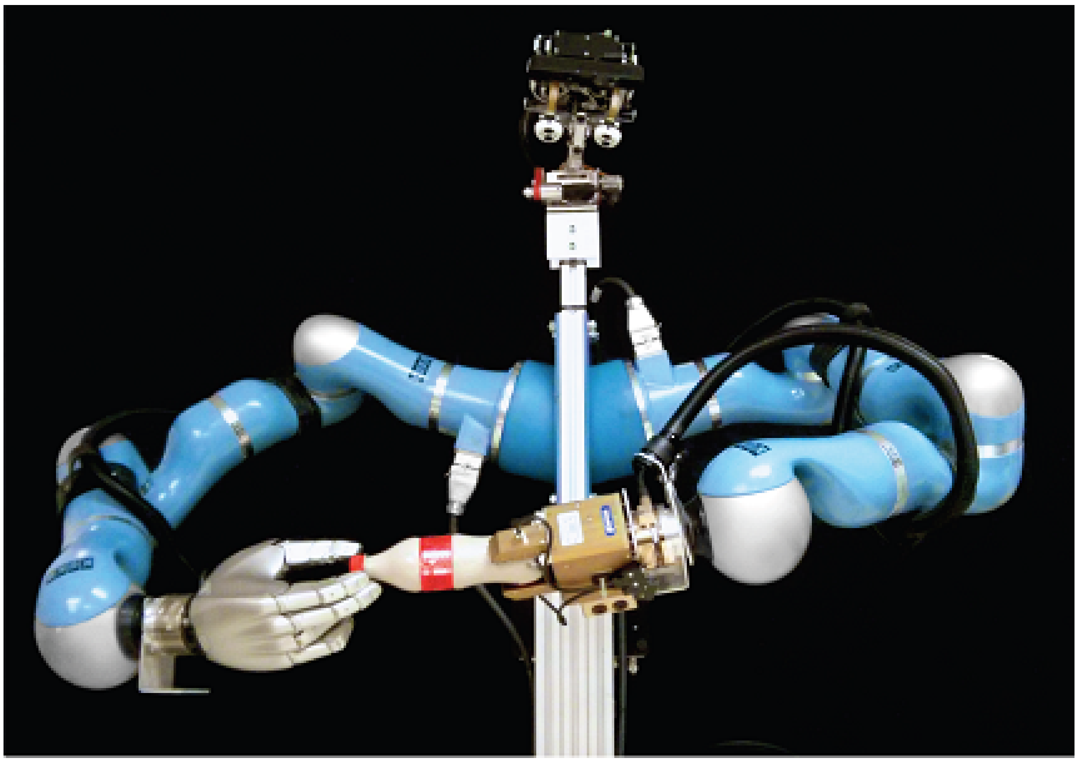


FIGURE 4.1: Boris - the bi-manual humanoid robot at University of Birmingham, UK, used for our experiments.

extensions are enabled respectively by: implementing a controller based on projected inverse dynamics with a force constraint term; and another controller which uses the solution provided by optimal control for the position task while using projected dynamics for the force task. As with the purely positional task, we also show that in the position-plus-force task, the commanded torques are much lower for the controller that uses the optimal control solution for the position task and projected dynamics for the force task.

The remainder of this Chapter is organised as follows. Section 4.1 describes several experiments to empirically measure and compare joint torques during task execution, using three variants of the proposed controller and a conventional comparison controller. Section 4.2 proposes a comparison between two force control strategies, namely one based on projected dynamics and one based on projected dynamics and optimal control. Subsection 4.2.1 introduces the control framework based on optimal control. Subsection 4.2.2 reports results obtained in a number of simulations. Finally Section 4.3 discusses the results and provides concluding remarks and suggestions for future work.

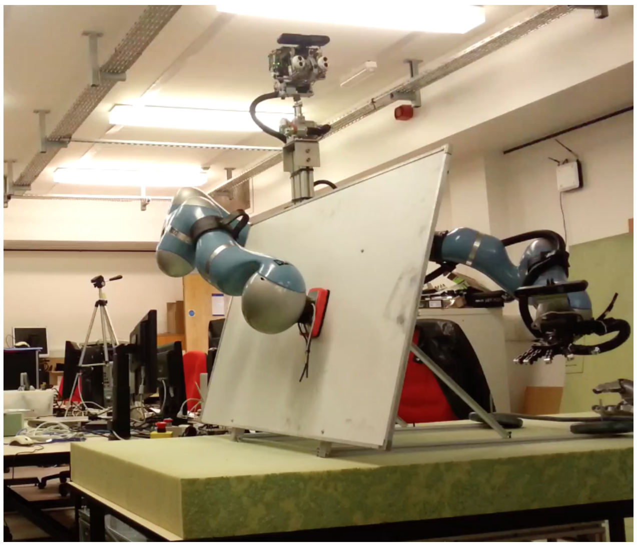


FIGURE 4.2: Experimental setting. We used the right arm of the humanoid platform Boris, and placed the board in between its two arms so that the z axis of the robot end effector frame, and the axis orthogonal to the board are parallel.

4.1 Experimental Study

To investigate projected operational space control of a robot during contacts, we have carried out several empirical experiments using a KUKA LWR IV manipulator. The robot is mounted on a humanoid torso platform, Boris, and a whiteboard has been placed between the two arms of Boris so that the z axis of the robot end effector frame, and the axis orthogonal to the whiteboard plane are parallel. Fig. 4.2 illustrates this experimental setup.

The experiments in this Section assume that the environmental constraints are known a priori, (*e.g.*, we assume we know the plane of the whiteboard which our robot is tasked with wiping). These assumptions seem to be reasonable for task. In Chapter 5, we propose a method to estimate the kinematic constraints due to contacts with the environment, using only kinematics, *i.e.*, performing an exploration when a contact is recognised, and analysing the set of observed velocities which differ from the expected velocities.

To control the robot during these experiments, we use the formulation proposed by Mistry and Righetti [7]

$$\boldsymbol{\tau} = \mathbf{P}\mathbf{J}^T\mathbf{F} + \mathbf{P}\mathbf{N}\boldsymbol{\tau}_0 + \boldsymbol{\tau}_C\,, \tag{4.1}$$

where

$$\mathbf{N} = \mathbf{I} - \mathbf{J}^T\mathbf{J}^{T\#} \tag{4.2}$$

and

$$\mathbf{J}^{T\#} = (\mathbf{J}\mathbf{M}_C^{-1}\mathbf{P}\mathbf{J}^T)^{-1}\mathbf{J}\mathbf{M}_C^{-1}\mathbf{P}\,, \tag{4.3}$$

in order to be dynamically consistent with the additional constraints. In particular, $\mathbf{P}\mathbf{J}^T\mathbf{F}$ is responsible for the end-effector motions, $\mathbf{P}\mathbf{N}\boldsymbol{\tau}_0$ does not generate accelerations on the end-effector (null space term) and can be used for the optimisation of an objective function, while $\boldsymbol{\tau}_C$ generates constraint forces but has no impact on motion.

$\boldsymbol{\tau}_C$ is defined as in [7]

$$\boldsymbol{\tau}_C = (\mathbf{I} - \mathbf{P})\mathbf{h} + (\mathbf{I} - \mathbf{P})\mathbf{M}\mathbf{M}_C^{-1}(\mathbf{P}\boldsymbol{\tau} - \mathbf{P}\mathbf{h} + \mathbf{C}_C\dot{\mathbf{q}}) + \mathbf{J}_C^T\boldsymbol{\lambda}_{des}\,. \tag{4.4}$$

This can be regarded as a means of controlling a desired interaction at the point of contact. In a sense, this term realigns this control technique to hybrid force/motion control. On the other hand, our approach also aims to reduce and minimise joint torques while exploiting constraint forces. Therefore $\boldsymbol{\tau}_C$ must be used only if strictly necessary, since it increases the commanded joint torques. Furthermore, $\boldsymbol{\tau}_C$ can actually be used to set a desired interaction force $\boldsymbol{\lambda}_{des}$ so as to effectively decouple motion and force control or, since this entire formulation assumes bilateral constraints, to enforce possible unilateral constraints. For example, our illustrative problem of the whiteboard is unilateral, in that it can only exert a constraining contact force on the end effector in one direction (away from the whiteboard). Once the end effector is perturbed away from the whiteboard surface, then the unilateral constraint disappears, and then the desired interaction force term is necessary in order to encourage the robot to return back towards the desired task of moving in the whiteboard plane.

The robot's task is defined as following a predefined trajectory on the whiteboard surface, keeping the orientation fixed (although this might be relaxed, being not critical for the sake of this study) while in contact with the board. For our experiments we assumed the robot to be already positioned on the task at $t = 0s$, but the controller can cope with this being relaxed in general.

More specifically, the trajectory is circular and the robot is controlled using Eqn. 4.1 with

$$\mathbf{F} = \boldsymbol{\Lambda}_C\ddot{\mathbf{y}} + \boldsymbol{\Lambda}_C(\mathbf{J}\mathbf{M}_C^{-1}\mathbf{P}\mathbf{h} - (\dot{\mathbf{J}} + \mathbf{J}\mathbf{M}_C^{-1}\mathbf{C}_C)\dot{\mathbf{q}}) \tag{4.5}$$

and

$$\boldsymbol{\tau}_0 = -\mathbf{K}_d\dot{\mathbf{q}} + \mathbf{K}_p(\mathbf{q}_d - \mathbf{q})\,, \tag{4.6}$$

where $\boldsymbol{\tau}_0$ is used to prevent a drift in the null space and $\boldsymbol{\tau}_C$ is defined as in Eqn. 4.4.

As already mentioned, this entire formulation assumes bilateral constraints while in our case, wiping a board, the constraint on the z axis is only unilateral. For this reason, the $\boldsymbol{\lambda}_{des}$ term is used to emulate the bilateral constraint. Thus the $\boldsymbol{\tau}_C$ term can be used to compensate for the robot dynamics at times when the bilateral constraint assumption would otherwise be violated. We modelled this task as having a single geometric constraint, of limiting motion on the axis orthogonal to the board. The controller then does not control motion in that direction.

Our experiments compare the results of projected operation space control against results obtained by using a conventional kinematic controller, which sends only position commands to the robot. This comparison is meaningful, because both controllers share a common task of causing the end effector to move along a desired circular motion (essentially a position control task). The difference is that the proposed projected operation space controller seeks to do this while also exploiting the environmental contact to minimise joint torques.

In contrast to hybrid force/motion control, we are not explicitly controlling forces so a comparison would not be meaningful in this case. In contrast to impedance control, we are not explicitly defining a desired behaviour of the robot interaction with the environment, and therefore a comparison against impedance control would also be misleading. For these reasons, we have decided to compare the proposed approach against the simple kinematic controller, in order to demonstrate how this method takes advantage of the constraints to enable performance of the desired task while employing reduced joint torques.

We conducted a total of four experiments. The first three experiments were performed using three different variants of the proposed projected operation space controller of Eqn. 4.1, while the fourth experiment was carried out using a conventional kinematic controller:

Exp. 1: Eqn. 4.1 with $\boldsymbol{\tau}_0 = \boldsymbol{\tau}_C = 0$, then the control law is reduced to $\boldsymbol{\tau} = \mathbf{P}\mathbf{J}^T\mathbf{F}$. In this case, we are computing the torques necessary to follow the trajectory;

Exp. 2: Eqn. 4.1 with $\boldsymbol{\tau}_C = 0$, then the control law is reduced to $\boldsymbol{\tau} = \mathbf{P}\mathbf{J}^T\mathbf{F}+\mathbf{P}\mathbf{N}\boldsymbol{\tau}_0$. In this version of the controller, besides providing the torques to perform the trajectory, a null space term is used in order to prevent a possible drift in the joint space and to stay close to the initial configuration;

Exp. 3: full Eqn. 4.1. Here $\boldsymbol{\tau}_C$ is used to enforce a virtual bilateral constraint (on a problem that is in practice unilateral due to the uni-directional normal contact force provided by the whiteboard) by means of specifying a desired force bias, λ_{des}. In our experiments we set $\lambda_{des} = -5$ N;

Exp. 4: we implemented a conventional kinematic controller which sends only position commands to the robot in order to execute the same circular motion in the same plane as the three previous experiments.

Table 4.1 reports the Root Mean Square Errors on the positions of the end effector during each experiment. The figures in the table show that projected operation space control delivers position errors (RMS error 4 to 5mm) that are somewhat larger but similar in magnitude to pure position control (RMS error 2mm). This difference is small compared to the workspace size and is negligible for the purposes of the experimental task (wiping the whiteboard). It is not surprising that the the kinematic controller has the lowest error, because it controls the robot's position directly. The highest RMS error is obtained when using $\boldsymbol{\tau}_C$, 5 mm. This is also unsurprising, because this variant of the controller imposes an additional force constraint on the system which is not present in the other three experiments.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
RMS error	0.0044	0.0042	0.0050	0.0020

TABLE 4.1: This table reports the root mean square error of the end effector position wrt tracking the desired end effector trajectory. The RMS error is 5mm or less in all four experiments.

Most importantly, we are interested in the comparison of the measured torques at the joints. The primary aim of this paper is to show how a robot can perform a desired trajectory, while exploiting contact constraints in order to minimise the joint torques. Fig. 4.3 reports the evolutions of the sum of squared measured torques in experiments 1, 2 and 4. The torques measured using two versions of the controller presented in this Section are generally lower as compared to those measured using the kinematic controller. This suggests that the proposed controllers can successfully reduce torques by taking advantage of the environmental contact constraints. Note that we do not compare experiment 3 here, because it involves a deliberate additional force term which naturally engenders larger torques, thus rendering such comparison meaningless.

Fig. 4.4 shows how adding the $\boldsymbol{\tau}_C$ term leads to increased torques in order to supply the additional specified end effector force. Note that the reason we have employed this additional task force is to mimic a bilateral constraint in a situation where the real constraints are only unilateral. In our case we set this desired force arbitrarily to -5 N. Using this additional term more sparingly will naturally lead to a decrease in the required joint torques. Alternatively, the term $\boldsymbol{\tau}_C$ could be used in conjunction with a boolean switch that activates such compensation only when the interaction force is below a threshold (*e.g.*, with the board wiping task, the threshold can be a minimal z axis force in the direction of the board, causing the robot to lightly push on the board

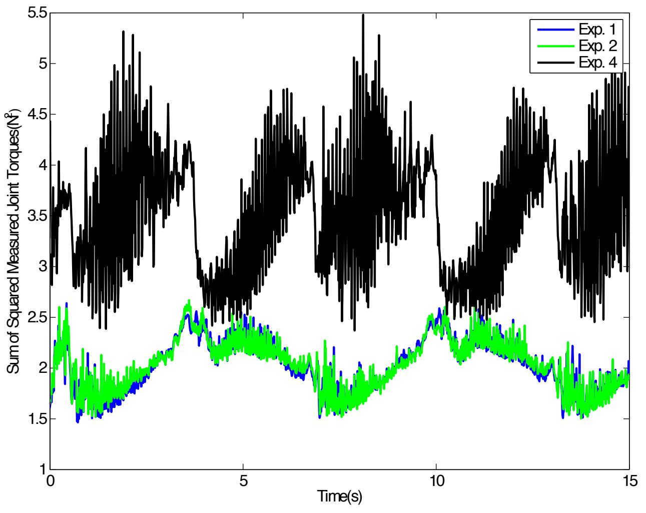


FIGURE 4.3: Comparison of the sum of squared measured joint torques. Exp. 1 and Exp. 2 are similar since the null space term in Exp. 2 does not have a big impact on the torques. Exp. 4 (conventional position control) shows significantly larger torques than Exp. 1 and Exp. 2 which are two variants of the proposed method.

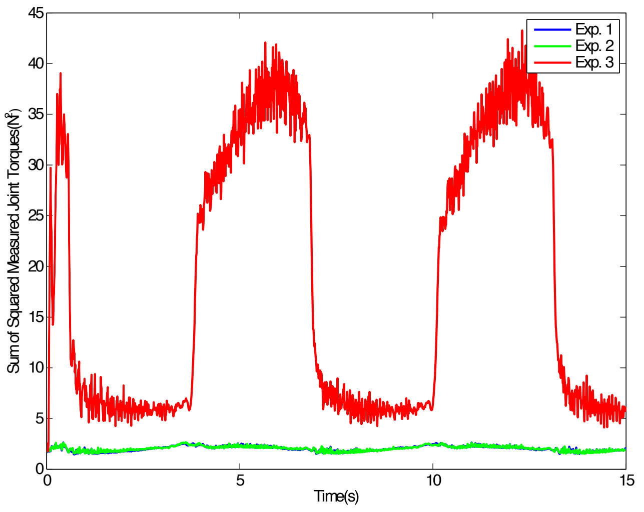


FIGURE 4.4: Comparison of the sum of squared measured joint torques. Exp. 3 has larger torques due to the $\boldsymbol{\tau}_C$ term. Reducing the size of this term would reduce the torques at the joints.

rather than losing contact with it). This switching approach will be investigated in future work.

There is no significant difference between the performances of Exp. 1 and Exp. 2, either in terms of positional accuracy or in terms of required torques. The null space term has a useful effect on improving the repeatability of the task, it avoids drifts in the task null space, and it gives a useful behaviour in the presence of external perturbations, without impacting performance as shown by the fact that the torques in Exp. 1 and Exp. 2 are very similar.

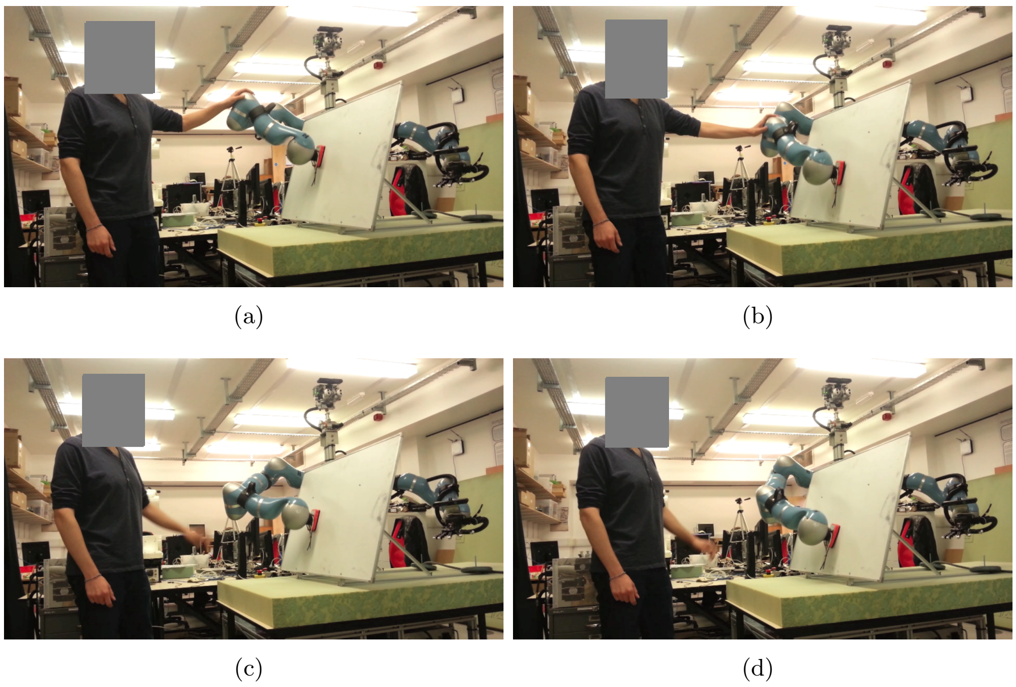
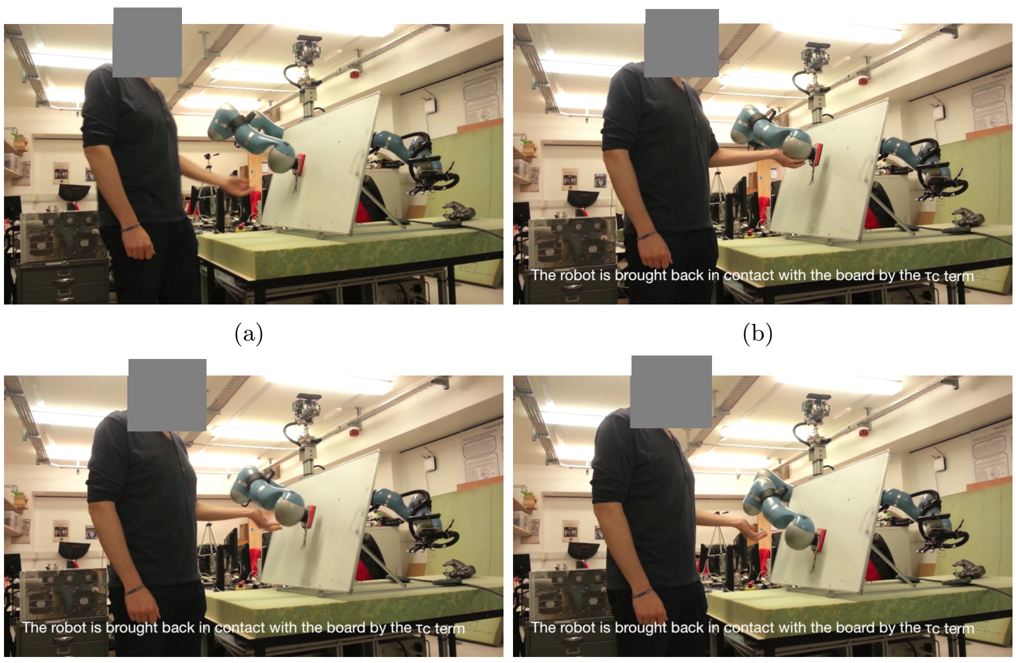


FIGURE 4.5: Sequence of images taken from the video attachment to Humanoids 2014. The control law is using $\boldsymbol{\tau} = \mathbf{PJ}^T\mathbf{F}$ (corresponding to experiment 1). The human operator perturbs the elbow of the robot, pushing it downwards in frames (a) and (b), thus perturbing the robot in the task null space. The robot uses its redundancy to continue correctly performing the task (the desired circular wiping motion) both during and after the disturbance, with minimal error induced to the task itself. Note that the robot does not recover the initial elbow configuration (d) since there is no such compensation (the $\mathbf{PN}\boldsymbol{\tau}_0$ term is absent).

Figs. 4.5 and 4.6 show frames taken from the video submitted in attachment to Humanoids 2014. The video shows the behaviour of the proposed controllers in the presence of external disturbances, first using only $\boldsymbol{\tau} = \mathbf{PJ}^T\mathbf{F}$ (corresponding to experiment 1) and then the entire term $\boldsymbol{\tau} = \mathbf{PJ}^T\mathbf{F} + \mathbf{PN}\boldsymbol{\tau}_0 + \boldsymbol{\tau}_C$ (corresponding to experiment 3). The first part of the video represents a test on the $\boldsymbol{\tau} = \mathbf{PJ}^T\mathbf{F}$ control law. It shows that the robot is capable of continuing to perform the task while being perturbed in the task null space, *i.e.*, the movements inducted on the elbow by the human operator do not prevent the robot from continuing to track the desired circular trajectory on the board. The second part of the video is an example of how to use the additional term $\boldsymbol{\tau}_C$ in order to specify an interaction force at the contact point and, in our case, to enforce a bilateral constraint when the whiteboard itself only provides a unilateral constraint. The video shows the human operator forcing the robot to lose contact with the board, after which the robot is able to successfully recover the contact and it returns to tracking the circular trajectory in the desired plane.


The robot is brought back in contact with the board by the τc term
The robot is brought back in contact with the board by the τc term
The robot is brought back in contact with the board by the τc term

(a) (b) (c) (d)

FIGURE 4.6: Second sequence of images taken from the video attachment to Humanoids 2014. In this case, the control law is $\boldsymbol{\tau} = \mathbf{P}\mathbf{J}^T\mathbf{F} + \mathbf{P}\mathbf{N}\boldsymbol{\tau}_0 + \boldsymbol{\tau}_C$. Thus when the human operator takes the robot end effector away from the board surface (a) and (b), thanks to the $\boldsymbol{\tau}_C$ term, as soon as the human operator releases it (c), the robot is able to recover the contact and the interaction on the contact surface (d). Due to the $\mathbf{P}\mathbf{N}\boldsymbol{\tau}_0$ term, the robot is also able to maintain an elbow configuration close the initial one (d).

4.2 Projected Inverse Dynamics Control and Optimal Control for Robots in Contact with the Environment: A Comparison

4.2.1 Optimal Control

Optimal control is a very flexible and powerful tool that can be used in a variety of situations. In this work, we formulate the control problem as one of finding a sequence of inputs (*i.e.*, commanded torques $\boldsymbol{\tau}$) and states (*i.e.*, $\mathbf{q}$ and $\dot{\mathbf{q}}$), such that the robot performs the task and the evolution of the states is dynamically consistent, while minimising a cost function. The cost function is dependent on both states and inputs, *i.e.*, it penalises rapid changes in states and also penalises high torques (the principal goal of this work).

In contrast to previous approaches, *e.g.*, [54], we define the dynamic model directly using projected dynamics, such that the problem definition becomes

$$\begin{aligned}
&\text{find} && \mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, t \in [0, T] \\
&\underset{\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}}{\text{minimize}} && \mathbf{q}^T \mathbf{Q}_q \mathbf{q} + \dot{\mathbf{q}}^T \mathbf{Q}_{\dot{\mathbf{q}}} \dot{\mathbf{q}} + \boldsymbol{\tau}^T \mathbf{R} \boldsymbol{\tau} \\
&\text{subject to} && \frac{\partial \mathbf{q}}{\partial t} = \dot{\mathbf{q}} \\
& && \frac{\partial \dot{\mathbf{q}}}{\partial t} = \mathbf{M}_C^{-1}(\mathbf{P}(\boldsymbol{\tau} - \mathbf{h}) + \mathbf{C}_C \dot{\mathbf{q}}) \\
& && \mathbf{J}_C \dot{\mathbf{q}} = 0 \\
& && \mathbf{q} \leq \mathbf{q}_{lim} \\
& && \dot{\mathbf{q}} \leq \dot{\mathbf{q}}_{lim} \\
& && \boldsymbol{\tau} \leq \boldsymbol{\tau}_{lim} \\
& && \mathbf{y}(T) = \mathbf{y}_d
\end{aligned}$$

where $\mathbf{M}_C = \mathbf{M} + \mathbf{PM} - (\mathbf{PM})^T$ and $\mathbf{C}_C = -\mathbf{M}\mathbf{J}_C^{\dagger}\dot{\mathbf{J}}_C$.

Within this framework we are then minimising torques over a time interval. This represents the main difference with respect to the approach described in the previous Section, which effectively minimises torques only instantaneously thanks to the $\mathbf{P}$ projector.

4.2.2 Results

As proof of concept, we ran simulations on a 3-degree-of-freedom planar manipulator with links of unitary mass and length, *i.e.*, $l = 1$m and $m = 1$kg, constrained to move parallel to the x axis of the workspace plane as in Fig. 4.7. The manipulator is asked to slide its end effector along a rigid wall (shown as thick black line), from the starting point (shown as a red dot), to the goal point (shown as a green dot). We used 200 simulation steps with $\delta t = 0.01s$ for a total simulation time of $2s$.

4.2.2.1 Comparison of four methods for a position task

We compare the results of the two control frameworks described in the previous sections with the ones obtained using a standard kinematic controller whose regulation law is as follows

$$\boldsymbol{\tau} = \mathbf{J}^T(\mathbf{q})[\mathbf{K}_P(\mathbf{y}_d - \mathbf{y})] - \mathbf{K}_D \dot{\mathbf{q}} + \mathbf{g}(\mathbf{q})\,, \tag{4.7}$$

where $\mathbf{g}(\mathbf{q})$ is the term for gravity cancellation.

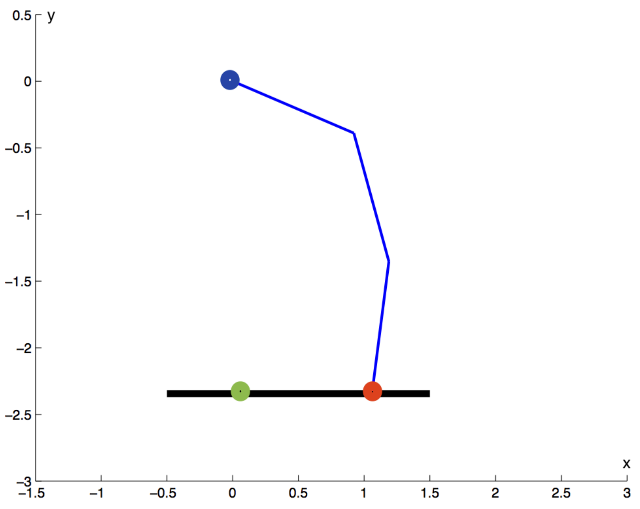


FIGURE 4.7: 3-degree-of-freedom planar manipulator asked to slide its end effector along a rigid wall (thick black line) from the starting point o to the goal point o. The base of the robot is set in the origin of the plane o.

We also compare against a classical inverse dynamics controller of the form

$$\boldsymbol{\tau} = \mathbf{M}\mathbf{J}^{\dagger}(\ddot{\mathbf{y}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{h}\,, \tag{4.8}$$

with $\ddot{\mathbf{y}}_r = \ddot{\mathbf{y}}_d + \mathbf{K}_D(\dot{\mathbf{y}}_d - \dot{\mathbf{y}}) + \mathbf{K}_P(\mathbf{y}_d - \mathbf{y})$. Both of these comparison methods disregard the constraint. In other words, they control the robot as if the constraint did not exist and take care of the whole position control as if the robot were in free motion. However, when integrating the model to compute $\mathbf{q}$, our simulator checks that the constraint is being respected.

We chose these comparison controllers because they fulfil the desired task (in terms of end effector poses), but do not take into account the constraints in order to generate the commanded torques to control the robot motion. The comparison methods are also chosen so as to be broadly representative of the most commonly used, conventional, classical methods of robot control. Unlike hybrid force/motion control, in this work we are not interested in explicitly controlling forces, *i.e.*, we are only interested in achieving a positional task, while trying to do so with minimum torque (and exploiting the environment to do so). Therefore, a direct comparison between our methods and hybrid force/motion control would not be meaningful here. Instead, we show that conventional classical methods of controlling position require greater torques than our proposed methods to achieve the same positions.

For implementing the controller based on projected dynamics, described in Sect. 4.1, we have used only motion control, *i.e.*, $\boldsymbol{\tau}_m$. In particular, we used the same control law as in the classical inverse dynamics controller, so we define

$$\boldsymbol{\tau} = \mathbf{P}\boldsymbol{\tau}_m \,, \tag{4.9}$$

where $\boldsymbol{\tau}_m = \mathbf{M}\mathbf{J}^{\dagger}(\ddot{\mathbf{y}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{h}$. This choice has been made to better justify a direct comparison and to highlight the contribution of the projector $\mathbf{P}$. Moreover, we are interested only in the motion control part, and this is the reason why there is no $\boldsymbol{\tau}_C$ component.

For implementing the optimal control law, the control problem is set up as defined in Subsect. 4.2.1. Particularly, we set $\mathbf{Q}_q = \mathbf{O}$, $\mathbf{Q}_{\dot{\mathbf{q}}} = \mathbf{I}$ and $\mathbf{R} = \mathbf{I}$ so that the cost function reduces to $\dot{\mathbf{q}}^T\dot{\mathbf{q}} + \boldsymbol{\tau}^T\boldsymbol{\tau}$.

To summarise, we have implemented and compared four controllers:

1. Classical Kinematic Control:

$$\boldsymbol{\tau} = \mathbf{J}^T(\mathbf{q})[\mathbf{K}_P(\mathbf{y}_d - \mathbf{y})] - \mathbf{K}_D\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) \,. \tag{4.10}$$

2. Classical Inverse Dynamics Control:

$$\boldsymbol{\tau} = \mathbf{M}\mathbf{J}^{\dagger}(\ddot{\mathbf{y}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{h} \,. \tag{4.11}$$

3. Projected Inverse Dynamics Control:

$$\boldsymbol{\tau} = \mathbf{P}(\mathbf{M}\mathbf{J}^{\dagger}(\ddot{\mathbf{y}}_r - \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{h}) \,. \tag{4.12}$$

4. Projected Inverse Dynamics Optimal Control:

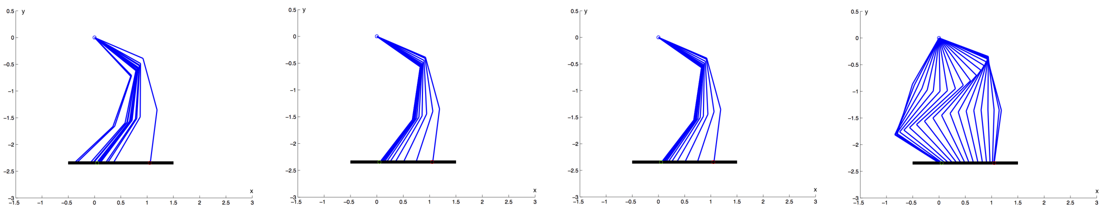
$$\frac{\partial \dot{\mathbf{q}}}{\partial t} = \mathbf{M}_C^{-1}(\mathbf{P}(\boldsymbol{\tau} - \mathbf{h}) + \mathbf{C}_C\dot{\mathbf{q}}) \,. \tag{4.13}$$

For the first three controllers we used Matlab[1]. For the fourth controller we used ACADO Toolkit [141]. Furthermore, because of numerical issues, some equality constraints have been relaxed by an ϵ, *e.g.*, the constraint on the y coordinate has been relaxed by $\epsilon = 0.001$m.

Fig. 4.8 reports the evolutions generated by each controller respectively. The evolution produced by the kinematic controller shows an overshooting behaviour, which we believe might be mitigated with a different choice of gains in the PD components. Such behaviour seems not to be present in the evolutions set by the other controllers. The

[1]MATLAB, version 8.3.0.532 (R2014a): The MathWorks Inc., 2014

gains were set to $K_P = 200$ and $K_D = 50$ in order to have the robot reach the goal configuration in the time limit, in other words to have the transient ending within the time interval. Importantly, note that the robot performs the same evolutions with both classical and projected inverse dynamics control. This is remarkable since the torques requested by the second controller are much lower (Fig. 4.10). To this extent, Fig. 4.9 reports the sum of squared torques commanded to the joints by each controller. The kinematic controller generates the highest torques. As expected, the commanded torques generated by the controller based on projected inverse dynamics are consistently lower than the ones generated by the classical inverse dynamics controller. Furthermore, the commanded torques generated with optimal control are the lowest over the whole time interval. These results suggest that optimal control, minimising the torques over a time horizon, successfully reduces torques with respect to the other controllers.



(a) Robot evolution - Kinematic Controller

(b) Robot evolution - Classical Inverse Dynamics Controller

(c) Robot evolution - Projected Inverse Dynamics Controller

(d) Robot evolution - Optimal Control

FIGURE 4.8: Evolution of the robot position on the workspace plane over time.

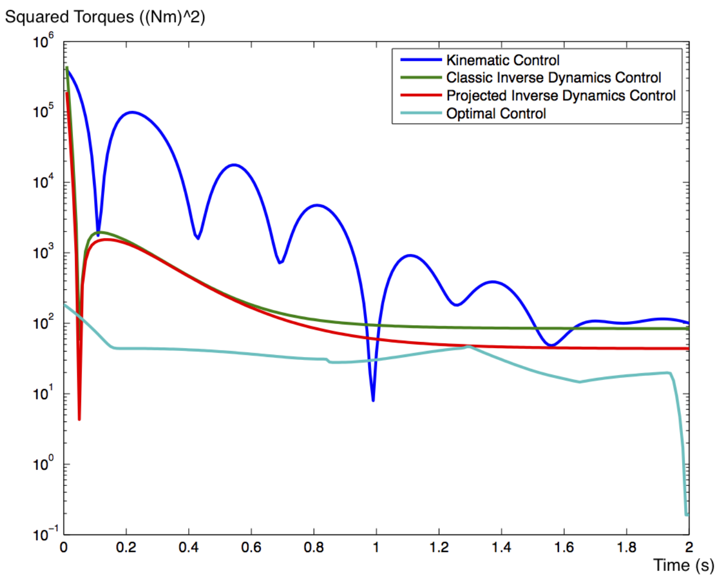


FIGURE 4.9: Torque comparison over time. This figure reports the sum of squared torques commanded to the joints by each controller. The commanded torques generated by the controller based on projected inverse dynamics are generally lower than the ones generated by the kinematic controller and classical inverse dynamics controller, while the commanded torques generated with optimal control are the lowest consistently over the entire time interval. The plot is semilogarithmic.

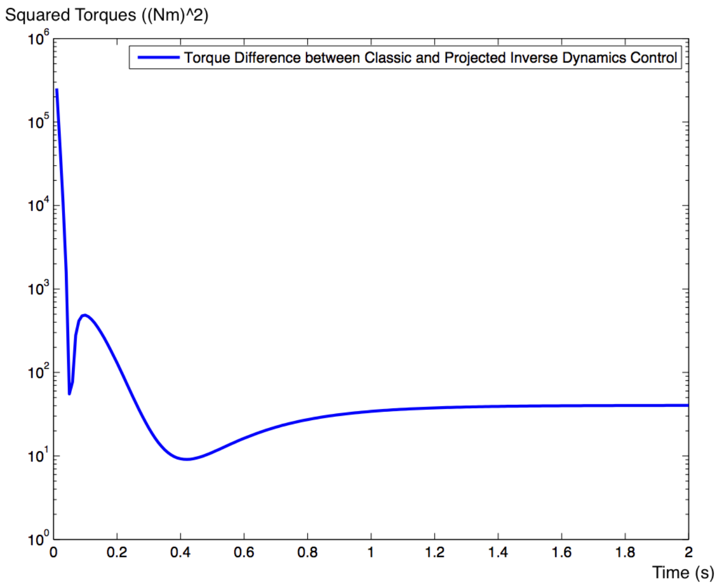


FIGURE 4.10: Torque difference between classical and projected inverse dynamics control, over time. This graph is obtained by subtracting the torques commanded by projected inverse dynamics control from the torques commanded by classical inverse dynamics control. Although the robot shows the same evolution (in terms of position), the torques commanded by projected inverse dynamics control are consistently lower than the ones commanded by classical inverse dynamics control. The plot is semilogarithmic.

4.2.2.2 Comparison of two methods for position control with a specified contact force task

It is interesting to extend our analysis, to examine how the behaviour of the controllers changes when specifying a desired active force exerted by the robot on the contact point, in addition to specifying a positional task in terms of a desired path. This type of control problem is most studied in the hybrid control literature, but here we focus on the control formulation with projected dynamics. The first part of this Subsection did not consider explicit control of the interaction force, but it is a feature that we have to take into account to achieve a good control quality in many real tasks, *e.g.*, cutting and grinding operations, or "scabbling" operations in nuclear decommissioning. Because of a cross-coupling effect [8], the interactive force is dependant also on the motion control part of the projected dynamics framework, specifically through a cross-coupling factor $\boldsymbol{\mu} = \mathbf{M}\mathbf{M}_C^{-1}$. As [8] reads, this cross-coupling vanishes if $\forall \mathbf{v} \in \mathcal{N}(\mathbf{J}_C) \ : \ \mathbf{M}\mathbf{v} \in \mathcal{N}(\mathbf{J}_C)$. However, this is not the case in our problem, so we extended our controllers to impose a desired interactive force, and in particular we set this value to $\lambda = 5$N. Specifically, we take the projected inverse dynamics control framework and extend it to control force as well, thus using the $(\mathbf{I} - \mathbf{P})\boldsymbol{\tau}_a$ term in the control law. We define $\boldsymbol{\tau}_C$ as in [1], formed of three components: one to compensate dynamics, one to compensate the cross-coupling effect and finally one to impose the desired active force. This controller and the one in

[1] differs then only for the motion part of the control. In fact, if the active force control is the same here as in [1]

$$\boldsymbol{\tau}_C = (\mathbf{I} - \mathbf{P})\mathbf{h} + (\mathbf{I} - \mathbf{P})\mathbf{M}\mathbf{M}_C^{-1}(\mathbf{P}\boldsymbol{\tau}_m - \mathbf{P}\mathbf{h} + \mathbf{C}_C\dot{\mathbf{q}}) + \mathbf{J}_C^T\lambda_{des}\,, \tag{4.14}$$

the part of motion control in [1] is defined in the projected operational space.

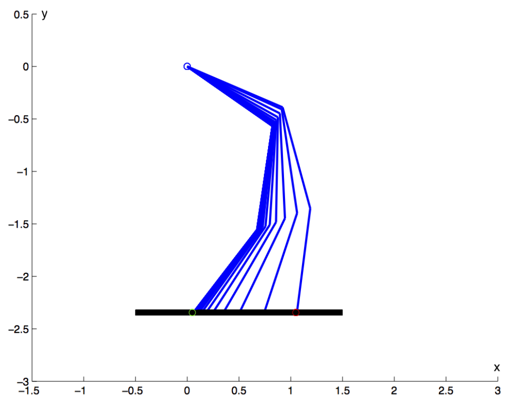
We compare the above method with a controller based on the same approach but on top of the motion control solution found with the optimal control framework, *i.e.*, $\boldsymbol{\tau}_m$ is the optimal control motion solution, and $\boldsymbol{\tau}_C$ is computed as in the other controller.

Figs. 4.11 and 4.12 summarise the results. Both controllers succeed in making the robot exert the desired active force. Moreover, the motion is identical to that of the results in the previous section (Fig. 4.8); in other words, the motion of the robot is not affected in either of the controller results. This characteristic is not surprising since $(\mathbf{I}-\mathbf{P})\boldsymbol{\tau}_C$ does not affect motion, [8]. Importantly, the controller built on top of the motion solution of optimal control generates significantly lower torques compared to the controller based purely on projected inverse dynamics without the optimal control framework. This confirms the fact that the latter method minimises the commands instant by instant, whereas the optimal control approach is based on minimising the torques over the entire time horizon of the experiment.

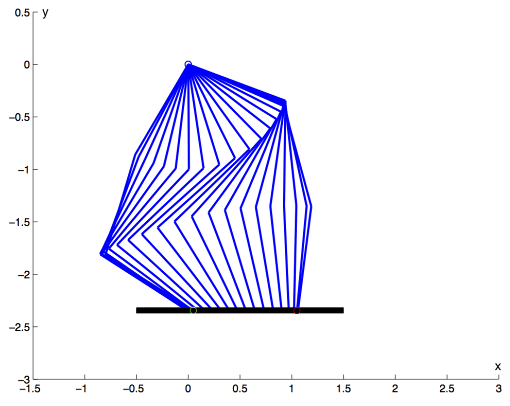
4.3 Conclusions

This Chapter represents a step towards mimicking useful human strategies of exploiting contacts. This line of research would potentially enable humanoid robots to complete everyday tasks while taking advantage of the environment, which is assumed to be known a priori in this initial study, but which could be detected and modelled online using sensors in future work.

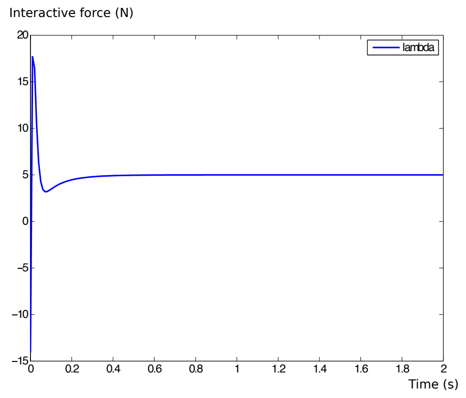
Sect. 4.1 has presented an experimental study on a robot control technique based on projected operational space dynamics. The proposed controllers are derived from the theory proposed in [5] and [6] (operational space dynamics formulation), [8] and [7] (projected operational space dynamics formulation). We have shown how these controllers can take advantage of the geometric constraints that the environment presents to the robot in terms of contacts, in order to enable lower torques to be used while performing tasks. We summarised the theory underpinning three variants of this control method, and reported the results of implementing the controllers on a KUKA LWR IV manipulator.



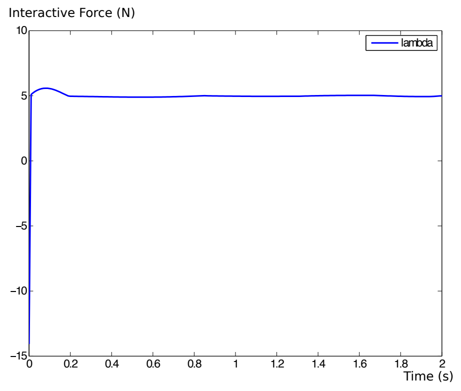
(a) Robot evolution - Projected Operational Space Dynamics Controller



(b) Robot evolution - Projected Operational Space Dynamics Controller based on Optimal Control for motion



(c) λ - Projected Operational Space Dynamics Controller



(d) λ - Projected Operational Space Dynamics Controller based on Optimal Control for motion

FIGURE 4.11: First line: evolution of the robot position on the workspace plane over time. Second line: reconstructed exerted interactive force.

Empirical experiments compare the proposed controllers against a conventional kinematic controller which sends position commands directly to the robot. Both approaches yield similar magnitudes of positional accuracy (pure position control is slightly better as expected), however the proposed controllers demonstrate significantly reduced torque requirements. A third version of the controller has also been demonstrated, which enables an additional desired end effector force to be specified. This is useful in situations where the environment provides only unilateral constraints, however this naturally requires increased joint torques. A switching mechanism to activate and deactivate such term may enhance such performance in future.

In Sect. 4.2, we chose projected inverse dynamics control and a novel optimal control method based on projected dynamics for our analyses. The first controller minimises torques instantaneously, reducing them substantially with respect to other classical controllers such as kinematic control (PD plus gravity cancellation) and classical inverse

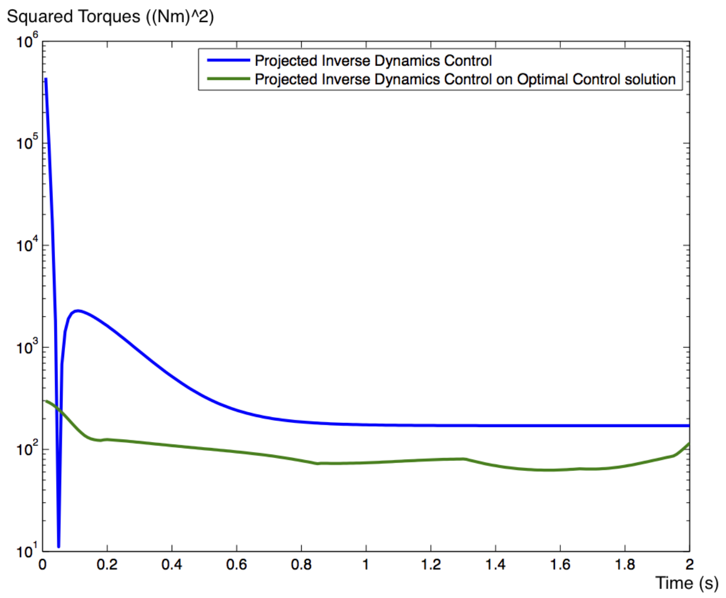


FIGURE 4.12: Torque comparison over time. This figure reports the sum of squared torques commanded to the joints by each controller. The torques computed by projected operational space dynamics controller based on optimal control for motion are dramatically lower than the torques computed by projected operational space dynamics controller. The plot is semilogarithmic.

dynamics control, as also shown in Sect. 4.1. This is due to the fact that these last two approaches control the entire motion space, even though that space is in fact limited by the contact constraint. Moreover, by means of the projector $\mathbf{P}$, it is possible to actually exploit the constraint so as to effectively reduce the commanded torques. This effect, due to $\mathbf{P}$, is an instant by instant minimisation of the torques, *i.e.*, it does not take into account the extended time interval of the control. In contrast, optimal control minimises torques over the entire time interval and thereby succeeds in reducing torques even further than the projector method. The optimal controller is in fact planning the entire trajectory over the time interval.

To support our analysis, we have reported the empirical results of experiments, in which we simulated a 3 DOF manipulator in contact with a rigid surface which constrains the robot's motion. We also explored how projected inverse dynamics control, and another controller based on optimal control for the motion component of a task, perform when some additional active contact force is desired. Our second set of empirical experiments support our analysis, that such controllers continue to accurately satisfy the positional part of the task, while the desired active contact force is also achieved by both controllers. However, the joint torques generated by the second architecture are much lower.

We are also investigating alternative methods to achieve a better exploitation of contacts. Future work will extend our optimal controller, based on projected dynamics, to also directly control the interactive force at the contact point. It will also be interesting to compare our results with an optimal controller defined with the classical formulation of dynamics. Moreover, we envision exploring a trajectory tracking problem instead of a regulation problem. Ongoing work is also developing experiments with real hardware, to implement and test these methods on a bi-manual robot platform while measuring real interaction forces with environmental contacts.

Next Chapter presents methods to estimate the kinematic constraints due to contacts through an analysis of desired velocities and observed velocities during an exploration phase. This estimation is critical to relax the assumption of complete knowledge of the constraints.

Chapter 5

Contact Estimation

Robots are increasingly being required to perform tasks which involve contacts with the environment. This Chapter addresses the problem of estimating environmental constraints on the robot's motion. We present a method which estimates such constraints, by computing the null space of a set of velocity vectors which differ from commanded velocities during contacts. We further extend this method to handle unilateral constraints, for example when the robot touches a rigid surface. Unlike the approaches described in Sect. 3.2, our method locally estimates the kinematic constraints due to contacts, without using any additional sensors (force, torque, tactile, vision) other than basic proprioception (rotation encoders at each joint). Specifically, we propose to perform a set of explorative actions and then estimate the kinematic constraints by observing the resulting motions. We present two variants, based either on a Cartesian space analysis or on a joint space analysis, and we also show how to discern unilateral constraints (*e.g.*, contact with a rigid surface which only constrains motion in one direction). We first demonstrate our method with a simulated redundant 3 DOF planar robot, and show how it can detect and estimate various kinds of constraints. Next we analyse the effect of different levels of observation noise on the accuracy with which such constraints can be estimated. Finally we show the results of experiments carried out using a KUKA LWR IV robot arm, which is tasked with estimating the environmental constraints when contacting surfaces of different inclinations.

Our proposed method would therefore be particularly useful for highly underactuated robotic arms, fingers or legs which contain passive (*e.g.*, spring-loaded) joints, provided that basic position/rotation sensing is available at each joint. Such passive/underactuated robots are attracting increasing attention from the research community, *e.g.*, [58, 59]. Additionally, in extreme environments such as nuclear decommissioning, beta and gamma radiation can destroy the delicate electronics needed for force, torque or tactile sensing. While intense gamma radiation can also destroy conventional proprioceptive rotation encoders (by causing the glass of optical encoders to become opaque), these can be replaced by electro-magnetic resolvers so that proprioception of each joint's rotation can still be reliably sensed in such environments. Our method could therefore be used to help such robots estimate the position and direction of constraint surfaces, *e.g.*, for tasks such as scabbling (already described).

Our laboratory currently lacks passively compliant, underactuated robots. Therefore, for proof of principle we have instead used an actively-compliant KUKA LWR IV robot to demonstrate our method. We only make use of this robot's proprioceptive rotation encoders at each joint, and we do not explicitly make any use of joint torque information in our experiments. However, this robot does use torque sensing internally for low level control, to achieve compliant behaviours when in contact with environmental constraint surfaces.

Our previous work [1, 2], detailed in Chapter 4, showed how contacts can actually be exploited to enable robots to perform a desired motion task more efficiently, with reduced torques and energy consumption. This work was motivated by the ways in which humans exploit contacts. In [1] and [2] we used projected dynamics to decouple a motion task from force control in the null-space of the desired motion. We demonstrated this approach by tasking the robot with wiping a whiteboard, while also resting some of its weight on the whiteboard to reduce motor torques and energy consumption at the joints. We also showed how to control a desired contact force on the board, independently of controlling motion of the end-effector in the plane of the board. However, that work relied on prior knowledge of the position and orientation of the whiteboard surface relative to the robot's coordinate frame. In contrast, this Chapter explores the problem of how to detect and estimate the contact constraints by performing exploratory motions and using proprioception.

The remainder of this Chapter is organised as follows. Sect. 5.1 describes our method to estimate constraints. Sect. 5.2 reports the results of experiments with both real and simulated redundant manipulators. Sect. 5.3 discusses the results and provides concluding remarks and suggestions for future work.

5.1 Method to Estimate the Constraints

First we propose a method to estimate kinematic constraints using exploration in the Cartesian space. Later we extend this reasoning to: explorations conceived in the joint space; and a method for handling unilateral constraints.

We consider that the contact Jacobian $\mathbf{J}_C(\mathbf{q})$ can be described as

$$\mathbf{J}_C(\mathbf{q}) = \mathbf{\Lambda}\mathbf{J}(\mathbf{q})\,, \tag{5.1}$$

where $\mathbf{\Lambda}$ is a matrix that specifies which dimension(s) in the task-space is (are) constrained due to contacts. Substituting Eq. 5.1 into Eq. 2.5, we get:

$$\mathbf{J}_C(\mathbf{q})\dot{\mathbf{q}} = \mathbf{\Lambda}\mathbf{J}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}$$

or equivalently

$$\mathbf{\Lambda}\dot{\mathbf{p}} = \mathbf{0}\,. \tag{5.2}$$

$\mathbf{\Lambda}$ is independent of the dimensionality of the configuration space of the robot and independent of the current configuration of the robot. However, the number of independent rows depends on the number of independent constraints.

In the example of Fig. 3.6, where the constraint is in the z direction of the end effector (approaching axis for the manipulator), $\mathbf{\Lambda}$ would have the form

$$\mathbf{\Lambda} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}\,, \tag{5.3}$$

such that

$$\mathbf{\Lambda}\dot{\mathbf{p}} = \mathbf{\Lambda}\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \dot{z} = 0\,, \tag{5.4}$$

i.e., the end-effector velocity along the z-axis is null.

In real-world tasks involving contacts, in general it will be non-trivial to compute $\mathbf{\Lambda}$. However, we know that $\mathbf{\Lambda}\dot{\mathbf{p}} = \mathbf{0}$ and equivalently, $\dot{\mathbf{p}}^T\mathbf{\Lambda}^T = \mathbf{0}$, from Eq. 5.2. Thus $\mathbf{\Lambda}^T$ is the solution to the homogeneous system

$$\dot{\mathbf{p}}^T\mathbf{\Lambda}^T = \mathbf{0}\,. \tag{5.5}$$

Let $\mathbf{B}_{\dot{\mathbf{p}}^T}$ be a set of observed $\dot{\mathbf{p}}^T$, *i.e.*, the end-effector velocities where the end-effector is in contact with the constrained surface, then the solution set of the homogeneous system in Eq. 5.5 can be found by computing the right null space of $\mathbf{B}_{\dot{\mathbf{p}}^T}$, using singular value decomposition

$$\mathbf{B}_{\dot{\mathbf{p}}^T} = \mathbf{U}\mathbf{S}\mathbf{V}^T\,, \tag{5.6}$$

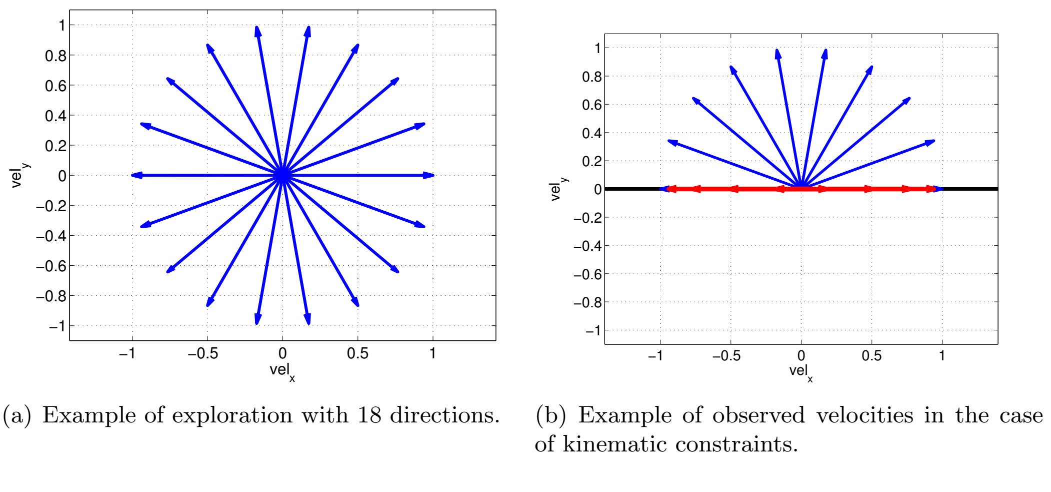
where $\mathbf{U}$ is the matrix of left singular vectors, $\mathbf{S}$ is a diagonal matrix such that $\mathbf{S}_{i,i}$ is the i^{th} largest singular value, and $\mathbf{V}$ is a matrix of the right singular vectors. $\mathbf{\Lambda}^T$ can then be computed by taking the columns of $\mathbf{V}$ with corresponding singular values smaller than a threshold value ϵ.

5.1.1 Exploration

To compose this $\mathbf{B}_{\dot{\mathbf{p}}^T}$, we propose to perform an exploration when the robot is in contact with the environment. When the robot is in contact, the observed task-space velocities $\dot{\mathbf{p}}^{obs}$ will be different from the expected $\dot{\mathbf{p}}^{exp}$ since the constraint restricts some subspace of the task space. Hence, there are two possibilities for the expected $\dot{\mathbf{p}}^{exp}$ and the observed $\dot{\mathbf{p}}^{obs}$:

1. in the free motion subspace, *i.e.*, when the motion is not in the direction of any contact: $\dot{\mathbf{p}}^{exp} = \dot{\mathbf{p}}^{obs}$;
2. in the constrained motion subspace, *i.e.*, when the motion is in the direction of any contact: $\dot{\mathbf{p}}^{exp} \neq \dot{\mathbf{p}}^{obs}$.

We collect a set $\dot{\mathbf{p}}^{obs}$ into $\mathbf{B}_{\dot{\mathbf{p}}^T}$ from the latter case and use the method based on Eq. 5.6 to estimate the selection matrix $\mathbf{\Lambda}$. Fig. 5.1(a) shows an example of exploration with 18 directions sampling velocities in the 2D Cartesian space. When in contact with a horizontal surface (black line), some Cartesian velocities cannot be performed. Fig. 5.1(b) shows the set $\dot{\mathbf{p}}^{exp} \neq \dot{\mathbf{p}}^{obs}$ in red, which go into $\mathbf{B}_{\dot{\mathbf{p}}^T}$, and the set of $\dot{\mathbf{p}}^{exp} = \dot{\mathbf{p}}^{obs}$ in blue.



(a) Example of exploration with 18 directions. (b) Example of observed velocities in the case of kinematic constraints.

FIGURE 5.1: An example of exploration with 18 directions. Fig. 5.1(a) shows commanded velocities $\dot{\mathbf{p}}^{exp}$, and Fig. 5.1(b) shows the resulting velocities $\dot{\mathbf{p}}^{obs}$, which are observed when a horizontal surface (shown in black) constrains the robot's motion. In Fig. 5.1(b), blue vectors denote observed velocities which match the expected velocities, while red vectors denote observed velocities which are different from the expected velocities. In this case $\mathbf{B}_{\dot{\mathbf{p}}^T}$ is composed from the red vectors.

5.1.2 Unilateral Constraints

Projected dynamics [8] typically assume bilateral constraints, *i.e.*, motion is blocked in both positive and negative directions of the constraint vector. However unilateral constraints are very common in real applications, *i.e.*, constraints that limit motion in only one of these directions. For example, the constraint in Fig. 3.6 limits the motion of the end effector into the table, but does not constrain motion away from the table. Aghili and Su [42] recently proposed an extension to projected dynamics to handle unilateral constraints and friction. Here, we also extend our formulation to unilateral constraints. We model unilateral constraints as

$$\boldsymbol{\Lambda}\dot{\mathbf{p}} \geq \mathbf{0}\,. \tag{5.7}$$

To find $\boldsymbol{\Lambda}$ for unilateral constraints, we follow the same method as in the previous subsection, using the set of $\dot{\mathbf{p}}^{obs}$ which do not match $\dot{\mathbf{p}}^{exp}$. However, we introduce an additional check on the remaining observed motions which are equal to those commanded, *i.e.*, $\dot{\mathbf{p}}^{exp} = \dot{\mathbf{p}}^{obs}$. In particular, we have to enforce

$$\boldsymbol{\Lambda}\dot{\mathbf{p}} \geq \mathbf{0}, \quad \forall \dot{\mathbf{p}} \;\mid\; \dot{\mathbf{p}}^{exp} = \dot{\mathbf{p}}^{obs}\,. \tag{5.8}$$

If this test holds false, then the sign of $\boldsymbol{\Lambda}$ is changed. This is particularly useful once the exploration phase is over and the robot can resume performing some other desired task. Specifically, for each Cartesian command $\dot{\mathbf{p}}_{cmd}$, if $\boldsymbol{\Lambda}\dot{\mathbf{p}}_{cmd} \geq \mathbf{0}$, then the command lies in the free motion subspace and can be executed as-is. On the other hand, if $\boldsymbol{\Lambda}\dot{\mathbf{p}}_{cmd} < \mathbf{0}$, then the command sent to the robot has to be projected using $\mathbf{P}(\mathbf{q})$ as computed in Eq. 2.12.

5.1.3 Exploration in the Joint Space

The previous analyses were carried out in the Cartesian space, however it is possible to derive equivalent equations in the joint space. In particular, we define the constraints as

$$\boldsymbol{\Lambda}^q(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}\,. \tag{5.9}$$

In this case $\boldsymbol{\Lambda}^q(\mathbf{q})$ can be regarded not only as a selection matrix, but also a constraint Jacobian. Moreover, $\boldsymbol{\Lambda}^q(\mathbf{q})$ is dependent on the configuration of the robot, in contrast to Eq. 5.2. Explorative movements can also be defined directly in the joint space. Similar to the Cartesian case, there are two possibilities for the expected $\dot{\mathbf{q}}^{exp}$ and the observed $\dot{\mathbf{q}}^{obs}$:

1. in the free motion subspace: $\dot{\mathbf{q}}^{exp} = \dot{\mathbf{q}}^{obs}$;
2. in the constrained motion subspace: $\dot{\mathbf{q}}^{exp} \neq \dot{\mathbf{q}}^{obs}$.

We collect a set of $\dot{\mathbf{q}}^{obs}$ from the latter case, and use singular value decomposition to estimate the selection matrix $\mathbf{\Lambda}^{q}(\mathbf{q})$.

5.2 Results

This section presents results from a number of experiments with both real and simulated robots, and demonstrates how our proposed method is able to successfully estimate a variety of constraints. First, we show examples of Cartesian space exploration, and evaluate how performance changes with various amounts of observation noise. Understanding the degree of robustness to noise is extremely important, because such observations are likely to be noisy in real applications. We next show examples of joint-space exploration, with an example task of using such exploration to detect the unilateral constraints imposed by joint limits. Later, we show another example of a robot tasked with following a circular trajectory. The robot detects a surface that blocks the trajectory, but is able to continue following the trajectory after exploring and estimating that constraint surface. Finally, we report on two experiments conducted on our bi-manual platform Boris, Fig. 3.6, using one of Boris' KUKA LWR IV arms to detect and estimate a horizontal surface and an inclined surface.

5.2.1 Cartesian exploration without observation noise

We begin with a simplified example, in order to clearly illustrate to the reader how our basic method works. We ran 10 simulations in a 2D environment, assuming frictionless interaction between the end-effector and constraining surface, and assuming perfect proprioception with zero observation noise. Each simulation corresponded to a setup similar to Fig. 5.1(b), but with the constraint surface tilted at a different angle in each of the ten trials. Table 5.1 shows that the estimated constraint directions perfectly match the true constraint directions in every trial, however sometimes the sign is reversed (experiments 6, 9 and 10). In these cases, despite the sign difference, the estimated and true constraints have identical null space, thus the estimated projector and the true projector $\mathbf{P}(\mathbf{q})$ are also identical. Information on the sign can be recovered using the test in Eq. 5.8. Nevertheless, the information about the sign is lost in $\mathbf{P}(\mathbf{q})$, due to its definition.

Exp.	Ang. Error	Real $\mathbf{\Lambda}$	Estimated $\mathbf{\Lambda}$
1	0°	$[-0.6870, -0.7267]$	$[-0.6870, -0.7267]$
2	0°	$[0.0840, -0.9965]$	$[0.0840, -0.9965]$
3	0°	$[0.5194, 0.8545]$	$[0.5194, 0.8545]$
4	0°	$[0.7787, -0.6274]$	$[0.7787, -0.6274]$
5	0°	$[0.9758, 0.2185]$	$[0.9758, 0.2185]$
6	0°	$[0.8388, 0.5444]$	$[-0.8388, -0.5444]$
7	0°	$[0.9894, -0.1454]$	$[0.9894, -0.1454]$
8	0°	$[-0.1445, 0.9895]$	$[-0.1445, 0.9895]$
9	0°	$[0.6178, 0.7864]$	$[-0.6178, -0.7864]$
10	0°	$[-0.4631, -0.8863]$	$[0.4631, 0.8863]$

Table 5.1: Results of simulations on 10 different constraint surfaces in a 2D environment. For each experiment, the table reports true and estimated $\mathbf{\Lambda}$ and the error (in degrees) of the estimated constraint surface inclination angle. Opposite signs in the estimated constraints are highlighted in blue.

5.2.2 Robustness of Cartesian exploration to observation noise

In this section we explore how the accuracy of constraint estimation degrades with increasing amounts of observation noise during proprioceptive sensing. To analyse this performance degradation, we added normally distributed noise, of various magnitudes, to the observed velocity vectors used for constraint estimation. We conducted 100 experiments for each level of noise, with nine different (progressively larger) normally distributed noise levels, ranging from $\sigma = 0.1$ up to $\sigma = 0.9$ in terms of observed velocity noise magnitude (as compared to true/commanded velocity magnitude). In all experiments, the simulated 2D robot was tasked with exploring the same 2D constraint vector $[0, 1]$.

Fig. 5.2 plots how constraint estimation errors (and error spread) increase with respect to increases in proprioceptive noise magnitude. These results suggest that, given a fixed number of exploratory movements, constraint estimation errors increase linearly with proprioceptive noise magnitude. However note that, even with very high levels of noise, a correspondingly high number of exploratory robot movements should still be able to recover an accurate estimate of the constraint surface. Additional experiments, to explore how errors decrease with increased amounts of exploration, will be a subject of ongoing and future research.

5.2.3 Joint-space exploration to detect joint limits

As well as the Cartesian task-space method evaluated in Section 5.2.2 above, Section 5.1.3 showed how exploration and estimation of constraints can also be performed in the configuration (joint angle) space of the robot. Unlike the Cartesian approach, the

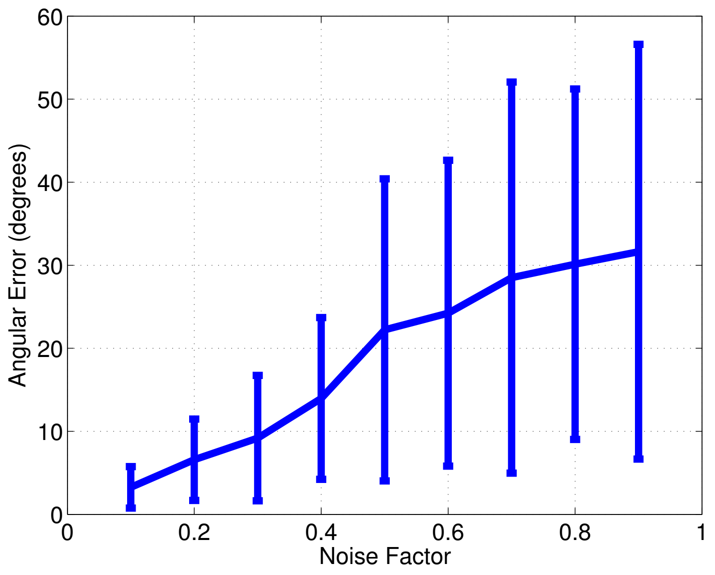


FIGURE 5.2: Plot showing how errors in the estimation of the constraint surface (angle of surface inclination) increase with magnitude of observation noise (erroneous observed velocity vectors). Results suggest that constraint estimation errors increase linearly with observation errors, as expected.

joint space approach estimates a configuration-dependent constraint, and is therefore particularly well suited to explorative estimation of the robot's joint limits. Fig. 5.3 illustrates a simulation experiment, in which a 3DOF planar arm is tasked with performing a circular trajectory. When a joint limit is hit (detected by observed joint angles deviating from commanded angles), an exploration in the joint space is triggered and our approach successfully understands which joint has hit its limit. In this example, our method returns an estimated constraint in the form $[1, 0, 0]$, indicating that joint 1 has hit its upper bound.

5.2.4 Adapting a task to overcome a detected constraint surface

Estimated constraints can be used to project the task motion onto the free motion subspace. After completing exploration and constraint estimation, if $\mathbf{\Lambda}\dot{\mathbf{p}}_{cmd} \leq \mathbf{0}$, velocity commands $\dot{\mathbf{q}}_{cmd}$ are projected using $\mathbf{P}(\mathbf{q})$, thus sending modified commands $\mathbf{P}(\mathbf{q})\dot{\mathbf{q}}_{cmd}$ to the robot's motors. Fig. 5.4 shows an example of such behaviour. The robot begins performing a circular trajectory task, but then collides with a horizontal surface. This constraint is then estimated through exploration. Once the constraint surface has been estimated, the robot resumes performing the commanded circular trajectory. During contact situations, the robot checks whether the commands lie in the free motion space or not. If not, such commands are projected using $\mathbf{P}(\mathbf{q})$. This modifies the trajectory to one which is as close as possible to the commanded trajectory, given the environmental constraint. Fig. 5.4 plots the resulting motion.

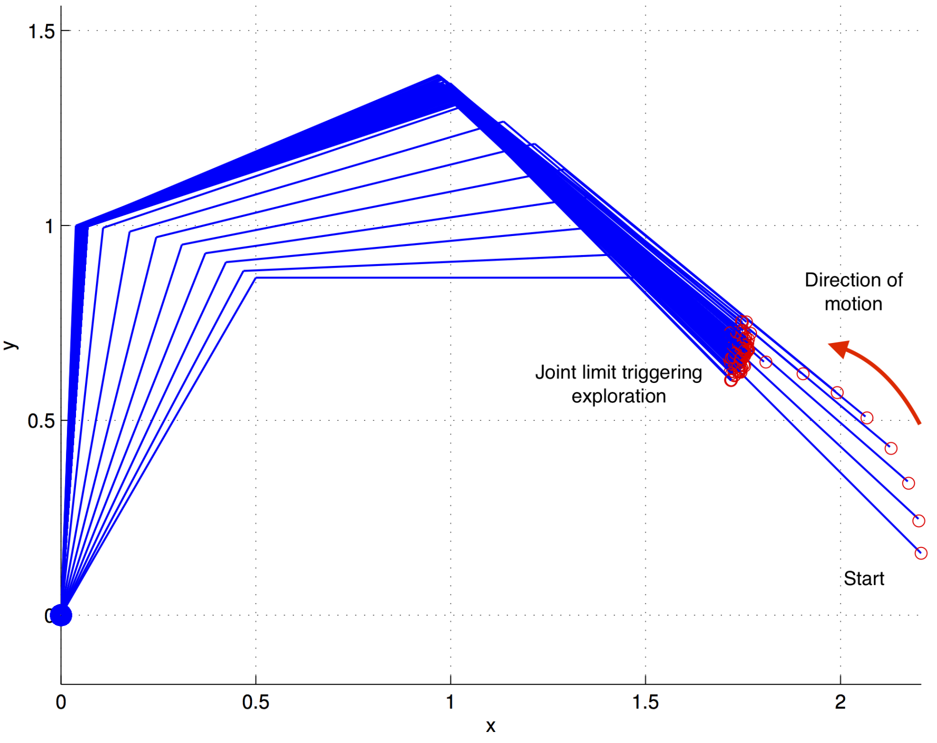


Figure 5.3: Estimation of a joint limit by performing exploratory motions in the joint space. The robot starts executing a circular motion task, and then a deviation from the commanded trajectory is detected when the robot hits a limit on its base joint. The robot then quits the commanded task, and performs a series of exploratory motions defined in the robot's joint space. Finally the constraint is correctly estimated. In this example, the estimated constraint is $\mathbf{\Lambda}^q(\mathbf{q}) = [1, 0, 0]$, which means that joint 1 has hit its upper bound.

5.2.5 Experiments with a real compliant robot arm

This section reports results of experiments using our bi-manual half-humanoid robot Boris. Boris is tasked with detecting and characterising constraints caused by a white-board, firstly when the board lies flat in the horizontal plane, Fig 3.6, and secondly when it is inclined at an angle of 47 degrees with respect to the horizontal, Fig 5.5. In both cases, the whiteboard and the materials that support it are significantly flexible. This flexibility is unmodelled, thereby presenting significant noise in the observed end-effector velocities. In particular, this flexibility allows a significant amount of perpendicular end-effector motion into the board surface.

In each experiment, Boris performed 64 exploratory motions in the 3D space. Each exploratory motion comprised a 5 cm movement of the end-effector, *i.e.*, the sphere of exploration had a 5 cm radius, and the azimuthal and altitudinal angles of rotation (wrt the tool-space) were sampled 8 times each. A contact situation was detected whenever observed motions differed from commanded motions, and relative end-effector velocity

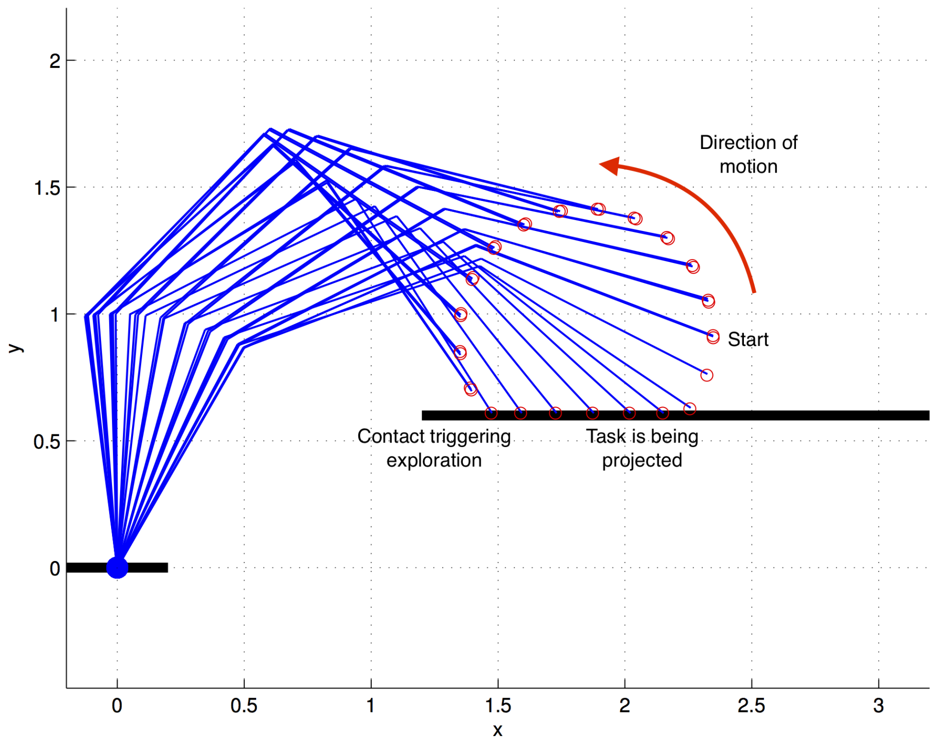


Figure 5.4: Example of adapting a commanded trajectory to overcome detected constraints. The robot modifies a commanded circular trajectory, by projecting commanded velocities via $\mathbf{P}(\mathbf{q})$, to comply as closely as possible with the commanded trajectory, while respecting the detected horizontal surface.

vectors were selected for the estimation. Note, for proof of principle the computation of surfaces from the selected observations was performed offline.

For each surface, we performed 10 experiments (each comprising 64 exploratory motions). The surface inclination angles were estimated with mean errors of 2.4484° ($\sigma = 1.0921°$) for the horizontal surface, and 4.1772° ($\sigma = 2.0684°$) for the 47° inclined surface. For the horizontal and inclined surfaces respectively: Tables 5.2 and 5.3 report angular errors and estimated $\boldsymbol{\Lambda}$ for all 10 experiments on each surface; Figs. 5.6 and 5.8 illustrate the estimated $\boldsymbol{\Lambda}$ vectors for all ten trials as red arrows, and the true $\boldsymbol{\Lambda}$ as a blue arrow; Fig. 5.7 and Fig. 5.9 show an example set of observed end-effector velocities for one of the ten experiments for each surface. Note that, for both experiments, the largest components of these velocity vectors lie in the planes of the contact surfaces (as expected), however these observed velocity vectors also have a small but significant component in the direction orthogonal to the contact surface. This is due to unmodelled flexibility of the whiteboard and its supporting materials, which allow the robot to move slightly in the constrained direction.

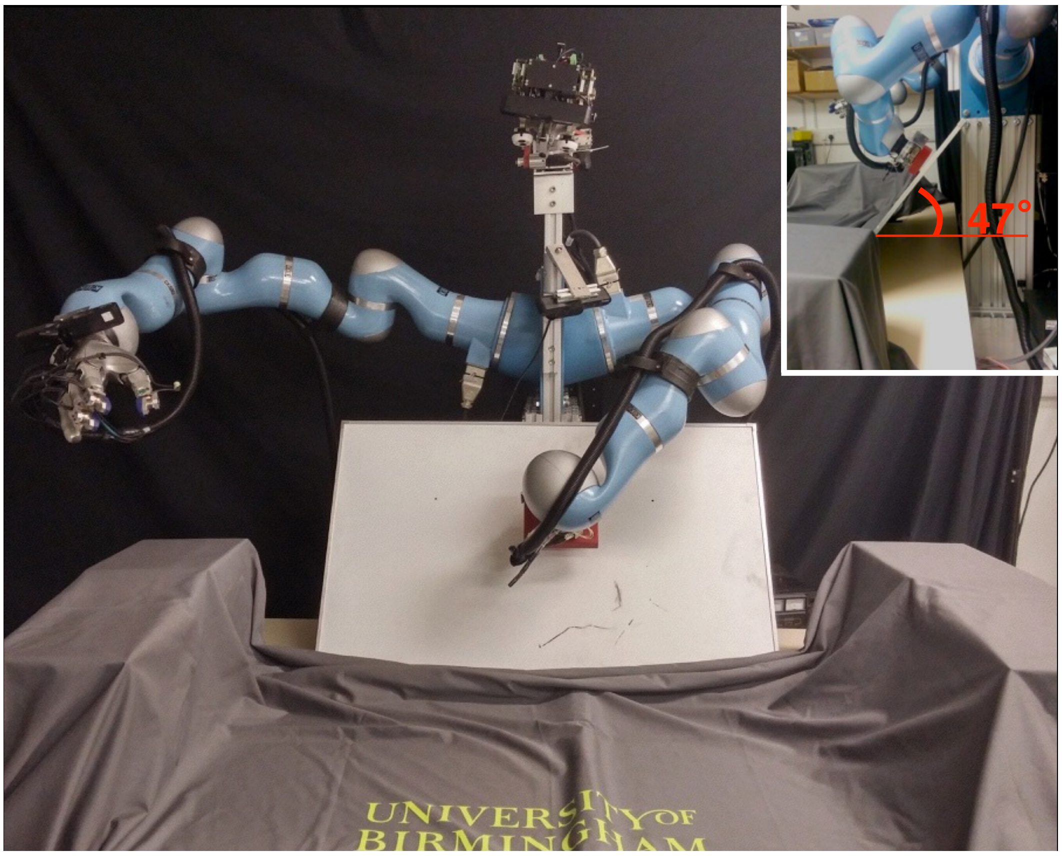


FIGURE 5.5: Boris uses proprioception to explore a whiteboard surface inclined at an angle of 47 degrees to the horizontal. Top right inset, side view showing the angle of inclination of the constraining surface.

Experiment	Angular Error	Estimated $\mathbf{\Lambda}$
1	3.0960°	$[-0.0458, 0.0286, 0.9985]$
2	2.1548°	$[-0.0349, 0.0140, 0.9993]$
3	2.6676°	$[-0.0464, 0.0031, 0.9989]$
4	2.8165°	$[-0.0485, -0.0081, 0.9988]$
5	2.8139°	$[-0.0487, 0.0062, 0.9988]$
6	2.6441°	$[0.0453, 0.0087, -0.9989]$
7	0.8245°	$[-0.0117, -0.0084, 0.9999]$
8	1.6986°	$[-0.0296, -0.0005, 0.9996]$
9	1.1103°	$[-0.0112, -0.0158, 0.9998]$
10	4.6582°	$[-0.0795, 0.0168, 0.9967]$

TABLE 5.2: Estimated $\mathbf{\Lambda}$ and error in estimated surface inclination angle (in degrees) for a horizontal contact surface with true $\mathbf{\Lambda} = [0, 0, 1]$.

5.2.6 Discussion of results

We deliberately designed the experiments with the real robot to test our method in non-ideal conditions, *i.e.*, a flexible, deforming contact surface. The results suggest that our method performs robustly in these circumstances, with mean errors of 2.4° for the horizontal surface, and 4.2° for the inclined surface. Note that, due to the highly

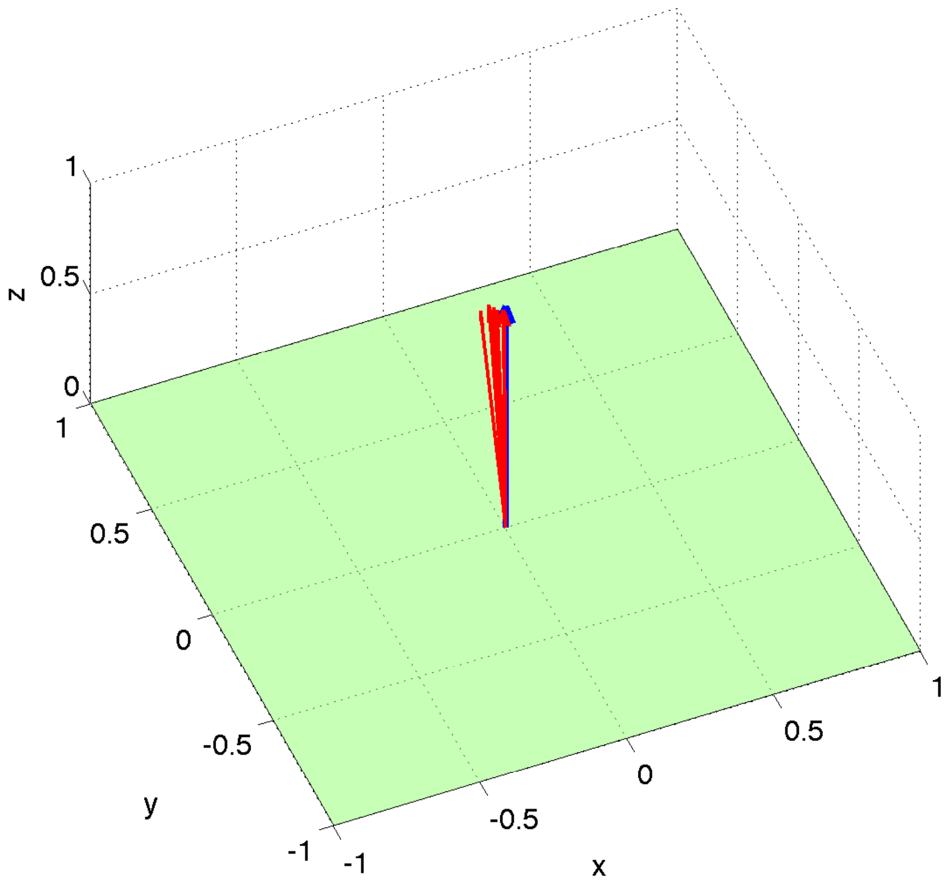


FIGURE 5.6: Visualisation of results for the horizontal contact surface. Blue arrow represents the true $\mathbf{\Lambda}$ vector, and the green plane is the contact surface. Estimated $\mathbf{\Lambda}$, for each of ten trials, are represented by red arrows.

deformable supporting structure and materials, as well as flexibility of the whiteboard itself, the true angles of inclination of these surfaces would not have been the same as those measured prior to the robot making contact (horizontal in the first experiment, and inclined at 47° in the second experiment). It is therefore possible that the estimation errors with respect to the real surfaces, deformed by the robot pushing on them during the experiments, may actually be substantially less than the figures reported above.

In comparison, the work in [73] reports smaller errors of less than 1° in the angles of inclination estimated by their quadrupedal trotting robot. However: their robot trots on a completely rigid flat surface, as compared to our flexible and deformable whiteboard structure; and they make use of optical force sensors on all four feet to detect contacts, which are not available in our case. Additionally note that the work in [73] exploits multiple contacts on the surface. In contrast, our method can work with as little as one contact trajectory in the 2D case (similar to the trotting robot's surface that has only a pitch angle with respect to the ground plane, with no roll or yaw angles). In the case of a truly 3D surface of arbitrary orientation, our method can work with as little as two (non-parallel) observed contact trajectories.

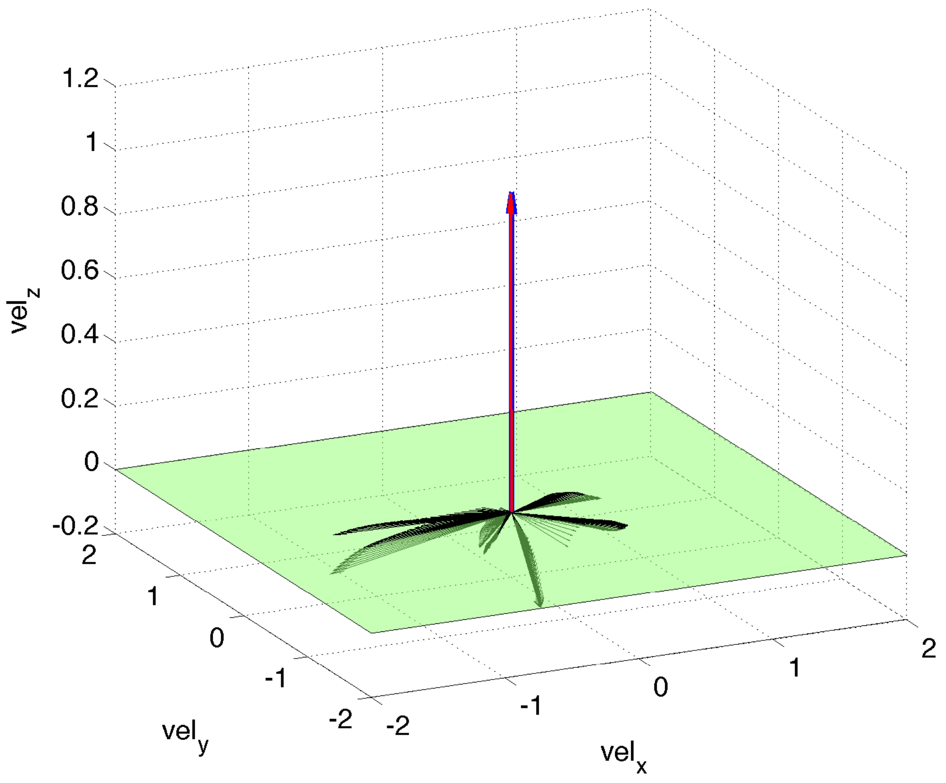


FIGURE 5.7: Observed velocity vectors (black arrows), estimated $\mathbf{\Lambda}$ (red arrow), and true $\mathbf{\Lambda}$ (blue arrow) from experiment 7 on the horizontal contact surface. Note that some velocity vectors have components in the negative z direction, due to the flexibility of the whiteboard which allows the end effector of the robot to move slightly in the negative z direction.

Experiment	Angular Error	Estimated $\mathbf{\Lambda}$
1	3.1708°	$[-0.0509, -0.7155, 0.6968]$
2	5.8918°	$[0.0726, -0.7770, 0.6253]$
3	5.5162°	$[-0.0941, -0.7413, 0.6645]$
4	5.3110°	$[0.0441, -0.7837, 0.6196]$
5	6.4421°	$[0.0839, -0.7776, 0.6232]$
6	5.9987°	$[-0.0986, -0.7038, 0.7035]$
7	4.8269°	$[0.0324, -0.7817, 0.6228]$
8	2.2875°	$[-0.0129, -0.7565, 0.6538]$
9	0.5057°	$[0.0082, -0.7335, 0.6797]$
10	1.8208°	$[0.0175, 0.7490, -0.6623]$

TABLE 5.3: Estimated $\mathbf{\Lambda}$ and error in estimated surface inclination angle (in degrees) for an inclined (47°) contact surface with true $\mathbf{\Lambda} = [0, -0.7314, 0.6820]$.

The quadrupedal robot in [73] is assumed to be trotting on a uniformly flat surface with constant slope, and all four feet must be trotting on the same slope. In contrast, our method could be extended to more complex surfaces, where the constraints can be modelled as

$$\mathbf{\Lambda}(\mathbf{p})\dot{\mathbf{p}} = \mathbf{0}\,, \tag{5.10}$$

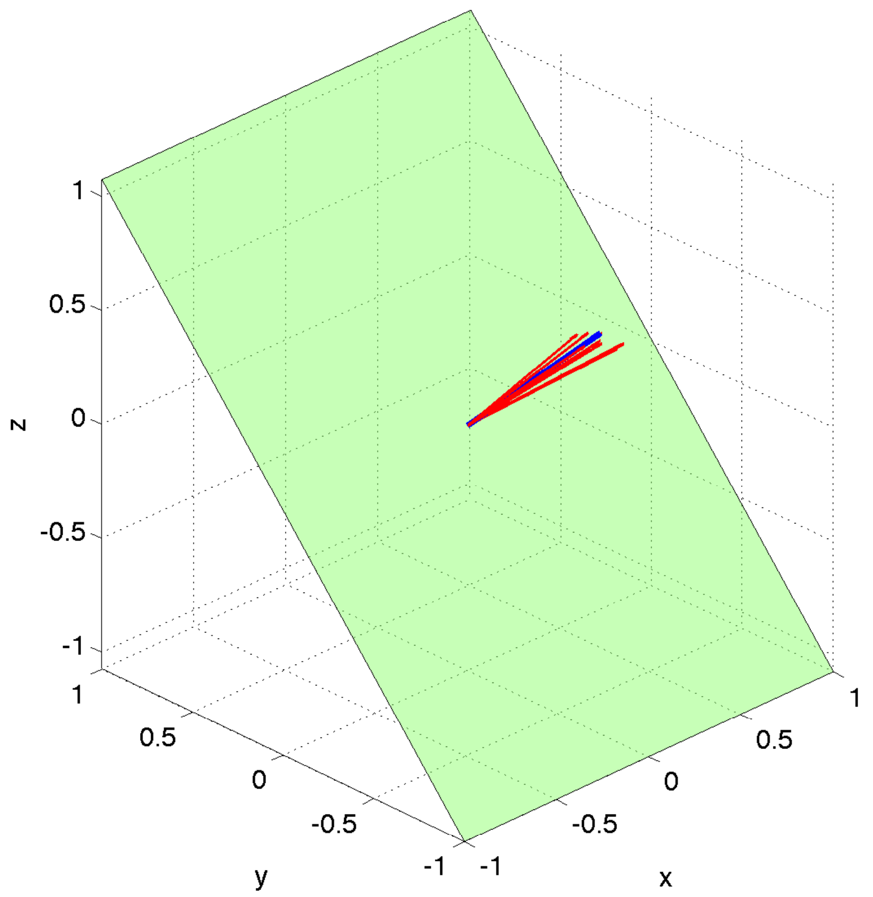


FIGURE 5.8: Visualisation of results for exploring the inclined surface. Blue arrow represents the true $\mathbf{\Lambda}$ vector, and the green plane is the contact surface. Estimated $\mathbf{\Lambda}$, for each of ten trials, are represented by red arrows.

where the dependency of the constraints on the position in space is explicit. In such case, $\mathbf{\Lambda}(\mathbf{p}_i)$ could be estimated by performing a local exploration around the point $\mathbf{p}_i$.

For the planar surfaces addressed in this Chapter, we chose the vector of the $\mathbf{V}$ matrix associated to the smallest singular value. However, if each contact provides constraints limiting motion in more than one direction, *i.e.*, contact with a non-planar surface, this can be detected by the fact that multiple singular values, in the decomposition of the set of observed velocities, will be very close to 0.

Finally, our method is not limited to end-effector contacts only. In principle, our method could be extended to include multiple contacts on different parts of the robot body, having one $\mathbf{\Lambda}$ per contact and devising explorative motions for different parts of the robot, *e.g.*, end effector and elbow.

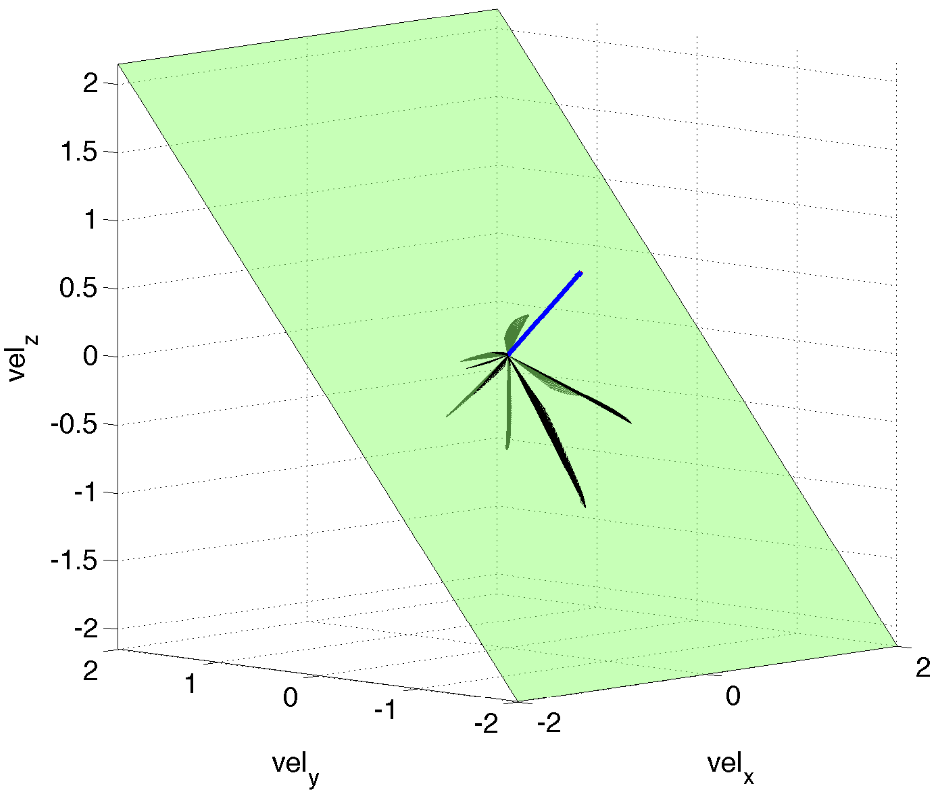


FIGURE 5.9: Observed velocity vectors (black arrows), estimated $\boldsymbol{\Lambda}$ (red arrow), and true $\boldsymbol{\Lambda}$ (blue arrow) from experiment 9 on the inclined contact surface. Note that some velocity vectors have components in the perpendicular direction penetrating the whiteboard surface. This is due to the flexibility of the whiteboard which allows the end effector of the robot to move slightly in the penetrating direction into the surface.

5.3 Conclusions and suggestions for future work

This Chapter presented a method to estimate kinematic constraints due to contacts, without the need for force-torque or tactile sensors at the contact points, or other sensing modalities such as vision.

Differently from approaches such as in [142], which use learning and force readings, our method is based only on kinematics, and works by computing the null space of a set of observed velocity vectors which differ from commanded velocities during explorative motions. Additionally, we showed how to extend this method to handle unilateral constraints, and showed how an equivalent formulation, defined in the joint space, is convenient for applications such as explorative estimation of a robot's joint limits.

We demonstrated the effectiveness of our method in a number of simulations, where we also explored how the accuracy of constraint estimations are affected by noisy observations of velocity vectors. We also conducted two sets of experiments with our bi-manual half-humanoid robot Boris, showing how it is possible to reliably detect and estimate parameters for constraining contact surfaces positioned at different angles of inclination with respect to horizontal. The constraints were successfully estimated, with mean errors of 2.4° for the horizontal surface, and 4.2° for the inclined surface, even though the surface itself was flexible and deformed significantly during contacts.

Future work will address the problem of defining quicker and smarter sets of explorative actions, where we will also reduce the extent of space that needs to be covered during exploration. We will also implement and demonstrate extensions of this method to handle non-planar surfaces. The questions of how to reason about, parameterise, characterise, and explore highly deformable or soft surfaces, remain open research problems.

Chapter 6

Vision-based State Estimation

This Chapter presents a vision-based approach for estimating the configuration of, and providing control signals for, an under-sensored robot manipulator using a single monocular camera. Some remote manipulators, used for decommissioning tasks in the nuclear industry, lack proprioceptive sensors because electronics are vulnerable to radiation. Additionally, even if proprioceptive joint sensors could be retrofitted, such heavy-duty manipulators are often deployed on mobile vehicle platforms, which are significantly and erratically perturbed when powerful hydraulic drilling or cutting tools are deployed at the end-effector. In these scenarios, it would be beneficial to use external sensory information, *e.g.*, vision, for estimating the robot configuration with respect to the scene or task. Conventional visual servoing methods typically rely on joint encoder values for controlling the robot. In contrast, our framework assumes that no joint encoders are available, and estimates the robot configuration by visually tracking several parts of the robot, and then enforcing equality between a set of transformation matrices which relate the frames of the camera, world and tracked robot parts. To accomplish this, we propose two alternative methods based on optimisation. We evaluate the performance of our developed framework by visually tracking the pose of a conventional robot arm, where the joint encoders are used to provide ground-truth for evaluating the precision of the vision system. Additionally, we evaluate the precision with which visual feedback can be used to control the robot's end-effector to follow a desired trajectory.

This Chapter shows how reliable feedback of robot configurations can be achieved by tracking several parts of the robot in monocular camera images. A framework is presented whose main component estimates the robot configuration by enforcing the equality between a set of transformation matrices relating frames set in the camera, world and the tracked robot parts. For this purpose, a model-based vision approach is used, derived from virtual visual servoing (VVS), in order to track various parts of the robot

[99]. The state of the robot, *i.e.*, the joint configuration, is estimated by combining this tracked information with the robot's kinematic model. In the following the terms state estimation and configuration estimation are used interchangeably. To solve this estimation problem, two different types of non-linear optimisation schemes are presented and compared. In addition to estimating the robot state, it is also shown how these estimations can be successfully used as quantitative feedback to a classical kinematic controller, in order to make the robot achieve a desired end-effector position.

Robots in research labs generally have joint encoders, which is the reason why the larger part of previous robotics literature including visual servoing literature, assumes full knowledge of joint angles. In contrast, we are of the opinion that our approach of estimating the robot configuration by using only monocular camera images, represents both novelty and considerable usefulness for robotics applications in harsh environments. Additionally, the methods proposed in this Chapter may have broader applications to other problems, *e.g.*: human-robot or robot-robot interaction; articulated robot calibration; use of a remote camera for servoing of mobile manipulator platforms with respect to surrounding objects.

The remainder of the Chapter is organised as follows. Sec. 6.1 describes the proposed vision-based configuration estimation scheme along with the details of each component. Sec. 6.2 reports the results of experiments to i) measure the accuracy of our vision-based state estimates, and ii) to measure the precision with which our vision-based approach can be used to control a robot to move its end-effector to a desired position. Sec. 6.3 provides concluding remarks and suggests directions for future work.

6.1 State estimation and control framework

The main goal of this work is to provide reliable quantitative configuration feedback for under-sensored robots, so neither the particular choice of visual tracking algorithm nor the particular choice of visual servoing controller form the primary focus of our contribution. This motivates our choice of architecture, which is composed of three separate components: visual tracking of parts of the robot; state estimation; and a controller. We concentrate on state estimation, and choose available methods for the other components. The modularity of this architecture enables flexibility by allowing modification of each of these components independently.

As stated earlier, our framework is composed of three main parts. The first component is visual tracking of individual robot links. The model-based visual tracker adopted in this Chapter is given the 3D models of a small number of selected robot parts, tracks the

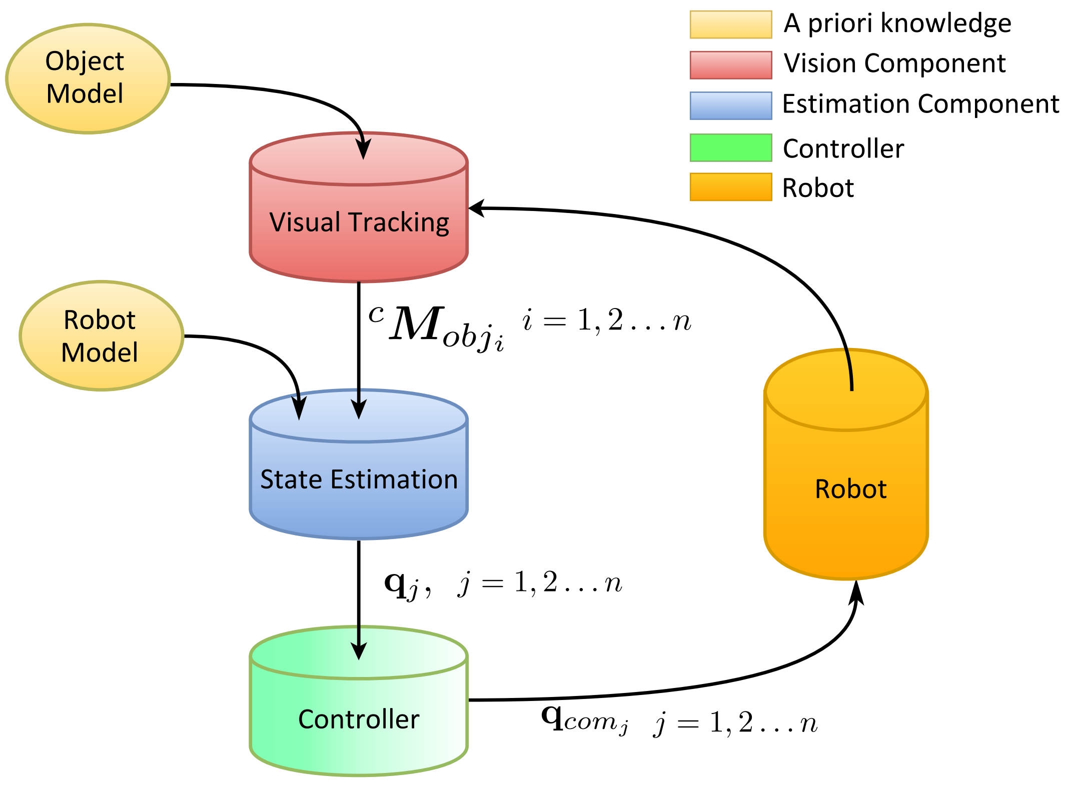


FIGURE 6.1: Overview of our modular pose estimation architecture. The visual tracking module uses RGB images provided by a camera, together with the models of a few selected robot parts, and returns the pose of those parts with respect to the camera frame. The part poses are then used by the state estimation module along with the robot kinematic model, which returns the robot joint configuration. Finally, the controller utilises the overall estimated state to servo the robot.

corresponding poses of these parts and returns the homogeneous transformation matrices between the camera and the tracked objects, $^{C}\mathbf{M}_{obj_i}$. Previously, such methods were used for virtual reality [143] and part assembling [144]. The second component makes use of these matrices in estimating the robot's state, *i.e.*, the joint configuration $\mathbf{q}$. We propose two alternative methods to accomplish this task, both based on optimisation. Finally, we implement a classical kinematic controller to show how these estimations can be used as feedback in a closed-loop control scheme. As previously discussed, this choice of architecture is motivated by the intention of being as modular as possible, *i.e.*, the proposed state estimation method can be easily replaced by another one, with no major modifications. The same applies to the visual part tracking module and the controller module. Fig. 6.1 illustrates the architecture of our proposed framework.

6.1.1 Visual Tracking

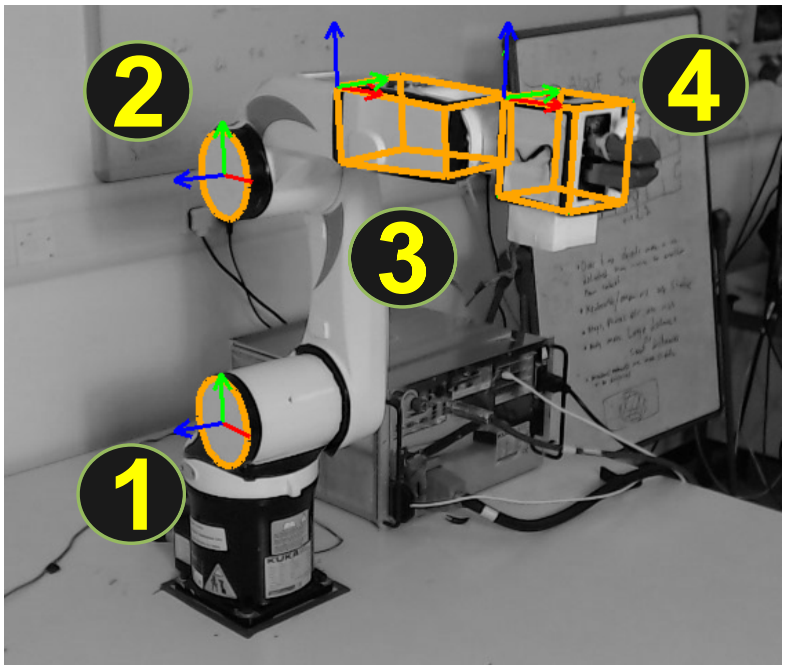
In this work, visual tracking of various parts of the robot has been accomplished using a model-based tracker available in ViSP [145], which projects CAD models of the parts onto camera images. Real-time tracking and pose estimation is achieved by using a Virtual Visual Servoing (VVS) framework [99]. Previous results [143, 144] suggest that such trackers are robust to lighting intensity variations and partial occlusions. Furthermore, this approach runs in real-time on an ordinary CPU without needing GPU acceleration.

Tracking 3D models of robot parts in images is related to the classical pose estimation problem. The underlying idea is to obtain a camera pose for which a projection of the 3D model best fits with the 2D image contours of the robot part. This process involves estimating a rigid transformation between the camera frame and the tracked object frame, ${}^{C}\mathbf{M}_{obj_i}$. The key steps include: projecting the model using an initial pose estimate (typically the pose estimated at the previous frame), perform a 1D search along the model edges to update the pose, and propagate the updated pose to the next frame. In general, the pose matrix ${}^{C}\mathbf{M}_{obj_i}$ links the 3D object features $\mathbf{P}$ in the world frame to their corresponding projections $\mathbf{p}$ in the image. Assuming the camera intrinsic parameters $\mathbf{K}$ are known, this relationship is given by

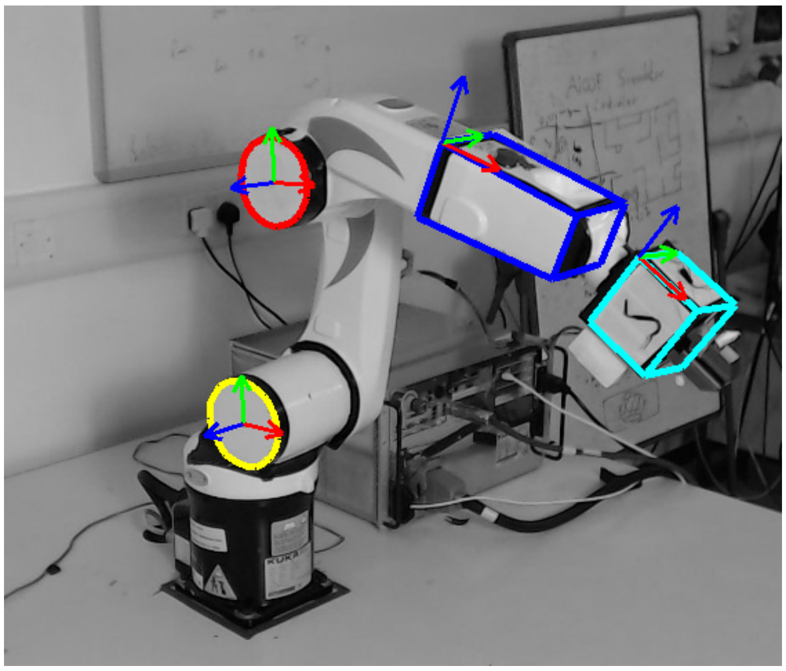
$$\mathbf{p} = \mathbf{K}{}^{C}\mathbf{M}_{obj_i}\mathbf{P}\,. \tag{6.1}$$

Next, it is possible to estimate the transformation parameters by minimising the error $\boldsymbol{\Delta}$ between the current values $\mathbf{s}(\mathbf{r})$, obtained by forward projection of the robot part model using the pose $\mathbf{r}$, and the edges $\mathbf{s}^*$ detected in the image. Its minimisation then corresponds to the movement of a virtual camera (associated with the model) by updating $\mathbf{r}$. The regulation of $\boldsymbol{\Delta}$ requires linking temporal variations of $\mathbf{s}(\mathbf{r})$ with the velocity screw of the virtual camera defined by pose $\mathbf{r}$. This is achieved by using an image Jacobian matrix $\mathbf{J}_s$. This algorithm is based on classical visual servoing, thus we refer the reader to [81] for further details.

In this work, for proof of principle, we tracked four different parts of a KUKA KR5sixx robot, Fig. 6.2. Trackers are initialised for each part by the user mouse-clicking on corresponding parts in the first image, while the robot is in its home position, Fig. 6.2(a). In the case of highly cluttered scenes the trackers' performance can become erratic due to redundant edges detected in the images. In order to minimise these phenomena, we use a Kalman Filter (KF) to predict and update the final pose after VVS, thus smoothing the changes and also reducing the reactiveness of the tracker. Specifically, we converted the ${}^{C}\mathbf{M}_{obj_i}$ into pose vectors, *i.e.*, tuples of six values, three for the orientation and three for the translation. These values are regarded as the states in the KF. We treat the pose from the estimated ${}^{C}\mathbf{M}_{obj_i}$ as noisy measurements and update the states, consequently filtering brisk changes. Finally, these updated ${}^{C}\mathbf{M}_{obj_i}$ from KF are supplied to the next module of our architecture, *i.e.*, state estimation.



(a)



(b)

FIGURE 6.2: Illustration of model-based tracking, using a KUKA KR5sixx robot for proof of principle. (a) Automatic initialised poses in the first frame. Numbers in circles represent the order of the parts selected for this work. (b) Tracked parts in a later frame.

6.1.2 State Estimation

In the following, the standard convention for symbols associated with the kinematics of the robot is observed, *e.g.*, we define $\mathbf{q}$ and $\dot{\mathbf{q}}$ as the configuration and the velocity respectively of the robot in joint space.

For state estimation, we use the following key idea. As shown in Fig. 6.3, there are two paths from the camera reference frame ${}^{C}RF$ (in yellow) to each tracked part frame ${}^{obj_i}RF$ (in red). As stated, we track four different parts of the robot, *i.e.*, $i = 1 \dots 4$. These two paths kinematically coincide, thus we enforce the following equalities to estimate the state

$$ {}^{C}\mathbf{M}_{obj_i} = {}^{C}\mathbf{T}_0 \; {}^{0}\mathbf{T}_{obj_i}(\mathbf{q}) \,, \tag{6.2}$$

where ${}^{C}\mathbf{T}_0$ is the transformation from camera to world frame and ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$ represents the transformation from world to object i frame parametrised over the joint values $\mathbf{q}$, *i.e.*, ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$ embeds the kinematic model of the robot. Specifically, for each tracked robot part, we get

$$ {}^{C}\mathbf{M}_{obj_1} = {}^{C}\mathbf{T}_0 \; {}^{0}\mathbf{T}_1(q_1) {}^{1}\mathbf{T}_{obj_1} \,, \tag{6.3}$$
$$ {}^{C}\mathbf{M}_{obj_2} = {}^{C}\mathbf{T}_0 \; {}^{0}\mathbf{T}_1(q_1) {}^{1}\mathbf{T}_2(q_2) {}^{2}\mathbf{T}_{obj_2} \,, \tag{6.4}$$
$$ {}^{C}\mathbf{M}_{obj_3} = {}^{C}\mathbf{T}_0 \; {}^{0}\mathbf{T}_1(q_1) {}^{1}\mathbf{T}_2(q_2) {}^{2}\mathbf{T}_3(q_3) {}^{3}\mathbf{T}_4(q_4) {}^{4}\mathbf{T}_{obj_3} \,, \tag{6.5}$$
$$ {}^{C}\mathbf{M}_{obj_4} = {}^{C}\mathbf{T}_0 \; {}^{0}\mathbf{T}_1(q_1) {}^{1}\mathbf{T}_2(q_2) {}^{2}\mathbf{T}_3(q_3) {}^{3}\mathbf{T}_4(q_4) {}^{4}\mathbf{T}_5(q_5) {}^{5}\mathbf{T}_6(q_6) {}^{6}\mathbf{T}_{obj_4} \,. \tag{6.6}$$

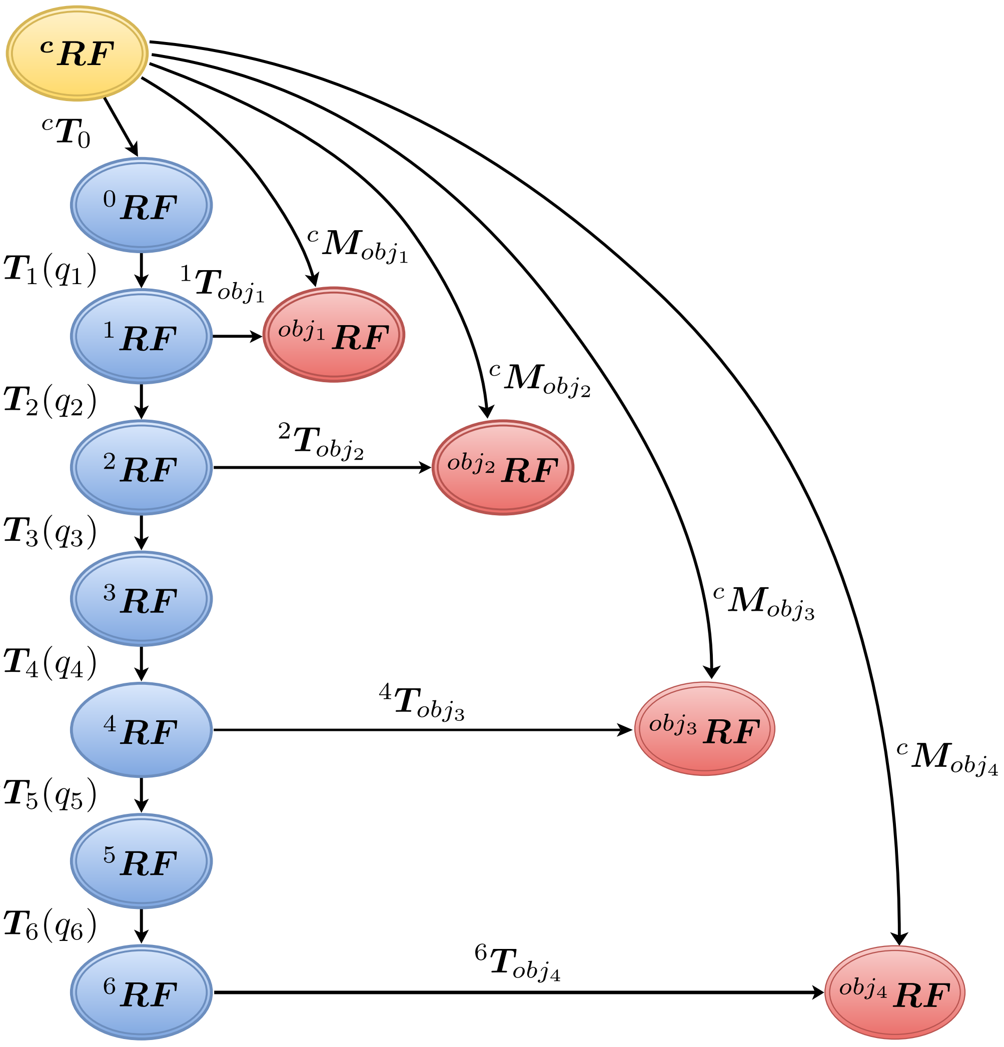
The state of the robot is now estimated by imposing the equality given in Eq. 6.2, and casting it as an optimisation problem. As already mentioned, we assume that we know the initial configuration of the robot (occupying its home position in the first image) and its kinematic model. The robot's initial configuration is used as a seed for the first iteration of the optimisation problem and the kinematic model is used to compute ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$. The optimisation problem is then stated as

$$ \begin{aligned} \underset{\mathbf{q}}{\text{minimise}} \quad & \sum_i \mathbf{e}_i(\mathbf{q}) \\ \text{subject to} \quad & |q_j| \leq q_{max} \,, \end{aligned} \tag{6.7}$$

where

$$ \mathbf{e}_j(\mathbf{q}) = vec({}^{C}\mathbf{M}_{obj_i} - {}^{C}\mathbf{T}_0 \; {}^{0}\mathbf{T}_{obj_i}(\mathbf{q})) \tag{6.8}$$

represents an error in the difference of the two paths shown in Fig. 6.3 to define a transformation matrix from the camera frame to the tracked objects frames, and q_{max} is the joint limit for the joint. In order to compute ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$, we use the convention of



$^{c}\boldsymbol{RF}$
$^{c}\boldsymbol{T}_0$
$^{0}\boldsymbol{RF}$
$^{0}\boldsymbol{T}_1(q_1)$
$^{1}\boldsymbol{RF}$
$^{1}\boldsymbol{T}_{obj_1}$
$^{obj_1}\boldsymbol{RF}$
$^{c}\boldsymbol{M}_{obj_1}$
$^{1}\boldsymbol{T}_2(q_2)$
$^{2}\boldsymbol{RF}$
$^{2}\boldsymbol{T}_{obj_2}$
$^{obj_2}\boldsymbol{RF}$
$^{c}\boldsymbol{M}_{obj_2}$
$^{2}\boldsymbol{T}_3(q_3)$
$^{3}\boldsymbol{RF}$
$^{3}\boldsymbol{T}_4(q_4)$
$^{4}\boldsymbol{RF}$
$^{4}\boldsymbol{T}_{obj_3}$
$^{obj_3}\boldsymbol{RF}$
$^{c}\boldsymbol{M}_{obj_3}$
$^{4}\boldsymbol{T}_5(q_5)$
$^{5}\boldsymbol{RF}$
$^{5}\boldsymbol{T}_6(q_6)$
$^{6}\boldsymbol{RF}$
$^{6}\boldsymbol{T}_{obj_4}$
$^{obj_4}\boldsymbol{RF}$
$^{c}\boldsymbol{M}_{obj_4}$

FIGURE 6.3: Illustration of the proposed state estimation model. Nodes represent reference frames and are classified by various colours: camera frame in yellow, robot frames in blue and tracked object frames in red. The two paths leading to each tracked object frame $^{obj_i}RF$ from the camera reference frame ^{C}RF can be seen.

Denavit-Hartenberg parameters. The overall estimation schema along with the transformation matrices between each reference frame are shown in Fig. 6.3. From Fig. 6.3, the following dependencies can be observed for each tracked object: 1. first object's position $^{obj_1}RF$ depends only on q_1; 2. second object's position $^{obj_2}RF$ on q_1 and q_2; 3. third object's position $^{obj_3}RF$ on q_1, q_2, q_3 and q_4; 4. finally fourth object's position $^{obj_4}RF$ on all the six joints. As shown in Fig. 6.2, in this work we track two cylindrical and two cuboid shaped parts of a KUKA KR5sixx robot for proof of principle. However, this choice is not a limitation of our work, and a variety of different parts could be chosen. Nevertheless, the parts must be chosen such that they provide sufficient information about all joints of the robot. Due to the modular architecture followed, using alternative parts for visual tracking will not affect the estimation scheme.

The trackers return a set of matrices, *i.e.*, one for each tracked part. These matrices can be used all together in one optimisation problem, and we term this the "full" method. Alternatively, the sets of equations coming from each of the four ${}^{C}\mathbf{M}_{obj_i}$ can be used in series to solve four optimisation problems, for subsets of joint variables, and we call this the "chained" method. Because the full method uses all of the tracked part poses at once, the optimisation problem is as given in Eq. 6.7, with $i = 4$ and $j = 6$. In this case, the solution is the tuple of joint angles that best fits all the equations. Alternatively, the chained method uses each object to estimate only a subset of joint values. These, in turn, are used as known parameters in the successive estimation problems. In the presented example, q_1 can be retrieved using obj_1, as in

$$\begin{aligned} \underset{q_1}{\text{minimise}} \quad & \mathbf{e}_1(q_1) \\ \text{subject to} \quad & |q_1| \leq q_{max} \end{aligned} \tag{6.9}$$

and from now on it is treated as known. q_2 is estimated using obj_2

$$\begin{aligned} \underset{q_2}{\text{minimise}} \quad & \mathbf{e}_2(q_1, q_2) \\ \text{subject to} \quad & |q_2| \leq q_{max} \,. \end{aligned} \tag{6.10}$$

In a similar fashion, q_3 and q_4 are estimated using obj_3

$$\begin{aligned} \underset{q_3, q_4}{\text{minimise}} \quad & \mathbf{e}_3(q_1, q_2, q_3, q_4) \\ \text{subject to} \quad & |q_j| \leq q_{max}, \ j = 3, 4 \,. \end{aligned} \tag{6.11}$$

And finally, obj_4 provides the equations to compute q_5 and q_6

$$\begin{aligned} \underset{q_5, q_6}{\text{minimise}} \quad & \mathbf{e}_4(\boldsymbol{q}) \\ \text{subject to} \quad & |q_j| \leq q_{max}, \ j = 5, 6 \,. \end{aligned} \tag{6.12}$$

There are a number of considerations regarding these two alternative approaches. Using only one object at a time, as in the chained method, the quality of configuration estimation becomes highly dependent on the tracking performance for each individual part. Although it induces the advantage of being robust to single part tracking failure (producing outliers that influence the estimation of only the relative subset of angles), it adds the disadvantage of propagating the possible error of already estimated angles in subsequent estimations. On the other hand, the full method overcomes this problem. However, it has to accommodate a solution for a higher number of equations. Thus, an error in any parameter will potentially lead to estimation errors on all joints, independently of the tracking performance.

6.1.3 Controller

For proof of principle, we implemented a classical kinematic controller of the form given in Eq. 6.13 to validate our methodology and also to demonstrate how the vision-derived state estimations can be used to servo the robot's end-effector to a desired point in the workspace.

$$\dot{\mathbf{q}}_{ref} = \mathbf{J}^{\dagger}(\mathbf{q})(\mathbf{K}_P\mathbf{e}) - \mathbf{K}_D\dot{\mathbf{q}} \tag{6.13}$$

Here, $\dot{\mathbf{q}}_{ref}$ is the desired/reference velocity, and $\mathbf{J}^{\dagger}(\mathbf{q})$ is the pseudo-inverse of the robot Jacobian computed using our estimated joint configuration. The pseudo-inversion is needed since the tasks are positional, thus the Jacobian $\mathbf{J}$ is 3x6. The error $\mathbf{e}$ has been defined as the difference between desired and estimated Cartesian positions of the end-effector. Note that the Cartesian positions are updated in each iteration using the direct kinematics with the estimated configuration. Finally, $\mathbf{K}_P$ and $\mathbf{K}_D$ are the proportional and derivative gain matrices. The estimation of the robot joint velocity has been computed as the difference between the present and the previous robot configuration. Since the commands sent to the robot are in position, Eq. 6.13 is integrated numerically to compute such commands

$$\mathbf{q}_{cmd} = \mathbf{q}_t + \boldsymbol{\Delta} t\ \dot{\mathbf{q}}_{ref}\ , \tag{6.14}$$

where $\mathbf{q}_{cmd}$ are the commands sent to the robot, $\mathbf{q}_t$ is the estimated robot configuration, $\boldsymbol{\Delta} t$ is the integration time and $\dot{\mathbf{q}}_{ref}$ is as defined in Eq. 6.13.

6.2 Experimental Results

6.2.1 Experimental Setup

The estimation and control framework presented in the previous section has been evaluated in real-time on an experimental set-up comprising a 6 DOF KUKA KR5 robot and a commercial Logitech c920 camera. Even though this robot is equipped with joint encoders, we do not use those values for state estimation or control, but we record them as ground truth which we use for performance evaluation. Our architecture was implemented in C++ and executed on an ordinary PC running Linux (8 GB RAM, 3.1 GHz Intel core i5 CPU). The communication between the PC and the robot controller was realised using TCP/IP. CAD models of robot parts were supplied to the tracker along with the part poses for a pre-defined home position during the initialisation step. All the proposed optimisation problems are solved using the constrained optimisation by linear approximation (COBYLA) algorithm available in NLopt C++ library [146].

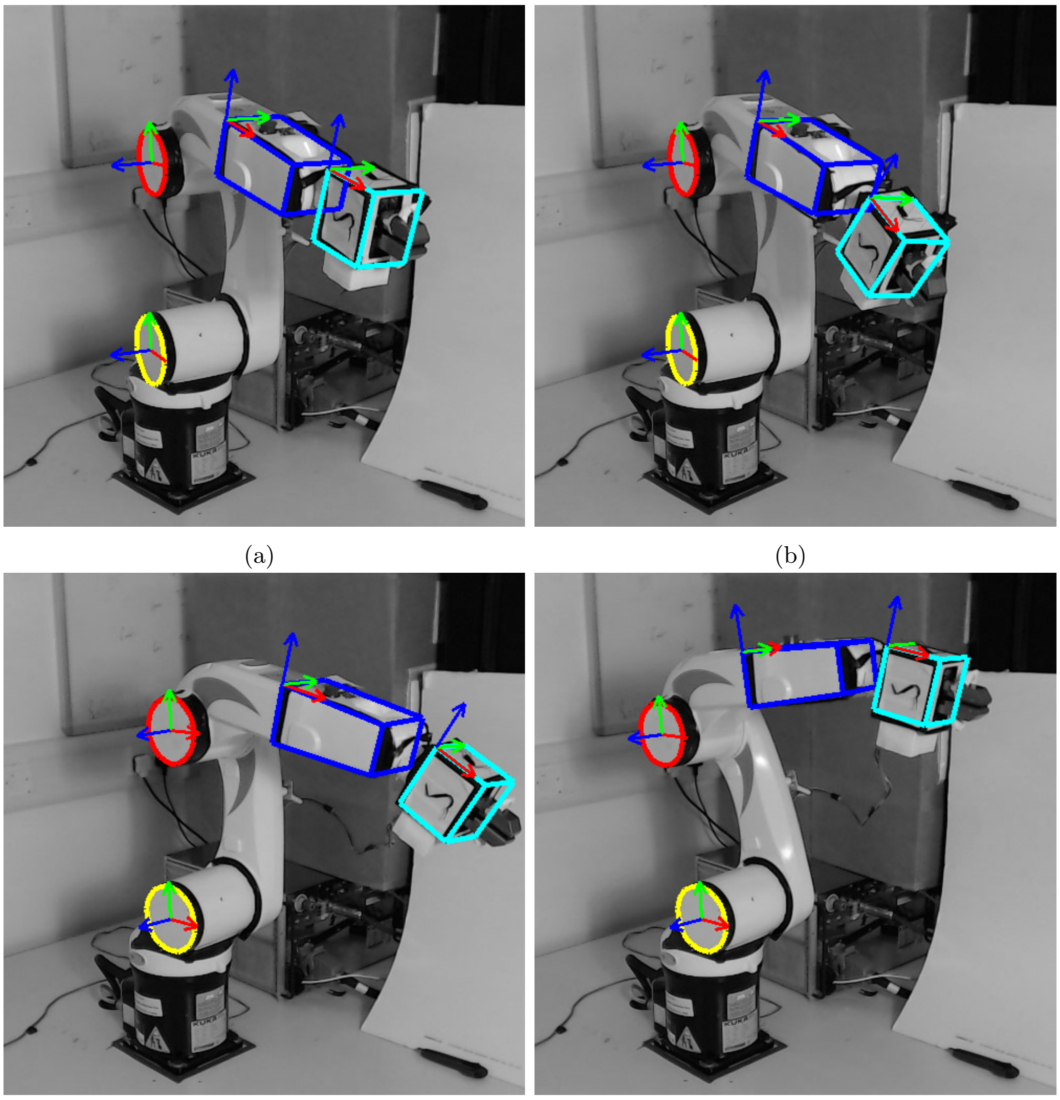
Two series of experiments have been conducted. Firstly, we assess the precision of the proposed framework in estimating the robot configuration. For this purpose, the robot was asked to repeatedly perform three different trajectories, and the vision-estimated joint angles were compared to the ground-truth angles derived from the robot's joint encoders. Secondly, we used the vision-derived state estimates as feedback in a kinematic control loop in order to perform regulation tasks. On average, a single control iteration including visual tracking of all the parts takes a computational time of 12 ms, *i.e.*, the system is capable of running at equivalent of > 80 frames per second on the CPU of an ordinary computer.

6.2.2 State Estimation

To analyse the performance of our state estimation module, 3 arbitrary trajectories were performed as shown in the first column of Table 6.1. The trajectories were selected such that all robot parts were visible in the camera field of view at all times. Trajectory 3 was chosen so as to excite one joint at a time. Fig. 6.4 shows screenshots of the tracking during various trajectories. Each trajectory was executed 5 times in order to perform quantitative analysis (measuring accuracy and precision). The obtained results are summarised in Table 6.1. It can be seen that the chained method consistently outperformed the full method in all the tests, and its average error typically remains lower than $4°$ on all joints. Fig. 6.5(a) and 6.5(b) show the estimated states of all joints during trajectories 2 and 3, respectively using the chained method. These results clearly demonstrate the efficiency of our state estimation framework in terms of accuracy and repeatability.

6.2.3 Controller

In this section, we use the values provided by the state estimation module as feedback for a kinematic controller. Five different goal positions were selected in the robot task space and the objective was to regulate the error using the estimated joint values and position the robot end-effector at the desired location. Fig. 6.6 reports the evolution of robot end-effector position values during one of the runs. For comparison purposes, we also show the values computed using joint encoders. It can be seen that the process converged at around ± 10 mm on all three axes, which is quite acceptable for the tasks this framework has been designed for. Additionally, the overall task space convergence was analysed using a cost function given by the squared error. Fig. 6.7 shows the cost variations during all five tasks. Also the trajectories followed by the end-effector during two of the tasks are shown in Fig. 6.8. The obtained results clearly demonstrate the

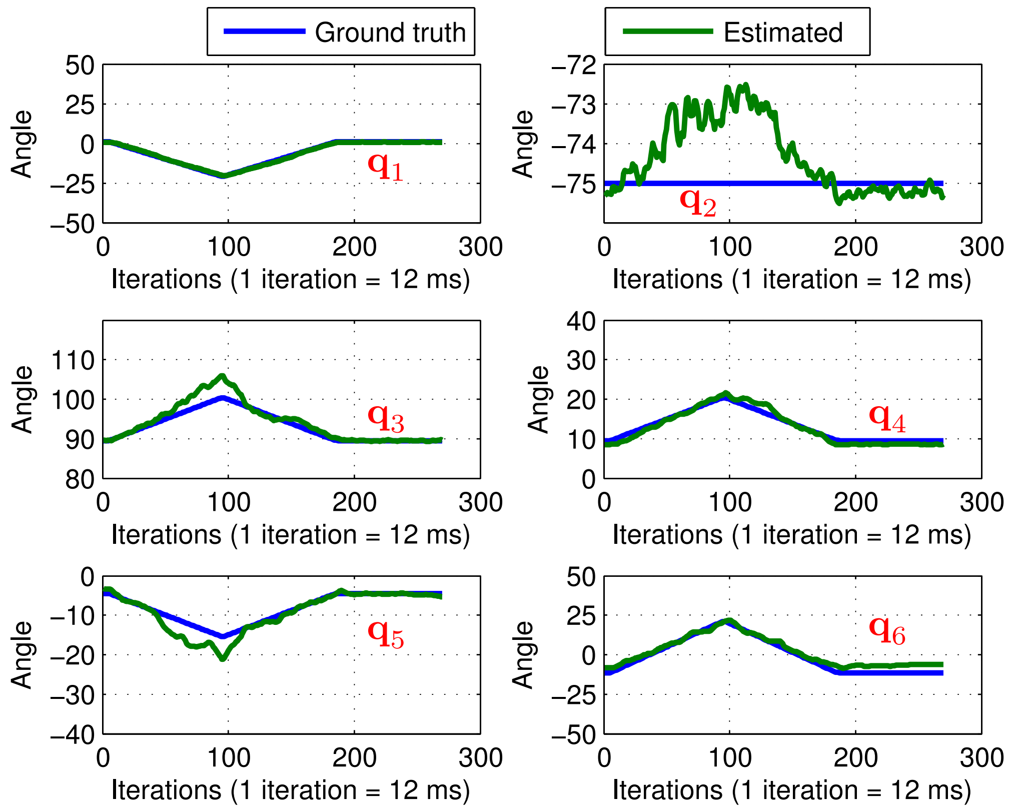


(a) (b) (c) (d)

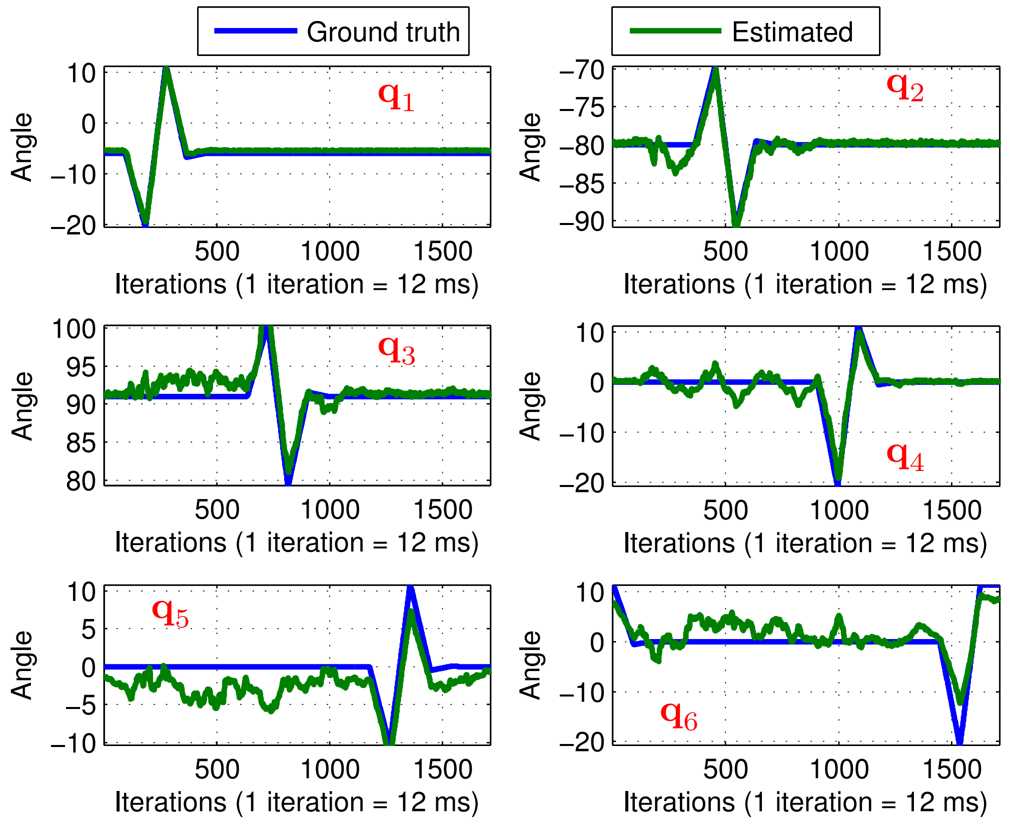
FIGURE 6.4: Tracking of parts for state estimation during various trajectories. More results can be found in the supplementary video submitted to IROS 2016.

robustness of our approach in estimating the robot joint state values and using them for regulation tasks.

Additionally, a test was performed in which the robot was required to move its end-effector along a trajectory tracing out the perimeter of a square. Corners of the square were supplied as goal positions to the controller. Results of three different runs are illustrated in Fig. 6.9.



(a) State estimation for trajectory 2.



(b) State estimation for trajectory 3.

FIGURE 6.5: Real and estimated states with various trajectories using chained method. Angles are expressed in degrees.

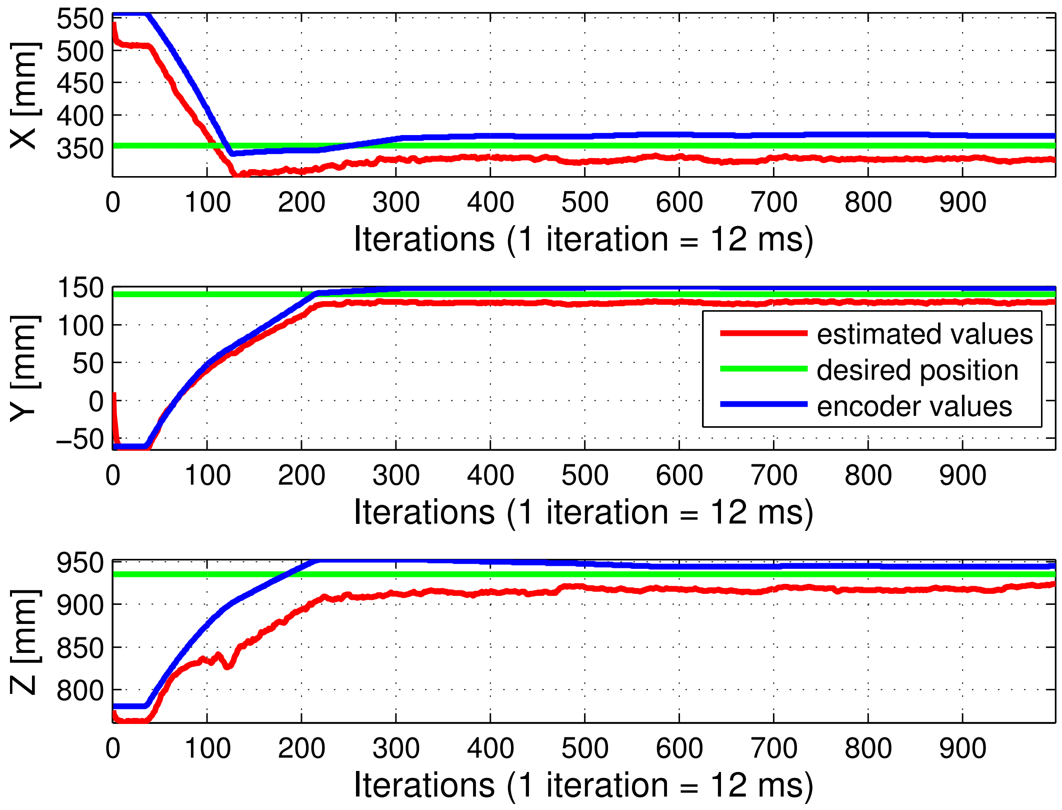


Figure 6.6: Evolution of the robot end-effector positional values on all three axes during a regulation task.

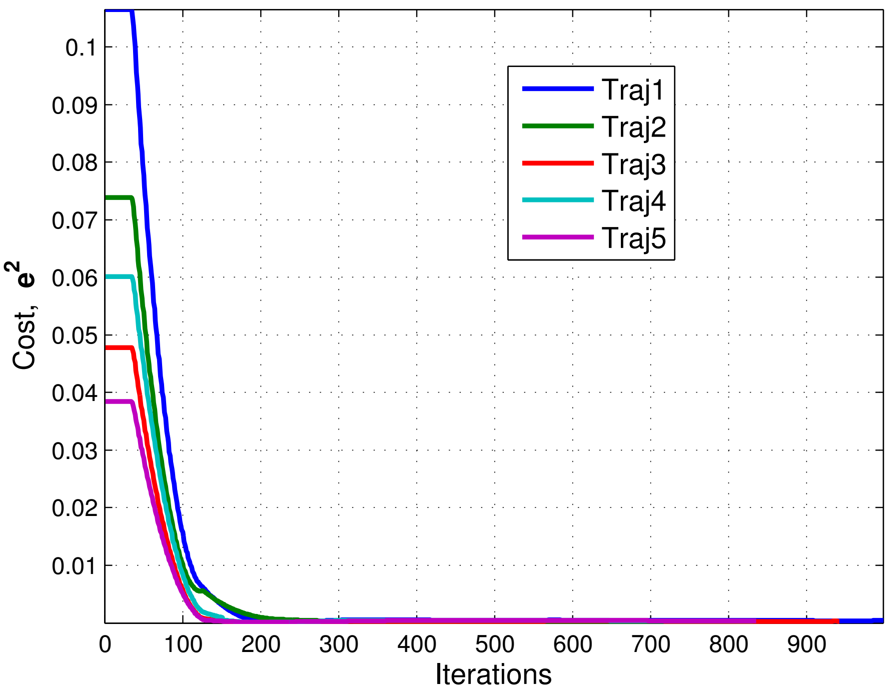
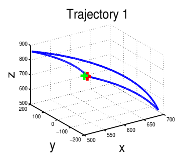
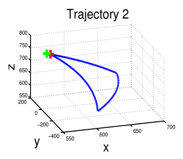
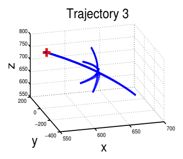
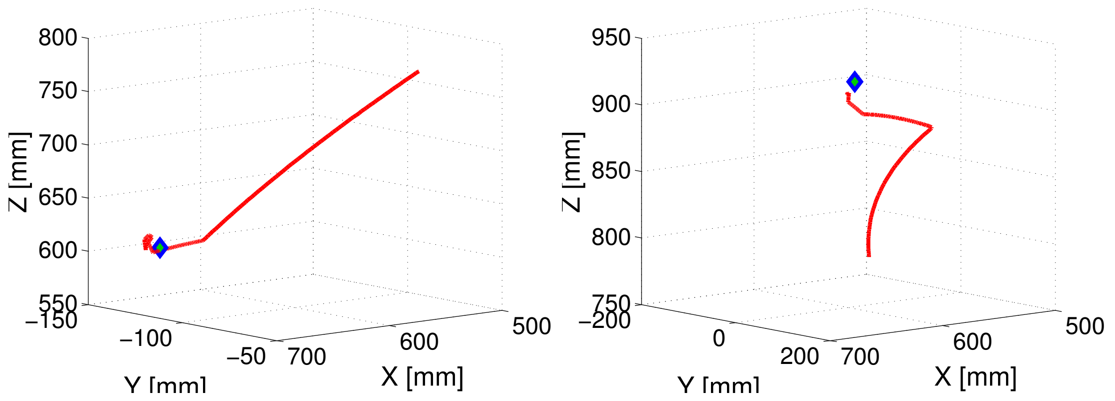


Figure 6.7: Cost variations during all five regulation tasks.

TABLE 6.1: Performance analysis of the proposed framework.

Trajectory	Joint	Chained method		Full method	
		RMSE	STD	RMSE	STD
	q_1	**0.9276**	**0.8853**	3.0474	2.8839
	q_2	**2.2015**	1.9099	5.6387	**1.7151**
	q_3	**3.5611**	**1.6257**	5.3594	2.8850
	q_4	**2.3186**	**2.2327**	2.7001	2.3568
	q_5	3.3825	2.4752	**3.0220**	**2.4087**
	q_6	3.8366	3.2841	**3.4607**	**2.6246**
	q_1	**0.4684**	**0.4647**	1.7158	1.6921
	q_2	**1.0553**	**0.8797**	6.5830	1.1917
	q_3	**2.1721**	**1.8104**	6.4424	3.5762
	q_4	**0.9231**	0.8509	1.3283	**0.7681**
	q_5	2.6000	2.0987	**2.2649**	**2.0423**
	q_6	3.4227	**2.0007**	**2.7552**	2.4180
	q_1	**0.5818**	**0.2619**	1.4042	0.9870
	q_2	**0.8010**	**0.7774**	9.7963	1.4538
	q_3	**1.3725**	**1.0542**	10.0045	1.9927
	q_4	**1.5080**	**1.5049**	2.1949	2.0882
	q_5	**2.5129**	1.6345	2.5898	**1.4391**
	q_6	4.1069	2.6367	**3.1573**	**2.5723**
Overall (avg. values)	q_1	**0.6593**	**0.5373**	2.0558	1.8543
	q_2	**1.3526**	**1.1890**	7.3393	1.4535
	q_3	**2.3686**	**1.4968**	7.2688	2.8180
	q_4	**1.5832**	**1.5295**	2.0744	1.7377
	q_5	2.8318	2.0695	**2.6256**	**1.9634**
	q_6	3.7887	2.6405	**3.1244**	**2.5383**

RMSE:- Root mean square error. STD:- Standard deviation.



(a) Run - 1 (b) Run - 2

FIGURE 6.8: Trajectories followed by the end-effector during two different regulation tasks. Diamond shaped point represents goal position.

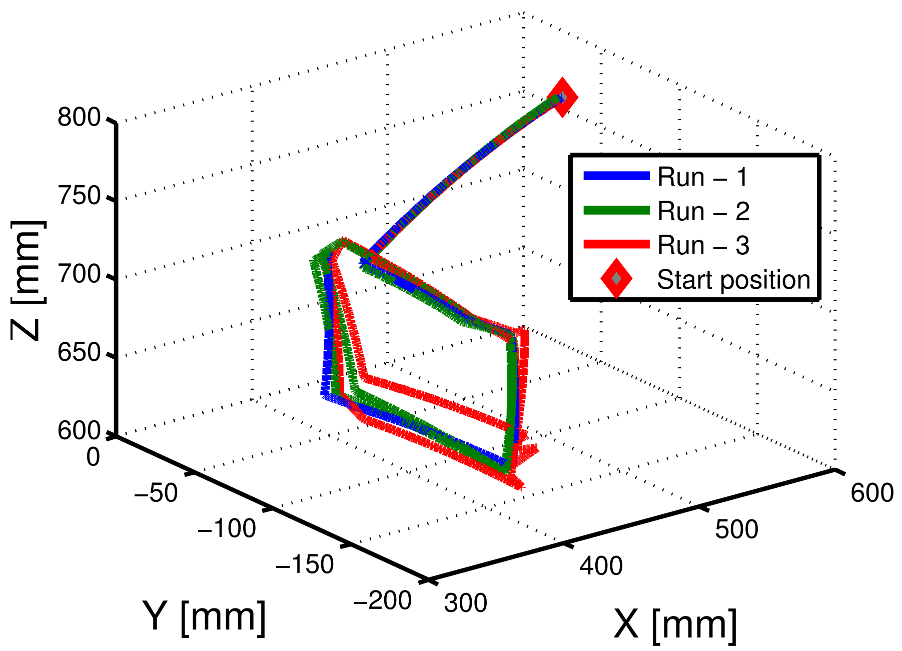


Figure 6.9: Square-perimeter trajectories followed by the end-effector. Diamond marker represents robot starting position.

6.2.4 Discussion

During these experiments, the root mean square errors of the joint angle estimates with respect to ground truth values were generally less than 4°. The chained method outperforms the full method most of the time. In our opinion, this can be explained by supposing that the first three robot parts were tracked very accurately and so the chained method was able to estimate the relative joint configurations very precisely. The last robot part (a cuboid section of the gripper mounted on the end-effector) represented more of a challenge to the tracking algorithm, and demonstrated slightly worse results. On the other hand, the full method averages out the performances of the trackers for each robot part, thereby achieving slightly better performances in the last two joints at the cost of a slightly higher error on the first two joints. In the Cartesian regulation tasks, the errors at pseudo-steady state are approximately 10 mm on each Cartesian axis.

6.3 Conclusion

This Chapter has presented a framework to estimate the configuration of an under-sensored robot through the use of a single monocular camera. First, we track several parts of the robot using an algorithm based on virtual visual servoing, then we use the information given by each part's tracker, combined with the kinematic model of the robot, to compute an estimation of the configuration of the robot. We present two alternative methods, and highlight their differences.

A study of the precision and robustness of the estimation methods was presented, as well as the results of using those state estimations in a kinematic controller used to perform Cartesian tasks. Joint angle errors not greater than 4° were achieved consistently in the estimation module using the chained method, and a Cartesian error of 10 mm on each axis was achieved while performing Cartesian regulation tasks. These results are acceptable for many practical tasks in the nuclear industry, however in future work, we believe that these estimates can be improved by: incorporating kinematic constraints into the visual tracking module; estimating velocities and accelerations; exploiting such physical information to robustify tracking [147]. Also, it might be useful to use probabilistic approaches as in [148].

This work has been motivated by the needs arising in the nuclear industry, where many robotic devices do not possess proprioceptive sensors. However, we believe that a larger community can benefit from this work. For example, this framework could be used in other fields of application, such as human-robot interaction, where robots ideally are asked to understand the movements of human agents in order to perform safe and effective interaction. In future work, we aim to generalise this framework to other robots and to different tasks. Finally, we are interested in estimating velocities and accelerations alongside joint values.

This framework can also be easily extended to take into consideration possible motions of the base of the robot. Because of forceful interactions of end-effector tools with the environment, the base of mobile manipulators can be significantly perturbed. These unpredictable motions of the base can be regarded as additional degrees of freedom to be modelled in the kinematics of the robot. A possible solution would be an additional tracker for the base.

Chapter 7

General Control Framework

In the nuclear industry, many tasks are complicated by a strong unreliability of the proprioceptive sensors and sometimes by a complete lack of such sensors. Tasks as simple as following a trajectory are then realised with the help of a human operator manually operating or tele-operating the robotic devices by turning switches on a teach pendant. In Chapter 6, we introduced a vision-based framework capable of estimating a manipulator configuration in real-time and using only an RGB camera. The goal of such framework is to increase the level of autonomy of the robotic manipulator with the aid of external sensors, *i.e.*, a camera, and without on-body hardware modification. We demonstrated in the experimental section, that such framework succeeds in estimating the configuration of the robot up to an acceptable precision. Also, we showed that these estimates can be effectively used by a kinematic controller to guide the robot to perform Cartesian regulations.

Previously, in Chapter 4 we described how projected dynamics can be effectively used to lower the torque effort while performing a task that involves contacts with the environment. This is most important in industrial situations, since lowering the required torques for a task, means to lower costs associated to such task. Moreover, in Chapter 5 we proposed a method to estimate the kinematic constraints due to contacts with the environment by the means of an analysis of the velocity vectors which differ from the velocity vectors commanded during an exploratory phase. This method affords to relax assumptions on the geometry of the contact and thus on the kinematic constraint.

This Chapter integrates the previous contributions into a general framework capable of: 1. estimating the robot configuration using a camera; 2. controlling the robot kinematically; 3. detecting contacts with the environment; 4. estimating kinematic constraints possibly arising from contacts with the environment. The pipeline of such framework is represented in Fig. 7.1.

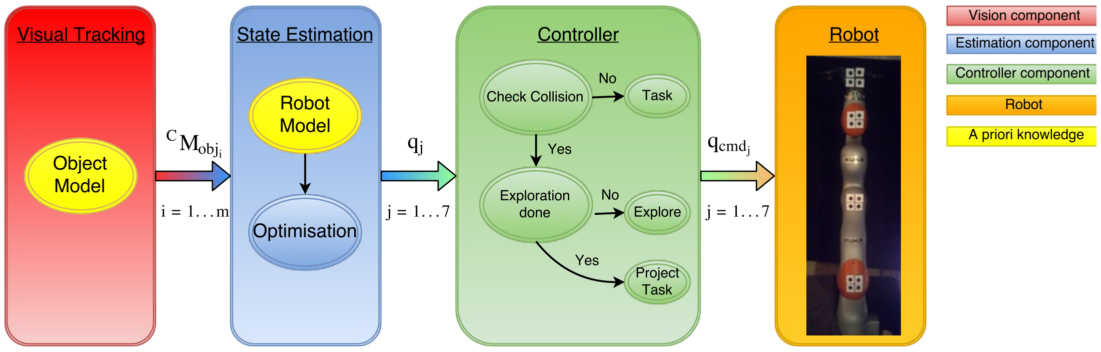


FIGURE 7.1: Illustration of the proposed framework pipeline. A single camera images are used to track the robot (visual tracking, in red). The robot model is used in an optimisation problem to estimate the configuration of the robot in real time (state estimation in blue). These estimates are used to control the robot and to detect possible contacts. Moreover, when a contact is detected, an exploratory phase starts and the kinematic constraints are estimated (controller, in green). The controller generates commands to send to the robot, in orange.

The remainder of this Chapter is organised as follows: Sect. 7.1 describes the framework used to estimate the robot configuration and highlights the differences with respect to Chapter 6. Sect. 7.2 describes two methods we used to detect contacts whilst Sect. 7.3 relates to Chapter 5 restating some of the key concepts of our method to estimate kinematic constraints. Sect. 7.4 shows results obtained on a KUKA iiwa robot on configuration estimation tasks and on robot servoing with tasks involving contacts with the environment, thus triggering contact detection and constraint estimation procedures. Finally, we conclude the Chapter with further considerations and future directions in Sect. 7.5.

7.1 Vision-based State Estimation

It was already stated in Chapter 6 that the main goal of the proposed vision-based estimation method is to compute reliable quantitative configuration feedback when robots are under-sensored, *i.e.*, when encoder measurements are not available. We constructed the framework to be modular as to be able to replace each component effortlessly. Fig. 6.1 summarises the architecture of the framework. We follow the same approach in this Chapter, and we show the versatility of the architecture: 1. using a different tracking approach; 2. using a different manipulator. The state estimation module is fundamentally unchanged, with the only exception of the kinematic relations relating the tracked reference frames to the robot. Such relations have been changed in accordance to the robot kinematic model. Next subsections detail our choice of visual tracking algorithm, recall our state estimation method and the classical kinematic controller which has been used.

7.1.1 Visual Tracking

As stressed in numerous occasions, our focus is not in designing a novel tracking algorithm. On the contrary, we used a very classical approach exploiting markers. In detail, we put markers on specific known parts of the robot's body. Those markers are composed of four dots on a white square. Given the positions of the dots of such markers with respect to the centre of the relative square in the 3D space, basic computer vision techniques provide the means to compute the pose of the centre of such square with respect to the camera frame.

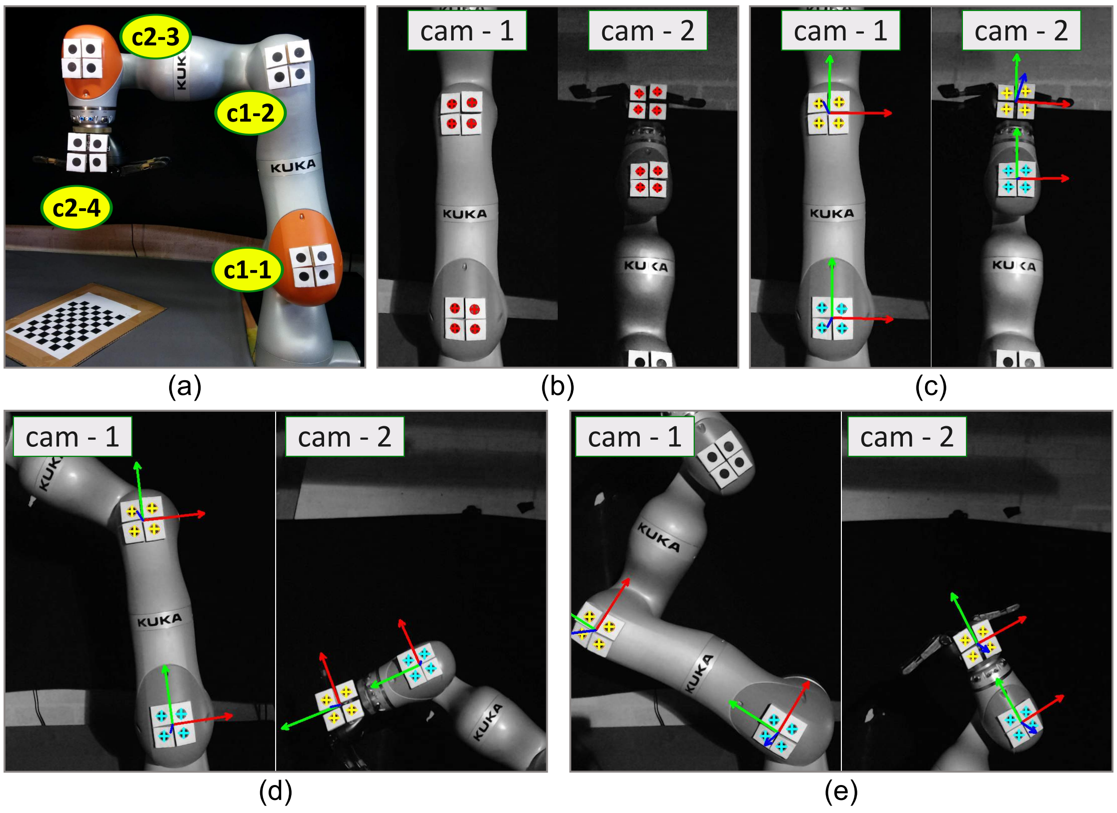


Figure 7.2: Illustration of visual tracking and object's pose estimation. (a) Four custom designed markers are placed on different links of a Kuka lbr iiwa manipulator arm. Each marker has four dots (one out-of-plane) whose position with respect to the centre of the marker reference frame is known a priori. In the image, markers are named $1 \cdots 4$ and the cameras by which they are tracked are indexed as c1 and c2. Hereafter the images of both cameras are shown in a single frame and for illustration purposes, the original frames are cropped-to-fit. (b) Automatically initialised dots for visual tracking in the initial frame of the state estimation process. (c)-(e) Tracked poses of each object in later frames while the robot is in motion.

In this work, we track four markers put on a KUKA iiwa robot as in Fig. 7.2. The trackers are initialised by the user mouse-clicking on the corresponding dots in the first image, while the robot is in its home position. This is to ensure the knowledge of the

3D position of the dots at time t = 0 and thus also to allow the state estimation routine to compute the fixed relationship between the camera frame and the robot base frame.

7.1.2 State Estimation

In the following, the standard convention for symbols associated with the kinematics of the robot is observed, *e.g.*, we define $\mathbf{q}$ and $\dot{\mathbf{q}}$ as the configuration and the velocity respectively of the robot in joint space.

We use the following key idea to estimate the configuration of the robot, as in Chapter 6. As shown in Fig. 7.3, there are two paths from the camera reference frame ${}^{C}RF$ (in yellow) to each tracked part frame ${}^{obj_i}RF$ (in red). As stated, we track four different parts of the robot, *i.e.*, $i = 1 \dots 4$. These two paths kinematically coincide, thus we enforce the following equalities to estimate the state

$$ {}^{C}\mathbf{M}_{obj_i} = {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_{obj_i}(\mathbf{q})\,, \tag{7.1} $$

where, ${}^{C}\mathbf{T}_0$ is the transformation from camera to world frame and ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$ represents the transformation from world to object i frame parametrised over the joint values $\mathbf{q}$, *i.e.*, ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$ embeds the kinematic model of the robot. Specifically, for each tracked robot part, we get

$$ {}^{C}\mathbf{M}_{obj_1} = {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_1(q_1){}^{1}\mathbf{T}_2(q_2){}^{2}\mathbf{T}_{obj_1}\,, \tag{7.2} $$
$$ {}^{C}\mathbf{M}_{obj_2} = {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_1(q_1){}^{1}\mathbf{T}_2(q_2){}^{2}\mathbf{T}_3(q_3){}^{3}\mathbf{T}_{obj_2}\,, \tag{7.3} $$
$$ {}^{C}\mathbf{M}_{obj_3} = {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_1(q_1){}^{1}\mathbf{T}_2(q_2){}^{2}\mathbf{T}_3(q_3){}^{3}\mathbf{T}_4(q_4){}^{4}\mathbf{T}_5(q_5){}^{5}\mathbf{T}_6(q_6){}^{6}\mathbf{T}_{obj_3}\,, \tag{7.4} $$
$$ {}^{C}\mathbf{M}_{obj_4} = {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_1(q_1){}^{1}\mathbf{T}_2(q_2){}^{2}\mathbf{T}_3(q_3){}^{3}\mathbf{T}_4(q_4){}^{4}\mathbf{T}_5(q_5){}^{5}\mathbf{T}_6(q_6){}^{6}\mathbf{T}_7(q_7){}^{7}\mathbf{T}_{obj_4}\,, \tag{7.5} $$

The state of the robot is now estimated by imposing the equality given in Eq. 7.1, and casting it as an optimisation problem. As already mentioned, we assume that we know the initial configuration of the robot (occupying its home position in the first image) and its kinematic model. The robot's initial configuration is used as a seed for the first iteration of the optimisation problem and the kinematic model is used to compute ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$. The optimisation problem is then stated as

$$ \begin{aligned} \underset{\boldsymbol{q}}{\text{minimise}} \quad & \sum_i \mathbf{e}_i(\mathbf{q}) \\ \text{subject to} \quad & |q_j| \le q_{max_j} \end{aligned} \tag{7.6} $$

where

$$ \mathbf{e}_j(\mathbf{q}) = vec({}^{C}\mathbf{M}_{obj_i} - {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_{obj_i}(\mathbf{q})) \tag{7.7} $$

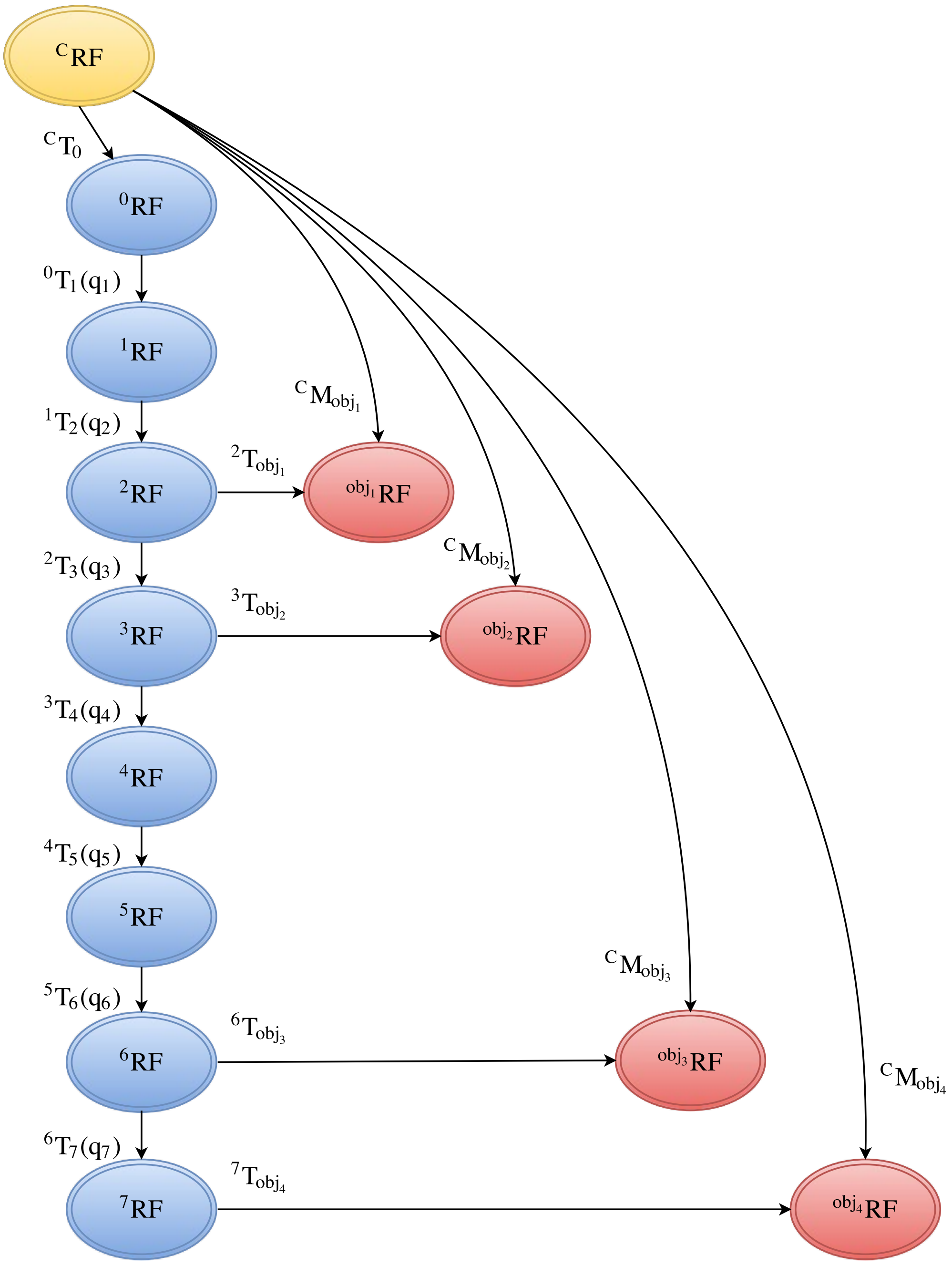


FIGURE 7.3: Illustration of the proposed state estimation model. Nodes represent reference frames and are classified by various colours: camera frame in yellow, robot frames in blue and tracked object frames in red. The two paths leading to each tracked object frame $^{obj_i}RF$ from the camera reference frame ^{C}RF can be seen.

represents an error in the difference of the two paths shown in Fig. 7.3, and q_{max_j} is the joint limit for joint j. In order to compute ${}^0\mathbf{T}_{obj_i}(\mathbf{q})$, we use the convention of Denavit-Hartenberg. The overall estimation schema along with the transformation matrices between each reference frame are shown in Fig. 7.3.

Because of the better performances achieved in Chapter 6, we decided to use the chained method. This method uses each object to estimate only a subset of joint values. These, in turn, are used as known parameters in the successive estimation problems. Namely, q_1 and q_2 can be retrieved using obj_1, as in

$$\begin{aligned} &\underset{q_1,q_2}{\text{minimise}} && \mathbf{e}_1(q_1,q_2) \\ &\text{subject to} && |q_j| \le q_{max}\; j=1,2, \end{aligned} \tag{7.8}$$

and from now on q_1 and q_2 are treated as known. q_3 is estimated using obj_2

$$\begin{aligned} &\underset{q_3}{\text{minimise}} && \mathbf{e}_2(q_1,q_2,q_3) \\ &\text{subject to} && |q_3| \le q_{max}\,. \end{aligned} \tag{7.9}$$

In a similar fashion, q_4, q_5 and q_6 are estimated using obj_3

$$\begin{aligned} &\underset{q_4,q_5,q_6}{\text{minimise}} && \mathbf{e}_3(q_1,q_2,q_3,q_4,q_5,q_6) \\ &\text{subject to} && |q_j| \le q_{max},\; j=4,5,6. \end{aligned} \tag{7.10}$$

And finally, obj_4 provides the equations to compute q_7

$$\begin{aligned} &\underset{q_7}{\text{minimise}} && \mathbf{e}_4(\mathbf{q}) \\ &\text{subject to} && |q_7| \le q_{max}\,. \end{aligned} \tag{7.11}$$

7.1.3 Controller

For proof of principle, we implemented a classical kinematic controller of the form given in Eq. 7.12 to validate our methodology and also to demonstrate how the vision-derived state estimations can be used to servo the robot's end-effector.

$$\dot{\mathbf{q}}_{ref} = \mathbf{K}_P\mathbf{e} + \dot{\mathbf{q}}_d \tag{7.12}$$

Here, $\dot{\mathbf{q}}_{ref}$ is the desired/reference velocity. The error $\mathbf{e} = \mathbf{q}_d - \tilde{\mathbf{q}}$ has been defined as the difference between desired and estimated joint positions $\tilde{\mathbf{q}}$. Finally, $\mathbf{K}_P$ is the proportional gain matrix. Since the commands sent to the robot are in position, Eq. 7.12

is integrated numerically to compute such commands

$$\mathbf{q}_{cmd} = \tilde{\mathbf{q}}_t + \Delta t \; \dot{\mathbf{q}}_{ref} \, , \tag{7.13}$$

where $\mathbf{q}_{cmd}$ are the commands sent to the robot, $\tilde{\mathbf{q}}_t$ is the estimated robot configuration, Δt is the integration time and $\dot{\mathbf{q}}_{ref}$ is as defined in Eq. 7.12.

7.2 Contact Detection

While performing a task, a robot can have contacts with the environment, and sometimes contacts are necessary to perform a task, *e.g.*, in manipulation tasks or while walking. Thus we implemented two methods to check contacts with the environment. The first method is based on the concept of residuals, and is described in detail in Subsec. 7.2.1. The second method is based on the sheer difference between expected configuration due to the task and current configuration, and is described in detail in Subsec. 7.2.2. We designed these methods to be used with the vision-based estimates computed with the techniques described in the previous section. However, these methods are general and can be used also when readings from sensors are available.

7.2.1 Residuals

The idea behind the method of the residuals is to model contacts as system's faults. This method has been proposed in [149], where the authors define residuals as

$$\mathbf{r} = \mathbf{K}[\int_0^t (\boldsymbol{\alpha} - \boldsymbol{\tau} - \mathbf{r}) \; dt + \mathbf{p}_m] \, , \tag{7.14}$$

with $\mathbf{r}(0) = \mathbf{0}$ and $\mathbf{K} > 0$. In detail, the momentum of inertia $\mathbf{p}_m = \mathbf{M}\dot{\mathbf{q}}$, yields if differentiated

$$\dot{\mathbf{p}}_m = \mathbf{M}\ddot{\mathbf{q}} + \dot{\mathbf{M}}\dot{\mathbf{q}} = \boldsymbol{\tau} + \boldsymbol{\tau}_C - \mathbf{h} + \dot{\mathbf{M}}\dot{\mathbf{q}} = \boldsymbol{\tau} + \boldsymbol{\tau}_C - \boldsymbol{\alpha}(\mathbf{q}, \dot{\mathbf{q}}) \, , \tag{7.15}$$

where $\boldsymbol{\alpha}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{h} - \dot{\mathbf{M}}\dot{\mathbf{q}}$.

Differentiating Eq. 7.14 to study how residuals change over time, we obtain

$$\dot{\mathbf{r}} = -\mathbf{K}\mathbf{r} + \mathbf{K}\boldsymbol{\tau}_C \, . \tag{7.16}$$

Analysing the dynamics of $\mathbf{r}$, $\mathbf{r} = \mathbf{0}$ holds true when $\boldsymbol{\tau}_C = \mathbf{0}$, while $\mathbf{r} \neq \mathbf{0}$ when $\boldsymbol{\tau}_C \neq \mathbf{0}$.

Residuals have been used recently also together with Particle Filters to identify external contacts as in [150]. Next subsections present our version of residuals defined in the Cartesian space and in the joint space. No dynamic model of the robot is required in our implementation.

7.2.1.1 Cartesian Space

We define

$$\mathbf{p} = \mathbf{f}(\mathbf{q})\,, \tag{7.17}$$

$$\dot{\mathbf{p}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} = \dot{\mathbf{p}}_r - \dot{\mathbf{p}}_C = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}_r - \mathbf{J}_C(\mathbf{q})\dot{\mathbf{q}}_r\,, \tag{7.18}$$

where

$$\mathbf{J}_C(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}\,. \tag{7.19}$$

$\dot{\mathbf{p}}_r$ is the commanded velocity, while $\dot{\mathbf{p}}_C$ can be interpreted as a virtual velocity induced by the contact at the end-effector. We define the residuals $\mathbf{r}$ as

$$\mathbf{r} = \mathbf{K}[\int_0^t (\dot{\mathbf{p}}_r - \mathbf{r})\; dt - \mathbf{p}]\,, \tag{7.20}$$

with $\mathbf{r}(0) = \mathbf{0}$. Differentiating we obtain

$$\dot{\mathbf{r}} = -\mathbf{K}\mathbf{r} + \mathbf{K}\dot{\mathbf{p}}_C\,. \tag{7.21}$$

Analysing the dynamics of $\mathbf{r}$, $\mathbf{r} = \mathbf{0}$ holds true when $\dot{\mathbf{p}}_C = \mathbf{0}$, while $\mathbf{r} \neq \mathbf{0}$ when $\dot{\mathbf{p}}_C \neq \mathbf{0}$.

7.2.1.2 Joint Space

We define

$$\dot{\mathbf{q}} = \dot{\mathbf{q}}_r - \dot{\mathbf{q}}_C\,. \tag{7.22}$$

$\dot{\mathbf{q}}_r$ is the commanded velocity, while $\dot{\mathbf{q}}_C$ can be interpreted as a virtual velocity induced by the contact at the end-effector. We define the residuals $\mathbf{r}$ as

$$\mathbf{r} = \mathbf{K}[\int_0^t (\dot{\mathbf{q}}_r - \mathbf{r})\; dt - \mathbf{q}]\,, \tag{7.23}$$

with $\mathbf{r}(0) = \mathbf{0}$. Differentiating we obtain

$$\dot{\mathbf{r}} = -\mathbf{K}\mathbf{r} + \mathbf{K}\dot{\mathbf{q}}_C \,. \tag{7.24}$$

Analysing the dynamics of $\mathbf{r}$, $\mathbf{r} = \mathbf{0}$ holds true when $\dot{\mathbf{q}}_C = \mathbf{0}$, while $\mathbf{r} \neq \mathbf{0}$ when $\dot{\mathbf{q}}_C \neq \mathbf{0}$. Also, in this case, $\mathbf{r}$ should give also the information on where the contact occurs (q_i such that first $r_i \neq 0$).

7.2.1.3 Kinematic Residuals using Estimates

Until now, the formulations of the residuals assumed perfect knowledge of the state of the robot, *i.e.*, torques or configurations and velocities. In our scenario, only estimates of the configurations are available. This introduces a change in the model. We describe the case in the joint space here. Because of the error in the estimation process, $\mathbf{q}$ is not known but an estimate of it, $\tilde{\mathbf{q}}$, is available. If we express the commanded configuration as $\mathbf{q}_{cmd} = \tilde{\mathbf{q}}_{t-1} + \Delta t \; \dot{\mathbf{q}}_r$, then we model the system as

$$\tilde{\mathbf{q}} = \mathbf{q}_{cmd} + \boldsymbol{\epsilon} \,, \tag{7.25}$$

where

$$\boldsymbol{\epsilon} = \boldsymbol{\epsilon}_\Delta + \mathbf{q}_C \,, \tag{7.26}$$

i.e., the mismatch between real configuration and estimates is due to the error in the estimation process ($\boldsymbol{\epsilon}_\Delta$) and a possible contact ($\mathbf{q}_C$). If we substitute $\mathbf{q}_{cmd}$ into Eq. 7.25, we obtain

$$\dot{\tilde{\mathbf{q}}} = \dot{\mathbf{q}}_d + \dot{\tilde{\boldsymbol{\epsilon}}} \,, \tag{7.27}$$

where $\dot{\mathbf{q}}_d$ is the desired velocity as in $\dot{\mathbf{q}}_r = \mathbf{K}\;(\mathbf{q}_d - \tilde{\mathbf{q}}) + \dot{\mathbf{q}}_d$ and $\tilde{\boldsymbol{\epsilon}} = \mathbf{K}\Delta t\;(\mathbf{q}_d - \tilde{\mathbf{q}}) + \boldsymbol{\epsilon}_\Delta + \mathbf{q}_C$. If we define the residuals as

$$\mathbf{r} = \mathbf{K}[\int_0^t (\dot{\mathbf{q}}_d - \mathbf{r}) \; dt - \tilde{\mathbf{q}}] \,, \tag{7.28}$$

then

$$\dot{\mathbf{r}} = -\mathbf{K}\mathbf{r} + \mathbf{K}\dot{\tilde{\boldsymbol{\epsilon}}} = -\mathbf{K}\mathbf{r} + \mathbf{K}(\boldsymbol{\alpha}((\mathbf{q}_d - \tilde{\mathbf{q}}), \tilde{\boldsymbol{\epsilon}}) + \dot{\mathbf{q}}_C) \,, \tag{7.29}$$

where $\boldsymbol{\alpha}((\mathbf{q}_d - \tilde{\mathbf{q}}), \tilde{\boldsymbol{\epsilon}})$ is the derivative over time of $\mathbf{K}\Delta t\;(\mathbf{q}_d - \tilde{\mathbf{q}}) + \boldsymbol{\epsilon}_\Delta$. Hence, the residuals become different than zero when there is an error in the estimates, or when there is a contact, or when both these two situations happen. We introduce a threshold in order to detect contacts. When the norm of the residuals is bigger than such threshold, a contact is detected. This does not avoid false detections to happen, however the choice of the threshold is critical to lower the number of such erroneous detections. We will expand this discussion in the experimental section.

7.2.2 Sheer difference

The idea behind this method is that if the robot is in a different position with respect to the expected/desired position, then it is possible that a contact has occurred and that contact has prevented the robot to go to the desired position. We propose to compute the norm of the difference between the current configuration and the configuration the robot should be in to perform the task. When this difference is greater than a threshold, then a contact is detected. In other words

$$|\mathbf{q} - \mathbf{q}_{des}| > \mathbf{q}_{thresh} \rightarrow \text{contact}. \tag{7.30}$$

This is a naive approach since a contact is not the only possible explanation for such difference, *e.g.*, the robot might have hit a joint limit or there might be an error in the estimation of the current configuration. However, it is very inexpensive computationally and represents a further check for collisions.

7.3 Contact Estimation using State Estimates

We proposed a method to estimate kinematic constraints in Chapter 5. One of the goals of this Chapter is to apply the methods already described in Chapter 5 to the case where only estimates of the robot configurations are available. This Section revises the key concepts of such methods, focusing on the definitions in the joint space.

7.3.1 Exploration in the Joint Space

We define the constraints as

$$\mathbf{\Lambda}^q(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}\,. \tag{7.31}$$

Here, $\mathbf{\Lambda}^q(\mathbf{q})$ can be regarded not only as a selection matrix, but also a constraint Jacobian. Moreover, $\mathbf{\Lambda}^q(\mathbf{q})$ is dependent on the configuration of the robot, in contrast to Eq. 5.2. Explorative motions can also be defined directly in the joint space. Similar to the Cartesian case already described in Chapter 5, there are two possibilities for the expected $\dot{\mathbf{q}}^{exp}$ and the observed $\dot{\mathbf{q}}^{obs}$:

1. in the free motion subspace: $\dot{\mathbf{q}}^{exp} = \dot{\mathbf{q}}^{obs}$;
2. in the constrained motion subspace: $\dot{\mathbf{q}}^{exp} \neq \dot{\mathbf{q}}^{obs}$.

We collect a set of $\dot{\mathbf{q}}^{obs}$ from the latter case, and use singular value decomposition to estimate the selection matrix $\mathbf{\Lambda}^q(\mathbf{q})$.

7.4 Experimental Results

7.4.1 Experimental Setup

The proposed robot state estimation and contact detection framework has been evaluated using the experimental setup shown in Fig. 7.4. The robot used for this work is an industrial collaborative robot, Kuka iiwa 14 R820 with 7 degrees of freedom and the vision system consists of two commercial Logitech c920 cameras. The cameras are placed outside the robot task space and are directed such that each camera can view a set of two fiducial markers placed on the robot. For the purpose of 3D pose estimation from visual tracking, we select the markers from ViSP [151], *i.e.*, each marker comprises 4 dots of 20 mm diameter glued to cardstocks. An example of visual tracking is reported in Fig. 7.5. All the hardware components of our setup are interfaced with a work computer (master) running Linux through their respective interfaces. Overall framework has been implemented in C++ on the master computer. All the matrix computations are performed using ViSP and for the sake of non-linear optimisation, we used the implementations from open source NLopt library. The asynchronous communication between the robot controller running Kuka Sunrise OS and master has been realised over UDP/IP with a communication speed of < 2 ms.

Three sequences of experiments have been conducted. In the first sequence of experiments, we first assess the precision of the proposed framework in estimating the robot configuration and, secondly, we used the vision-derived state estimates as feedback in a kinematic control loop in order to perform some tasks.

In the second sequence of experiments, we detect contacts occurring when the robot is performing a task. First we show how the mentioned contact detection methods work when readings from encoders are available. Then we rely on the vision-based estimates of the joint configuration to control the robot and simultaneously detect contacts. This sequence of experiments has been conceived i) to show how the presented contact detection methods work in situations when encoders readings are available and ii) to show the impact of the errors in the estimates of joint configurations on the contact detection quality.

In the third sequence of experiments, we use these configuration estimates to also estimate kinematic constraints deriving from contacts with the environment. This final sequence of experiments want to show how it is possible to use all the methods described in the previous sections in one framework.

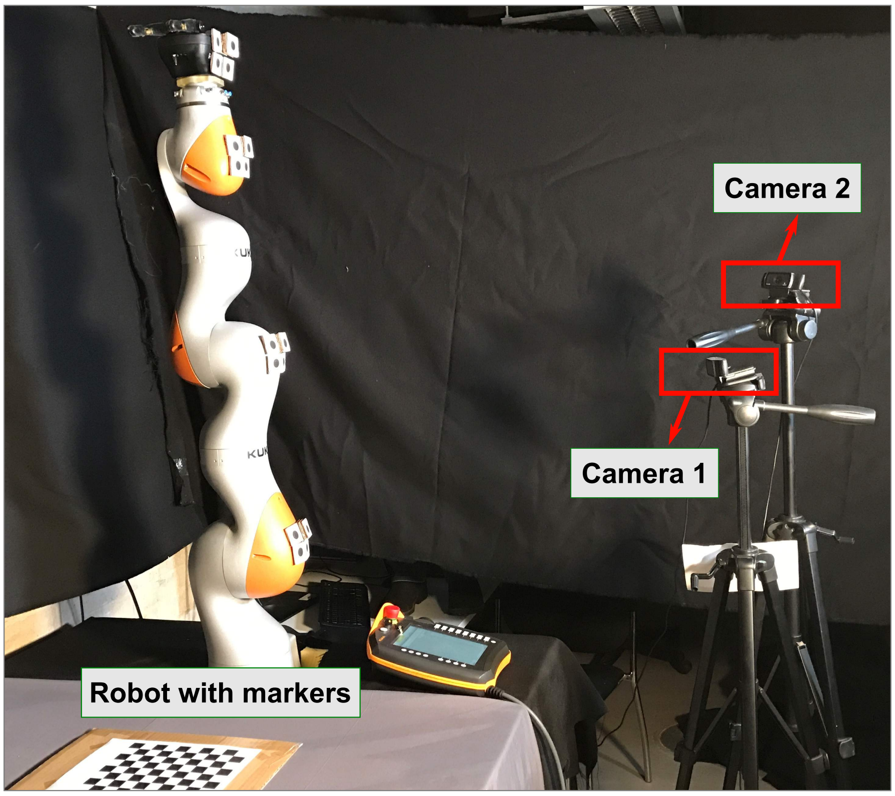
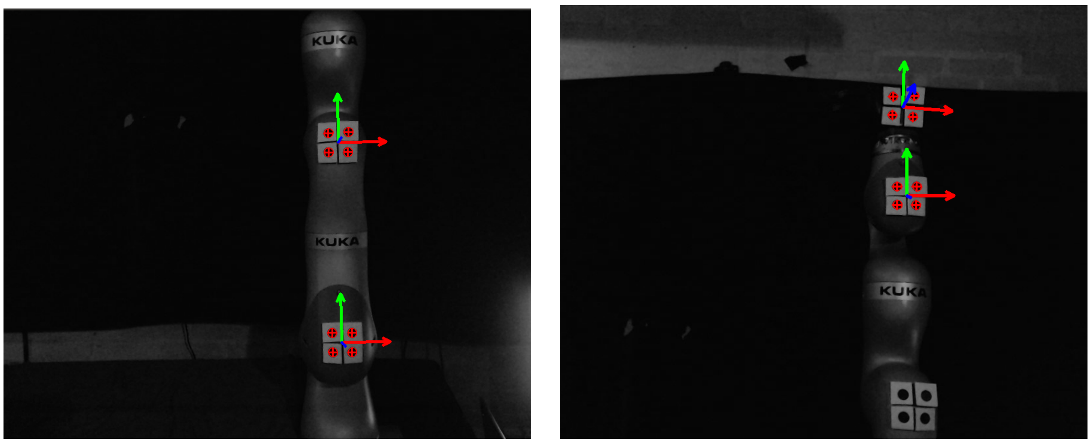


FIGURE 7.4: Experimental set-up illustrating the used Kuka robot with attached markers on its various links and the cameras. The cameras are positioned in the workspace such that each camera can successfully track two markers.



(a) Camera 1 - Visual tracking of markers 1 and 2

(b) Camera 2 - Visual tracking of markers 3 and 4

FIGURE 7.5: Visual tracking example. Two cameras are respectively in charge of tracking markers 1 and 2 and markers 3 and 4.

7.4.2 Vision-based Estimation

This section comprises experiments to study precision and reliability of the presented vision-based estimation method. In detail, first we provide the reader with a thorough analysis of the results of experiments where the robot performs different trajectories

Table 7.1: RMSE of estimates with respect to ground truth values during experiments with trajectories involving only a joint at a time. Errors are expressed in degrees.

joint(#)	RMSE (in degrees (°))						
	traj 1	traj 2	traj 3	traj 4	traj 5	traj 6	traj 7
q_1	2.2891	2.6875	0.2936	0.4165	0.2121	0.2133	0.2888
q_2	2.7439	0.4683	0.0760	0.0639	0.0964	0.0713	0.1222
q_3	1.1297	1.2694	2.2049	0.8130	0.6930	0.4263	0.4964
q_4	4.5347	4.0912	4.4199	1.1997	10.6481	0.3550	0.3451
q_5	2.7826	5.4091	3.0292	1.9575	3.2157	0.6363	0.4099
q_6	3.4006	3.5237	5.0617	0.9501	10.1702	0.5623	0.2285
q_7	1.9477	2.1483	1.7199	2.0463	2.5113	1.7858	5.7916

Table 7.2: RMSE of estimates with respect to ground truth values during experiments with trajectories involving multiple joints. Errors are expressed in degrees.

joint(#)	RMSE (in degrees (°))						
	traj 8	traj 9	traj 10	traj 11	traj 12	traj 13	traj 14
q_1	3.6334	2.1395	0.5796	0.1808	2.6107	1.8684	2.8039
q_2	2.5905	1.1701	1.6013	0.1294	0.5681	1.7348	1.5421
q_3	2.7890	1.4507	0.7948	0.4666	1.1378	0.8673	1.6197
q_4	5.2781	5.2637	4.8120	3.8783	3.4090	5.6384	4.1801
q_5	2.2680	3.2093	2.2618	1.1640	5.2083	5.1882	6.0621
q_6	2.7531	4.5648	3.8020	4.1278	3.2126	4.3473	3.9683
q_7	2.4156	2.8233	7.7342	1.6220	3.0252	3.3468	4.1356

and our method estimates the robot joint values in realtime. Secondly, we show how these estimates can be successfully used as feedback in a kinematic controller. These experiments replicate the experiments carried out in [4].

7.4.2.1 Estimating joint configurations

This first set of experiments focussed only on the estimation of the robot joint configuration. The robot was asked to perform 14 different trajectories. The first 7 trajectories excite only one joint each, *i.e.*, trajectory 1 uses only joint 1, trajectory 2 only joint 2, and so on. The remaining 7 trajectories contain motions for multiple joints at a time. We report each joint RMSE between real values and estimates for each trajectory and also the average RMSE and standard deviation over the RMSE of all the trajectories.

Tables 7.1 and 7.2 show the RMSE errors of each joint for each of the 14 trajectories. Table 7.3 provides the reader with mean and standard deviation of such RMSE as computed over the results presented in Tables 7.1 and 7.2. Figs. 7.6 and 7.7 show results for trajectories 5 and 11, where the real trajectory of the joint is the blue line, and the estimated trajectory is in red.

TABLE 7.3: Mean and standard deviation (expressed in degrees) computed for the trajectory RMSE values depicted in Tables 7.1 and 7.2.

joint(#)	Mean(°)	*Std.* (°)
q_1	1.4441	1.2409
q_2	0.9270	0.9659
q_3	1.1542	0.6834
q_4	4.1467	2.5694
q_5	3.0573	1.8139
q_6	3.6195	2.4143
q_7	3.0752	1.7520

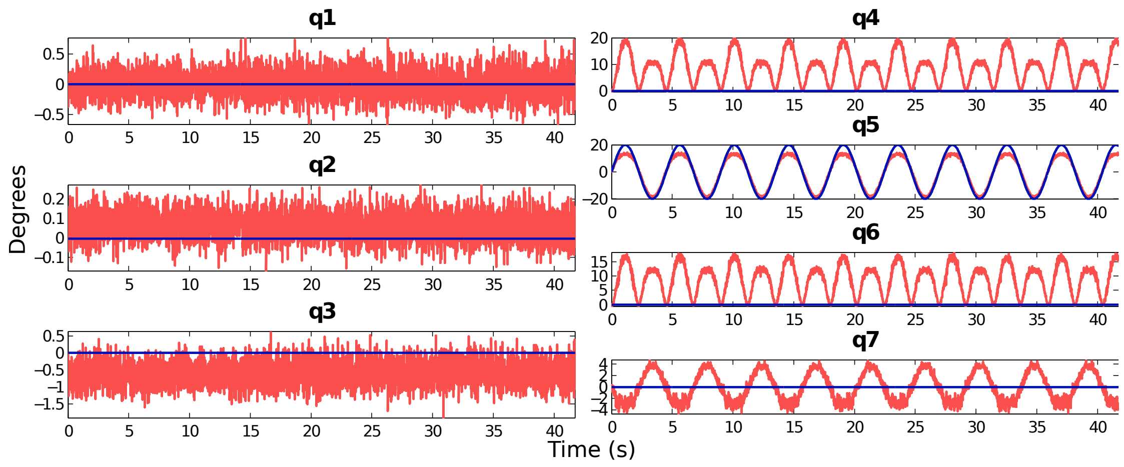


FIGURE 7.6: Trajectory 5. Blue line represents the real trajectory, while the estimated trajectory is in red.

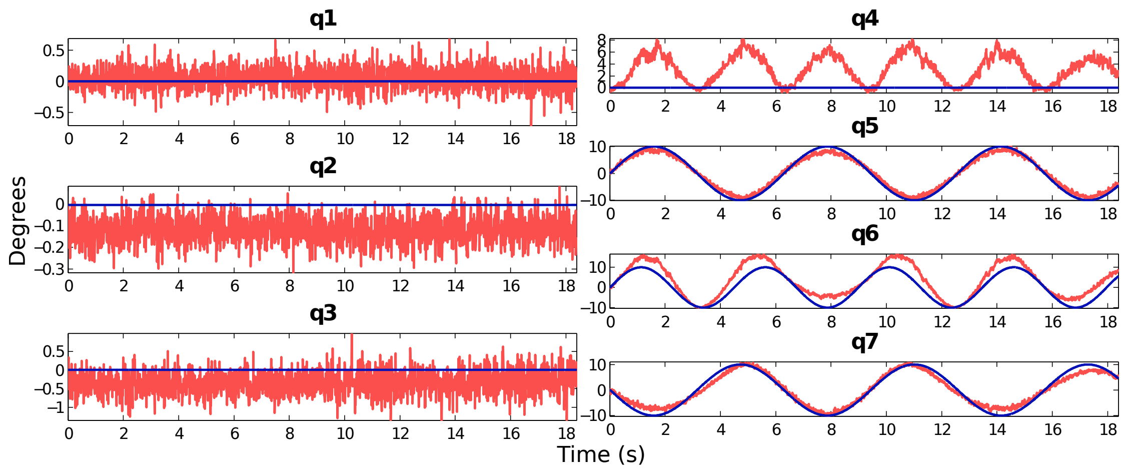


FIGURE 7.7: Trajectory 11. Blue line represents the real trajectory, while the estimated trajectory is in red.

These results are comparable to the results presented in [4]. Overall, RMSE are less than 4 degrees, reached only in the case of joint 4, which represents the most challenging one to estimate. We are currently investigating the reason, however we strongly believe that such errors on the estimates are caused by an inherent difficulty in placing the markers due to the particular shape of the robot parts. In particular, the last two markers are very difficult to place precisely and results are obviously affected. Also, the errors are higher when the robot is far from the zero position, *i.e.*, all joint values are 0.

We used a different tracking algorithm, based on markers, and a different robot, but results are comparable to the results we achieved in our previous work. This proves that our framework is general enough to be used with different tracking algorithms and on different robots, which was a quality we looked for when designing the framework to be functionally modular.

7.4.2.2 Using Estimates as Feedback in a Kinematic Controller

The second set of experiments had the goal to study the use of such vision-based estimates as feedback to control the robot to perform a kinematic task. In particular, the robot was asked to follow trajectories devised directly in the joint space.

Difficulty of placement for markers 3 and 4 and the relative errors induced on the estimations of joints 4, 5, 6 and 7, and the propagation of the errors in estimating the first joints make the control of the remaining axes of the robot very challenging. Thus we will first focus on the first 2 markers, thus limiting the motion of the robot to the first 3 joints, as proof of concept.

The robot performed 6 trajectories: the first 3 trajectories excited each joint singularly, *i.e.*, trajectory 1 uses only joint 1, and so on. The other 3 trajectories presented simultaneous motions for all the 3 joints. Table 7.4 reports errors between the estimates used in the controller and the ground truth values as read from the encoders. Table 7.5 presents task errors computed as difference between reference positions and ground truth values as read from the encoders. Notice that the controller did not use these differences to control the robot, but rather the difference between estimates and reference positions. Fig. 7.8 shows the evolution of estimates (red), ground truth values (blue) and references (green) during the performance of trajectories 4 and 5.

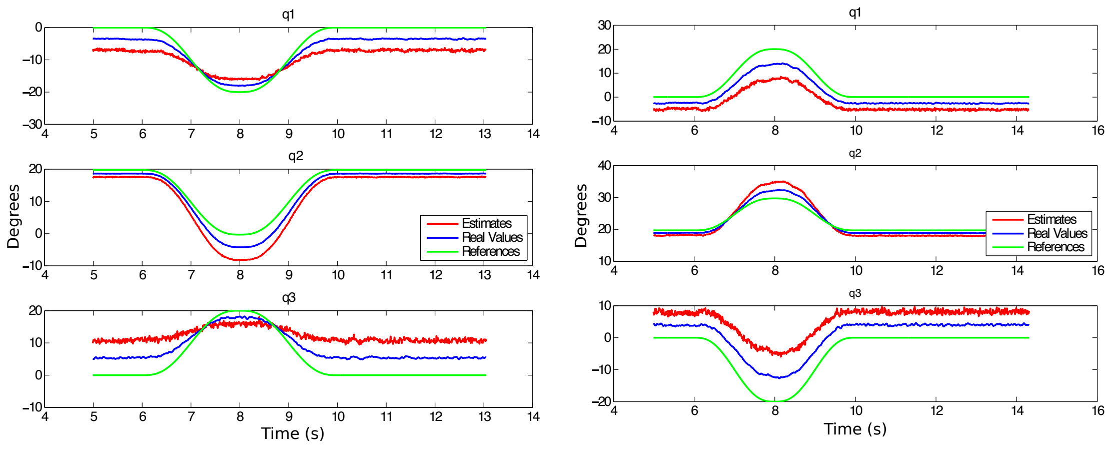
Table 7.4: RMSE of estimates with respect to ground truth values. Errors are expressed in degrees.

joint(#)	RMSE (in degrees (°))					
	traj 1	traj 2	traj 3	traj 4	traj 5	traj 6
q_1	2.1895	2.3207	2.6822	2.9667	3.6575	2.7492
q_2	2.6638	0.8453	0.7577	2.0913	1.1689	1.6167
q_3	4.2416	3.2690	4.5955	4.5042	5.0706	3.5712

Results show that estimates can be used to control the robot and that task errors induced by the errors in the estimates are of the same magnitude of the estimation errors. This

Table 7.5: RMSE errors of real joint values and reference motion. Errors are expressed in degrees.

joint(#)	RMSE (in degrees (°))					
	traj 1	traj 2	traj 3	traj 4	traj 5	traj 6
q_1	2.1665	2.3003	2.6762	2.9561	3.6405	2.7670
q_2	2.6606	0.8421	0.7547	2.1028	1.1752	1.7797
q_3	4.1938	3.2269	4.5817	4.4774	5.0540	3.5266



(a) Trajectory 4. (b) Trajectory 5.

Figure 7.8: These figures show the evolution of the estimates (red), real values (blue) and reference (green) for each experiment.

is clear comparing Table 7.4 and Table 7.5. Also, task errors do not go to zero because of the errors in the estimations. This is to be expected because, although the kinematic controller described in Subsec. 7.1.3 guarantees convergence, we are using estimates of the state, thus convergence to zero is not guaranteed anymore.

However, we also ran experiments controlling all the joints. We report the results in Tables 7.6 and 7.7. Fig. 7.9 reports graphically estimates, ground truth values and references of trajectory 1. As clear from Tables 7.6 and 7.7, the errors of estimation and task regarding the last 4 joints are higher than the first 3 joints, but errors never are higher than 8 degrees.

7.4.3 Contact Detection

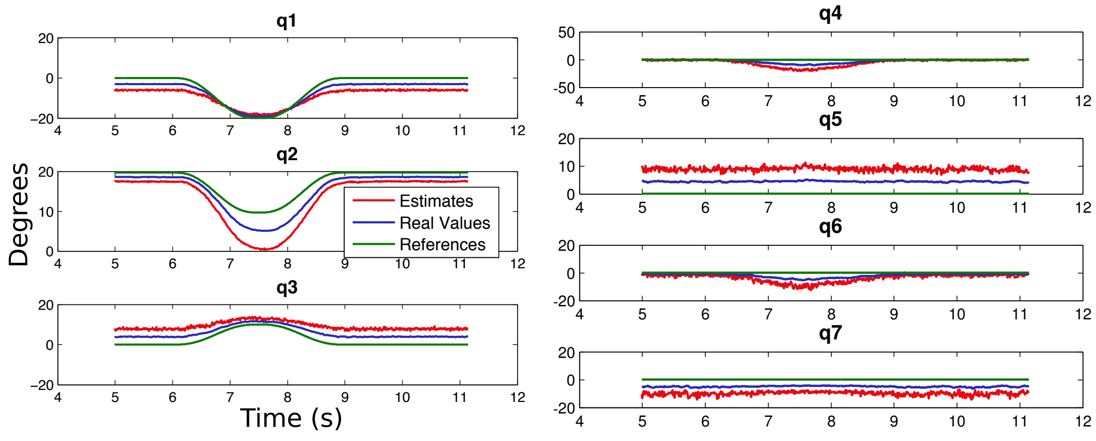
We described our contact detection methods in Sec. 7.2. During the experiments, we detect a contact only when both methods in Subsec. 7.2.1 and Subsec. 7.2.2 detect a contact. This has been done in order to achieve a higher robustness. In the following, we show how these techniques perform when readings from encoders are available. We analyse these results in order to provide the reader with a benchmark for later results.

Table 7.6: RMSE errors of estimates with respect to ground truth values. Errors are expressed in degrees.

joint(#)	RMSE (in degrees (°))		
	traj 1	traj 2	traj 3
q_1	2.5402	2.0437	2.5011
q_2	2.3265	1.9597	2.2247
q_3	3.4943	3.3668	3.8382
q_4	4.0870	4.1576	7.4786
q_5	4.5054	3.4631	6.6244
q_6	2.4200	3.3296	5.2717
q_7	5.0569	3.6256	4.9288

Table 7.7: RMSE errors of real joint values and reference motion. Errors are expressed in degrees.

joint(#)	RMSE (in degrees (°))		
	traj 1	traj 2	traj 3
q_1	2.5383	2.0509	2.5088
q_2	2.3439	1.9861	2.2505
q_3	3.4641	3.3279	3.7986
q_4	4.0082	4.0636	7.4306
q_5	4.4579	3.4045	6.5923
q_6	2.2266	3.2488	5.2017
q_7	4.9333	3.4119	4.8382



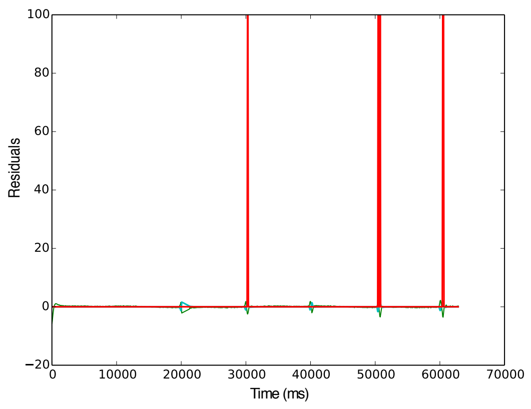
(a) (b)

Figure 7.9: These figures show the evolution of the estimates (red), real values (blue) and reference (green) while performing trajectory 1.

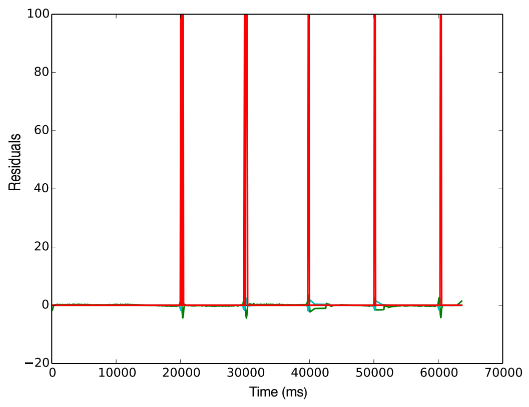
Then we rely on the vision-based estimates of the joint configuration to control the robot and simultaneously detect contacts. This sequence of experiments has been conceived to show the impact of the errors in the estimates of joint configurations on the contact detection quality and to understand how these detections can be used to estimate the kinematic constraints due to contacts.

7.4.3.1 With encoder readings

We applied our methods to detect contacts to situations where encoder readings are available. Particularly, we pushed the robot for 5 times every 10 seconds starting from 20 seconds in each experiment. In other words, every experiment is 1 minute long, and there are 5 contacts every 10 seconds, *i.e.*, at 20 s, 30 s, 40 s, 50 s and 60 s. We will focus our analysis on the number of detected contacts, stressing the number of false positives and missed detections. Table 7.8 shows the results of the 5 trials. False positives detected in those 5 trials are due to the bounces created by the pushes to generate the contacts. Fig. 7.10 reports the results of trial 2 and trial 4. They show when a contact has been missed, as in Subfig. 7.10(a) at 20 s and at 40 s, and also those bounces that create false positives.



(a) Trial number 2.



(b) Trial number 4.

Figure 7.10: Contact Detection. In Subfig. 7.10(a) two contacts were not detected (at 20 s and 40 s), while the contact at 50 s presents a multiple detection due to vibration of the robot - due to the push. In Subfig. 7.10(b) all contacts were detected, but multiple detections are present due to vibration of the robot - due to the pushes.

Table 7.8: Summary of results. This table shows the results of the 5 trials highlighting the number of detected contacts and the number of false detections.

Trial	Detected Contacts	False positives
1	5/5	3
2	3/5	0
3	5/5	1
4	5/5	0
5	4/5	4

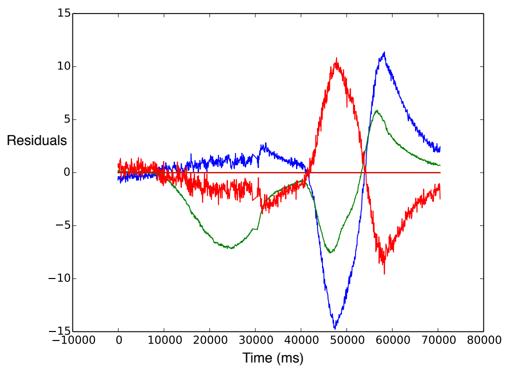
7.4.3.2 With vision-based estimates used in controller

For reasons similar to the ones adduced in Subsec. 7.4.2.2, we restricted the robot to be a 3DOF manipulator for this set of experiments. Results of experiments where the robot was asked to perform a trajectory with $\Delta q_1 = -20$, $\Delta q_2 = -30$, and $\Delta q_3 = 10$ are reported in Fig. 7.11(a). In particular, the residuals are non-zero because of the error in estimating the configuration. We analysed this to set the threshold of the norm of the residuals to detect contacts. Then, we forced software-induced joint brakes to emulate contacts, *i.e.*, these brakes are meant to emulate the result of hitting an object and preventing the robot to move the joint further. In detail, we set a brake to -5 degrees for joint 1 and whenever the robot was commanded to go to a configuration minor than that, the robot stayed in the -5 position. The values for joint 2 and 3 were respectively 15 and 2. Figs. 7.11(b), 7.11(c) and 7.11(d) report the results of performing the same trajectory while enabling those brakes one joint at a time. Every graph reports the residuals and a red line has 0 value when no contact is detected and climbs to 100 when a contact is detected. When the brake was applied to joints 1 or 2, a contact was detected, as expected. Fig. 7.12 reports results of experiment with brake on joint 2. The brake in the joint 3 did not trigger any detection however. This is due to the trajectory displacement being not too important and thus causing the relative residual to not increase enough.

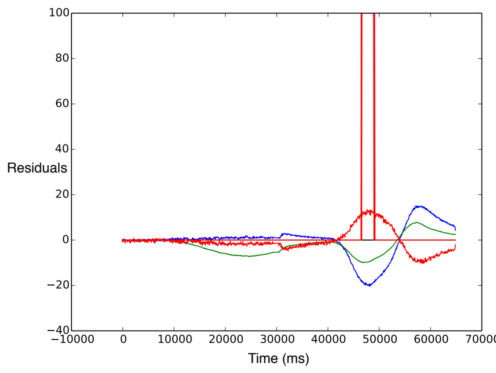
In Subsubsec. 7.2.1.3 we introduced the revised model for the residuals using estimates and said how critical thresholds are in order to correctly detect contacts. Results show that setting the thresholds is particularly hard and it has to depend on: 1) the trajectory and 2) the accuracy of the estimates. It has to be noticed that a high threshold would add tolerance to errors in the estimates, but it would make the system less sensitive to the real contacts. On the other hand, low thresholds increase the number of false positives due to the errors in the estimations. A trade-off is complicated but critical to find.

7.4.4 Contact Constraint Estimation

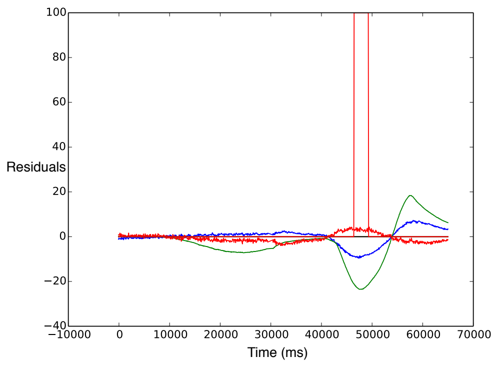
We continue the line of experiments described in the previous subsection performing an exploration after detecting a contact. This exploration has been defined directly in the joint space and has the goal of estimating the constraint(s) arising from contacts with the environment or brakes such as joint limits. The method used for this estimation has been already detailed in Sec. 7.3.



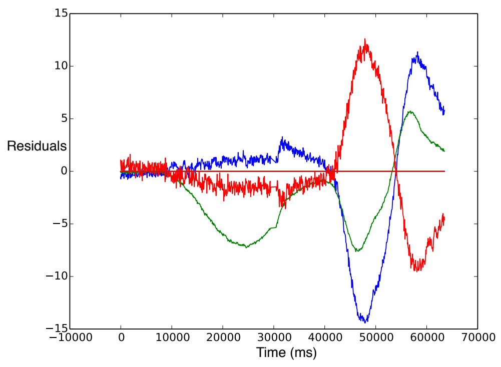
(a) No contact.



(b) Brake on joint 1.



(c) Brake on joint 2.

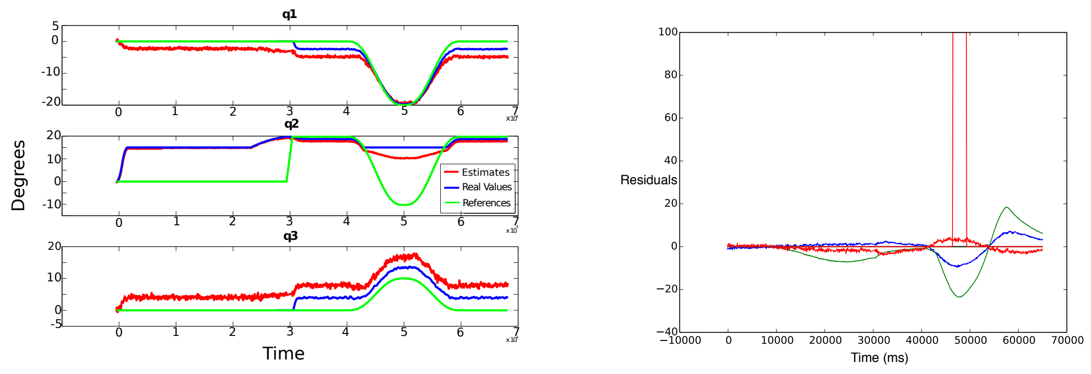


(d) Brake on joint 3.

FIGURE 7.11: Contact Detection. Fig. 7.11(a) reports the residuals while performing the trajectory in presence of no brake. Residuals are non-zero because of imprecision of estimates. Figs. 7.11(b), 7.11(c) and 7.11(d) report the results of the performance of the same trajectory but applying brakes respectively to joint 1, 2 and 3. Residuals are reported in multiple colours and a red line represents the detection of contacts: when 0 no contact is detected, whilst when 100, a contact is detected. In detail, when the brake was applied to joints 1 or 2, the brake caused a contact detection, as expected. The brake in the joint 3 did not trigger any detection however.

We were interested in investigating the behaviour of our framework in presence of bilateral and unilateral constraints. Thus we conducted two sets of experiments. We replicated the brakes introduced in the previous section to emulate unilateral constraints, *e.g.*, when the brake was on joint 1, the robot was not able to go to less than -5 degrees but it could go to higher values. In another set of experiments, brakes were hard constraints thus the robot was not permitted to move from the blocked configuration, *e.g.*, the brake was set to -5 degrees for joint 1 and joint 1 was not moved at all. This was to emulate bilateral constraints.

In particular, following the results described in the previous section, three experiments were run to estimate unilateral constraints and three for bilateral constraints. In the first experiment of each set, joint 1 was limited to be more than or equal to -5 degrees. In the second experiment joint 2 was blocked to be more than or equal to 15 degrees. And



(a) Joint configuration evolution

(b) Residuals and contact detection.

Figure 7.12: Contact Detection: brake on joint 2. Once the difference between the desired position and the estimated position is big enough for the residuals to be higher than the set thresholds, a contact is detected. Controller starts at t = 3 s (3000 ms in the graphs). Brake can be seen as blue line of joint 2 never goes below 15 degrees after controller has started (software brake is introduced after the controller is active and the trajectory is being performed). Contact is detected around 4.6 s, just after the brake is preventing the robot to perform desired motion.

finally, in the third experiment both brakes were enabled at the same time. This was done to investigate the behaviour of our method in presence of multiple simultaneous constraints.

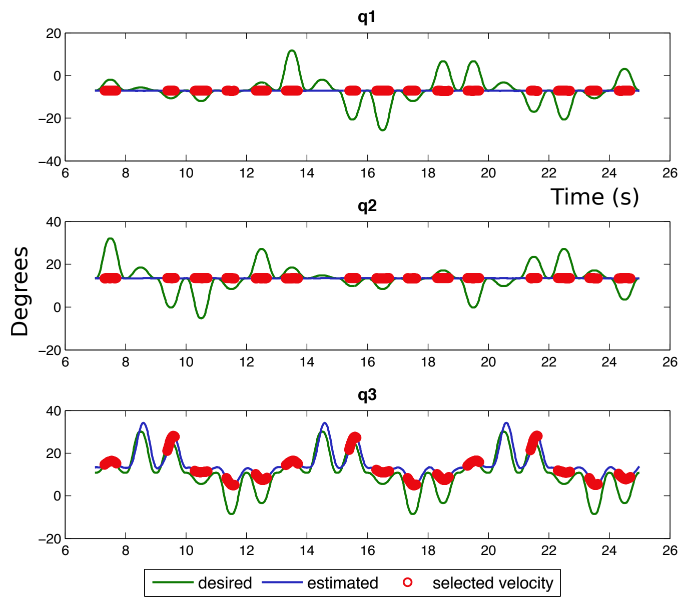
Velocity sets were recorded storing velocity vectors associated to high positional error and analysed offline in Matlab. The selection criterion of such velocity vectors relies on the assumption that if there is an important positional error this must be due to a contact. We first report the case of bilateral constraints and then we proceed with the results of the experiments on unilateral constraints.

7.4.4.1 Bilateral constraints

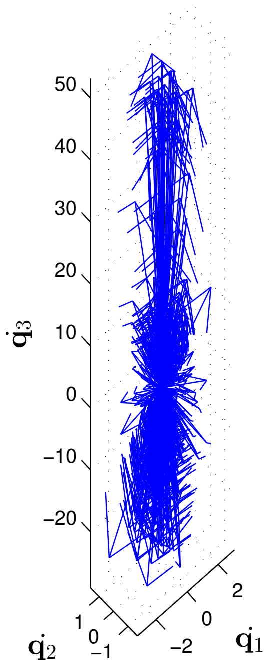
Bilateral constraints fully constrain motion, *i.e.*, motion in a specific direction is not admissible. These are very special constraints happening in specific situations, like a train following the railways. Table 7.9 reports the estimated $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$ (Eq. 7.31) and the real $\mathbf{\Lambda}^q(\mathbf{q})$ in all three experiments. Fig. 7.13 reports the set of joint velocities used to compute the estimate of the constraint in experiment 3. Results show that our method succeeds in estimating the kinematic constraint to a high degree of precision. Moreover, the sign inversion between $\mathbf{\Lambda}^q(\mathbf{q})$ and $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$ does not have any influence in this case, since motion is blocked in the entire direction ($\mathbf{\Lambda}^q(\mathbf{q})\dot{\mathbf{q}} = 0$), *i.e.*, $[0, 1, 0]$ and $[0.2593, -0.9658, -0.0030]$ both represent the same constraint. This is not true in the case of unilateral constraints.

TABLE 7.9: Summary of results. This table shows the real constraint $\mathbf{\Lambda}^q(\mathbf{q})$ and the estimated constraint $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$.

Experiment	$\mathbf{\Lambda}^q(\mathbf{q})$	$\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$
1	[1,0,0]	[0.9867, 0.1622, -0.0105]
2	[0,1,0]	[0.2593, -0.9658, -0.0030]
3	$\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$	$\begin{bmatrix}-0.9991 & -0.0414 & 0.0006\\ -0.0414 & 0.9991 & -0.0003\end{bmatrix}$



(a) Joint configuration evolution.



(b) Velocity set (degrees/s).

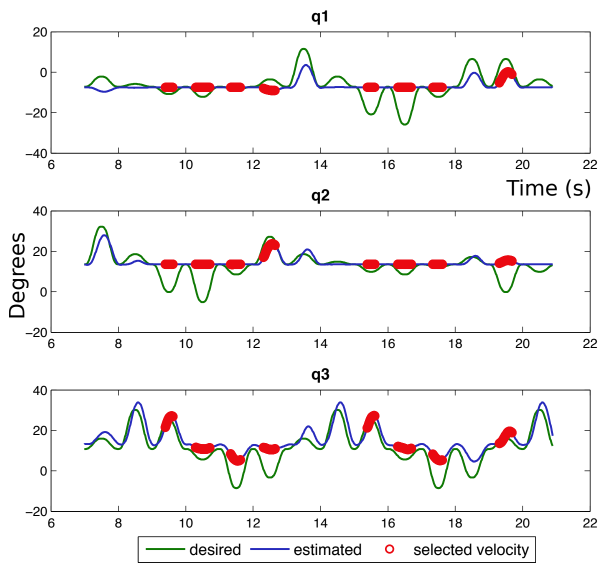
FIGURE 7.13: (a) Evolution of joint configuration and desired configurations during the exploration. Red circles denote when the configuration error between desired and estimated configurations is higher than a threshold thus regarded as due to a contact. (b) Illustrates the relative velocity set built selecting the estimated velocities when configuration error is higher than the threshold. Since the constraints are bilateral, the set of velocity has components only in the $\dot{q}_3$ direction of the graph. Small components in the other directions are due to configuration estimation errors.

7.4.4.2 Unilateral constraints

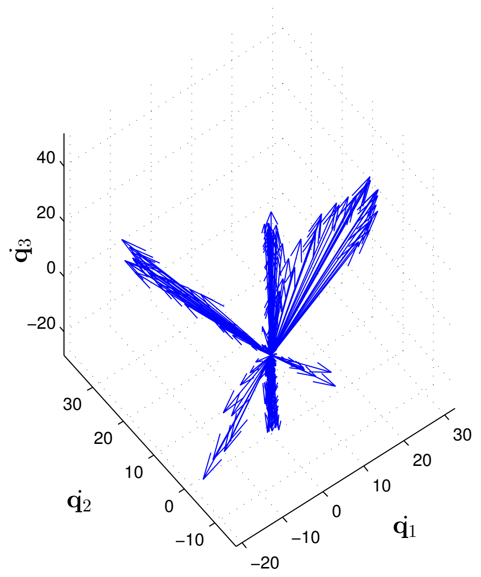
Unilateral constraints represent the vast majority of constraints due to contacts in everyday life tasks such walking and touching. In these situations, contacts limit motion only towards the contacts themselves, but the robot is able to leave the contacts and get back to free motion. Table 7.10 reports the estimated $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$ (Eq. 7.31). Fig. 7.14 reports the set of joint velocities used to compute the estimate of the constraint in experiment 3. Results show that our method succeeds in estimating the kinematic constraint to a high degree of precision, although in the case of experiment 1 the sign is reversed. A check on the sign would ensure that $\mathbf{\Lambda}^q(\mathbf{q})$ and $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$ have the same sign ($\mathbf{\Lambda}^q(\mathbf{q})\dot{\mathbf{q}} \geq 0$). However due to the errors on the estimates, velocities might have components also in the prohibited direction and a simple check is not enough. Future work includes a more accurate way to test the sign of the constraints.

TABLE 7.10: Summary of results. This table shows the real constraint $\mathbf{\Lambda}^q(\mathbf{q})$ and the estimated constraint $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$. In experiment 1, the sign is reversed. In experiment 3, $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$ spans the same space as $\mathbf{\Lambda}^q(\mathbf{q})$.

Experiment	$\mathbf{\Lambda}^q(\mathbf{q})$	$\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$
1	[1,0,0]	[-0.9922, -0.1225, -0.0233]
2	[0,1,0]	[-0.1558, 0.9853, 0.0705]
3	$\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\end{bmatrix}$	$\begin{bmatrix}0.1039 & 0.9945 & -0.0128\\0.9752 & -0.1044 & -0.1949\end{bmatrix}$



(a) Joint configuration evolution.



(b) Velocity set (degrees/s).

FIGURE 7.14: (a) Illustrates the evolution of joint configuration and desired configurations during the exploration. Red circles denote when the configuration error between desired and estimated configurations is higher than a threshold thus regarded as due to a contact. (b) Shows the relative velocity set built selecting the estimated velocities when configuration error is higher than the threshold. Since the constraints are only unilateral, the set of velocity has components not only in the $\dot{q}_3$ direction of the graph, as in the case of Fig. 7.13(b). However notice that in the $\dot{q}_1$ and $\dot{q}_2$ directions, only positive velocities are present. This is due to the fact that during the exploration, a constraint can be active while the other might not, and vice versa.

7.5 Conclusion

We presented a general control framework which integrates all the methods and techniques presented in the previous Chapters. In detail, we used the vision-based framework presented Chapter 6 to estimate a robot configuration, and we applied it to a different robot using a different visual tracking algorithm. This showed how the modularity of such framework allows the user to change its modules with straightforward ease. We added a contact detection technique so as to trigger the procedure to estimate the kinematic constraints due to contacts. Such procedure was proposed already in Chapter 5 but here we devised the exploration directly in the joint space, thus estimating the configuration-dependent $\mathbf{\Lambda}^q(\mathbf{q})$.

Results show that the general framework can indeed be used to estimate the configuration of a robot and these estimates can be used to control the robot (Subsec. 7.4.2). Subsec. 7.4.3 shows that those same estimates can be used to detect contacts using the kinematic residuals proposed in Subsec. 7.2.1. Finally, Subsec. 7.4.4 presented the promising results of experiments where a contact detection triggered the procedure to estimate the relative kinematic constraints.

We believe it is possible to achieve more precise results in estimating the joint configuration using more sophisticated tracking methods. However, depth information and a GPU for computation might be needed. Also, the iiwa is challenging to track because of its shape and colour. This also influenced the estimation results in the sense that the markers position was hard to set and hard to relate to the robot reference frames. We strongly believe that this has affected negatively our results.

It would be also possible to understand which joints are involved in a contact by examining the residuals. This property was already discussed in [149]. However, in our case residuals depend also on the error on the estimates, thus this has to be taken into account as well.

Current and future work includes the use of the whole body of the robot for further experiments on contact detection and constraint estimation. Also, velocity sets will be analysed online and constraints will be estimated in realtime, so that the robot will be able to resume the task and project it when needed. Finally, we intend to apply this framework to control the robot in dynamics.

Chapter 8

Conclusion

This Thesis has explored the problems of controlling robots which, firstly, lack proprioceptive sensors and, secondly, interact forcefully with their surroundings. A particular motivation for this work has been the prevalence of such problems in the use of robots for nuclear decommissioning.

For these reasons, this Thesis has developed and evaluated possible solutions to several key questions: how to estimate a robot's configuration using a remote vision system, robustly in real time; how to use this visually acquired information as feedback to control the robot; how to detect and learn contact constraints which are imposed by the robot's environment; how to control a robot when it is subjected to such contact constraints; furthermore, how to actually exploit such contact constraints, in order to reduce the torque effort required at the robot's joints, thus performing the task more efficiently.

Chapter 4 describes a control framework based on projected inverse dynamics which enables an exploitation of the constraints due to contacts to effectively reduce commanded torques during tasks. We analyse contact usage and investigate techniques to enable robots to mimic some useful human strategies for exploiting such contacts. In order to replicate or at least understand this exploitation of the contacts, dynamics play a dominant role, not only because of the involvement of force and torque exchanges, but also because the robot might need to perform tasks with certain velocities and accelerations. Specifically, we use a controller based on projected dynamics and we formulate tasks both in the joint space and in the Cartesian space. Experiments with a KUKA LWR IV arm tasked with wiping a board are reported to corroborate such results.

However, in Chapter 4, we assume to have a priori knowledge on the constraints arising from the contacts with the environment. Chapter 5 presents methods to overcome this drawback, and to estimate locally the kinematic constraints due to contacts with the environment. Such methods do not need force/torque sensors at contact points, but rather use only kinematic quantities.

Chapter 6 proposes a framework to estimate the configuration of a manipulator using a single monocular camera. In particular, a modularised framework is presented which is formed of three components: visual tracking, state estimation and robot controller. Experiments on a KUKA KR5 sixx show how the proposed framework is able to estimate the configuration of the robot and how these estimates can be used to control the robot to perform tasks in its workspace.

Finally, Chapter 7 describes a general control framework which uses all of the above contributions to servo a manipulator. Single monocular camera images are used to estimate the configuration of a KUKA iiwa 14 R820 manipulator. We use a different tracking algorithm with respect to the one in Chapter 6 to show how the modularity of the architecture gives freedom in the choice of each component. Those estimates are used to control kinematically the robot. Furthermore, whenever a contact is detected, the robot performs an exploration in order to estimate the kinematic constraints due to the contact. We show results of experiments with the KUKA iiwa 14 R820 in both estimating the robot's configuration and also in estimating the kinematic constraints.

In addition to directions already described in each Chapter, future work includes the use of a more reliable tracking algorithm in the estimation framework, *e.g.*, a tracking algorithm using also depth data. This should increase the precision of such estimates and improve the overall performance of the control framework. Moreover, a smarter exploration can reduce time and increase precision of the constraint estimation methods, also needed to enhance the behaviour of the control framework.

Chapter 9

Publications

1. G. A. Hollinger, A. Pereira, V. Ortenzi and G. Sukhatme: Towards Improved Prediction of Ocean Processes Using Statistical Machine Learning. In Robotics: Science and Systems, Workshop on Robotics for Environmental Monitoring , Sydney, Australia, Jul 2012.

2. V. Ortenzi, M. Adjigble, J. A. Kuo, R. Stolkin, M Mistry: An experimental study of robot control during environmental contacts based on projected operational space dynamics. Proceeding of IEEE-RAS International Conference on Humanoid Robots, Madrid, 2014.

3. V. Ortenzi, S. Tarantino, C. Castellini, C. Cipriani: Ultrasound imaging for hand prosthesis control: a comparative study of features and classification methods. In IEEE International Conference on Rehabilitation Robotics (ICORR), Singapore, 2015.

4. V. Ortenzi, R. Stolkin, J. A. Kuo, M. Mistry: Projected inverse dynamics control and optimal control for robots in contact with the environment: A comparison. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, 2015.

5. N. Marturi, V. Ortenzi, J. Xiao, M. Adjigble, R. Stolkin, A. Leonardis: A real-time tracking and optimised gaze control for a redundant humanoid robot head. In 15th IEEE-RAS International Conference on Humanoid Robots, Humanoids, Seoul, 2015.

6. V. Ortenzi, N. Marturi, R. Stolkin, J. A. Kuo, M Mistry: Vision-guided state estimation and control of robotic manipulators which lack proprioceptive sensors. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016.

7. V. Ortenzi, H. Lin, M. Azad, R. Stolkin, J. A. Kuo, M. Mistry: Kinematic-based Estimation of Contact Constraints using only Proprioception. In 2016 IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2016.

8. M. Azad, V. Ortenzi, H. Lin, E. Rueckert, M. Mistry: Parameter Estimation and Control of Complaint Contact Normal Force. In 2016 IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2016.

Bibliography

[1] V. Ortenzi, M. Adjigble, J. Kuo, R. Stolkin, and M. Mistry. An experimental study of robot control during environmental contacts based on projected operational space dynamics. In *14th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 407 – 412. IEEE, 2014.

[2] Valerio Ortenzi, Rustam Stolkin, Jeffrey A. Kuo, and Michael Mistry. Projected inverse dynamics control and optimal control for robots in contact with the environment: A comparison. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[3] V. Ortenzi, H-C. Lin, M. Azad, R. Stolkin, J.A. Kuo, and M Mistry. Kinematics-Based Estimation of Contact Constraints Using Only Proprioception. In *Humanoids 2016: IEEE-RAS International Conference on Humanoid Robots*, 2016.

[4] Valerio Ortenzi, Naresh Marturi, Rustam Stolkin, Jeffrey A. Kuo, and Michael Mistry. Vision-guided state estimation and control of robotic manipulators which lack proprioceptive sensors. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[5] O. Khatib. The operational space formulation in the analysis, design, and control of manipulators. In O. Faugeras and G. Giralt, editors, *Proc. of the International Symposium on Robotics Research*, pages 261–270, Gouvieux-Chantilly, France, December 1986. The MIT Press, Cambridge, MA, USA.

[6] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal of Robotics and Automation*, RA-3(1):43–53, February 1987.

[7] Michael Mistry and Ludovic Righetti. Operational space control of constrained and underactuated systems. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.

[8] Farhad Aghili. A unified approach for inverse and direct dynamics of constrained multibody systems based on linear projection operator: applications to control and simulation. *IEEE Transactions on Robotics*, 21(5):834–849, 2005.

[9] David Orin Roy Featherstone. Robot dynamics: Equations and algorithms. In *IEEE Int. Conf. Robotics and Automation*, pages 826–834, 2000.

[10] Bruno Siciliano and Oussama Khatib, editors. *Springer Handbook of Robotics*. Springer, 2008.

[11] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 1st edition, 2008. ISBN 1846286417, 9781846286414.

[12] P.C. Watson. Remote center compliance system, 1978.

[13] S Nicosia and P Tomei. Robot control by using only joint position measurements. *Automatic Control, IEEE Transactions on*, 35(9):1058–1061, 1990.

[14] P. Hsu, J. Hauser, and S. Shankar. Dynamic control of redundant manipulators. In *Proceedings of IEEE Robotics and Automation*, 1988.

[15] Torsten Kröger, Bernd Finkemeyer, Markus Heuck, and Friedrich M. Wahl. Adaptive implicit hybrid force/pose control of industrial manipulators: compliant motion experiments. In *IROS*, pages 816–821. IEEE, 2004.

[16] M.T. Mason. Compliance and force control for computer controlled manipulators. *Systems, Man and Cybernetics, IEEE Transactions on*, (6):418–432, 1981. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4308708`.

[17] Alessandro De Luca and Costanzo Manes. Modeling of robots in contact with a dynamic environment. *IEEE T. Robotics and Automation*, 10(4):542–548, 1994.

[18] J. J. Craig. *Introduction to Robotics: Mechanics and Control (2nd ed.)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[19] S. Chiaverini, B. Siciliano, and L. Villani. A Survey of Robot Interaction Control Schemes with Experimental Comparison. *IEEE Robotics & Automation Magazine*, 4(3):273–285, 1999. ISSN 10834435. doi: 10.1109/3516.789685. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=789685`.

[20] J. De Schutter, H. Bruyninckx, W. Zhu, and M.W. Spong. Force control: a bird's eye view. In *IEEE CSS/RAS International Workshop on Control Problems in Robotics and Automation: Future Directions*, number December, pages 1–14, 1997. ISBN 0170-8643. URL `http://link.springer.com/chapter/10.1007/BFb0015073`.

[21] R. P. Paul and B. Shimano. Compliance and Control. In *Joint Automatic Control Conference*, pages 1694–1699, 1976.

[22] D.E. Whitney. Force feedback control of manipulator fine motions. *Journal of Dynamic Systems, Measurement, and Control*, 1977. URL `http://dynamicsystems.asmedigitalcollection.asme.org/article.aspx?articleid=1402627`.

[23] N Hogan. Impedance control: An approach to manipulation. *American Control Conference, 1984*, (March), 1984. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4788393`.

[24] J.K. Salisbury. Active Stiffness Control of a Manipulator in Cartesian Coordinates. *19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, 1980. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4046624`.

[25] M. T. Mason. Compliance and force control for computer controlled manipulators. Master's thesis, MIT, 1979.

[26] J.J. Craig and M.H. Raibert. A systematic method of hybrid position/force control of a manipulator. *Computer Software and Applications Conference, 1979. Proceedings. COMPSAC 79. The IEEE Computer Society's Third International*, pages 446–451, 1979. URL `http://www.researchgate.net/publication/3798567_A_systematic_method_of_hybrid_positionforce_control_of_a_manipulator`.

[27] M.H. Raibert and J.J. Craig. Hybrid Position / Force Control of Manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 102(June 1981), 1981. URL `http://dynamicsystems.asmedigitalcollection.asme.org/article.aspx?articleid=1403159`.

[28] H. Zhang and R.P. Paul. Hybrid Control of Robot Manipulators. *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, pages 602–607, 1985. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1087304`.

[29] C.H. An and J.M. Hollerbach. Kinematic stability issues in force control of manipulators. *Proceedings of IEEE International Conference on Robotics and Automation*, 1987. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1087933`.

[30] H. Zhang. Kinematic stability of robot manipulators under force control. *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pages 80–85, 1989. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=99971`.

[31] W. D. Fisher and M. S. Mujtaba. Hybrid position/force control: a correct formulation. *International Journal of Robotics Research*, 11(4):299 – 311, 1992.

[32] N.H. McClamroch and D. Wang. Feedback stabilization and tracking of constrained robots. *Automatic Control, IEEE Transactions on*, 33(5), 1988. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1220`.

[33] T Yabuta. Nonlinear Basic Stability Concept of the Hybrid Position/Force Control Scheme for Robot Manipulators. *Robotics and Automation, IEEE Transactions on*, (0), 1992. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=163791`.

[34] T Yoshikawa. Dynamic Hybrid Position/Force Control of Robot Manipulators - Description of Hand Constraints and Calculation of Joint Driving Force. *Robotics and Automation, IEEE Journal of*, 11:386–392, 1987. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1087120`.

[35] H. West and H. Asada. A method for the design of hybrid position/force controllers for manipulators constrained by contact with the environment. *Proceedings of IEEE International Conference on Robotics and Automation*, 2(5):251 – 259, 1985.

[36] T. Yoshikawa, T. Sugie, and M. Tanaka. Dynamic Hybrid Position/Force Control of Robot Manipulators - Controller Design and Experiment. *Robotics and Automation, IEEE Journal of*, 4(6):699 – 705, 1988. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=9307`.

[37] T Yoshikawa and A Sudou. Dynamic Hybrid Position/Force Control of Robot Manipulators: On-Line Estimation of Unknown Constraint. *Robotics and Automation, IEEE Transactions on*, 1993. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=238286`.

[38] S. Chiaverini and L. Sciavicco. The Parallel Approach to Force/Position Control of Robotic Manipulators. *Robotics and Automation, IEEE Transactions on*, 9 (4), 1993. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=246048`.

[39] S Chiaverini, B. Siciliano, and L. Villani. Force/Position Regulation of Compliant Robot Manipulators. *Automatic Control, IEEE Transactions on*, (2), 1994. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=280780`.

[40] A. De Luca and C Manes. Modeling of robots in contact with a dynamic environment. *Robotics and Automation, IEEE Transactions on*, 1994. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=313104`.

[41] R. Featherstone, S. Sonck, and O. Khatib. A general contact model for dynamically-decoupled force/motion control. *Experimental Robotics V*, 1998. URL `http://link.springer.com/chapter/10.1007/BFb0112956`.

[42] F. Aghili and C. Su. Control of constrained robots subject to unilateral contacts and friction cone constraints. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2347 – 2352, Stockholm, May 2016.

[43] S. Hayati. Hybrid position/Force control of multi-arm cooperating robots. *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, 1986. doi: 10.1109/ROBOT.1986.1087650.

[44] T. Yoshikawa and X. Zheng. Coordinated dynamic hybrid position/force control for multiple robot manipulators handling one constrained object. *Proceedings., IEEE International Conference on Robotics and Automation*, pages 0–5, 1990. ISSN 1342-5668. doi: 10.1109/ROBOT.1990.126156.

[45] R.K. Kankaanranta and H.N. Koivo. Dynamics and simulation of compliant motion of a manipulator. *J. Robot. Automat.*, pages 163–173, 1988.

[46] R. Lozano and B. Brogliato. Adaptive hybrid force-position control for redundant manipulators. *29th IEEE Conference on Decision and Control*, 37(10):1501–1505, 1990. doi: 10.1109/CDC.1990.203962.

[47] T. Hsia. Adaptive control of robot manipulators - A review. *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, 1986. doi: 10.1109/ROBOT.1986.1087696.

[48] J. E. Slotine and W. Li. Adaptive manipulator control: A case study. *IEEE Transactions on Automatic Control*, 33(11):995–1003, 1988. ISSN 00189286. doi: 10.1109/9.14411. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=14411`.

[49] Suguru Arimoto, Yun-Hui Liu, and Tomohide Naniwa. Model-based adaptive hybrid control for geometrically constrained robots. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation, Atlanta, Georgia, USA, May 1993*, pages 618–623, 1993.

[50] C.C. Cheah, S. Kawamura, and S. Arimoto. Stability of hybrid position and force control for robotic manipulator with kinematics and dynamics uncertainties. *Automatica*, 39(5):847–855, May 2003. ISSN 00051098. doi: 10.1016/S0005-1098(03)00002-5. URL `http://linkinghub.elsevier.com/retrieve/pii/S0005109803000025`.

[51] E. Magrini, F. Flacco, and A. De Luca. Estimation of Contact Force using a Virtual Force Sensor. *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, (Iros):2126–2133, 2014. doi: 10.1109/IROS.2014.6942848.

[52] M. Anitescu and F.A. Potra. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics*, (93):1–21, 1997. URL `http://link.springer.com/article/10.1023/A:1008292328909`.

[53] Y. Tassa, T. Erez, and E. Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *IEEE International Conference on Intelligent Robots and Systems*, pages 4906–4913, 2012. ISBN 9781467317375. doi: 10.1109/IROS.2012.6386025.

[54] M. Posa and R. Tedrake. Direct trajectory optimization of rigid body dynamical systems through contact. In *Algorithmic Foundations of Robotics X - Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics, WAFR*, pages 527–542, 2012.

[55] M. Posa, C. Cantu, and R. Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2013. ISSN 0278-3649. doi: 10.1177/0278364913506757. URL `http://ijr.sagepub.com/content/33/1/69.short`.

[56] T. J. Tarn, Y. Wu, N. Xi, and A. Isidori. Force regulation and contact transition control. *IEEE Control Systems Magazine*, 16(1):32–40, 1996. ISSN 02721708. doi: 10.1109/37.482135.

[57] M.l Posa, M. Tobenkin, and R. Tedrake. Lyapunov analysis of rigid body systems with impacts and friction via sums-of-squares. In *Proceedings of the 16th international conference on Hybrid systems: computation and control - HSCC '13*, 2013. ISBN 9781450315678. doi: 10.1145/2461328.2461340.

[58] Steve Collins, Andy Ruina, Russ Tedrake, and Martijn Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science*, 307(5712):1082–1085, 2005.

[59] Lael U Odhner, Leif P Jentoft, Mark R Claffee, Nicholas Corson, Yaroslav Tenzer, Raymond R Ma, Martin Buehler, Robert Kohout, Robert D Howe, and Aaron M Dollar. A compliant, underactuated hand for robust manipulation. *The International Journal of Robotics Research*, 33(5):736–752, 2014.

[60] Paul J. Besl and Ramesh C. Jain. Invariant surface characteristics for 3d object recognition in range images. *Comput. Vision Graph. Image Process.*, 33(1):33–80, January 1986.

[61] Joydeep Biswas and Manuela Veloso . Depth camera based indoor mobile robot localization and navigation. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1697–1702. IEEE, May 2012.

[62] David Schiebener, Jun Morimoto, Tamim Asfour, and Ales Ude. Integrating visual perception and manipulation for autonomous learning of object representations. *Adaptive Behaviour*, 21(5):328–345, 2013.

[63] Kester Duncan, Sudeep Sarkar, Redwan Alqasemi, and Rajiv V. Dubey. Multi-scale superquadric fitting for efficient shape and pose recovery of unknown objects. In *2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, May 6-10, 2013*, pages 4238–4243, 2013.

[64] Antonio Bicchi, J. Kenneth Salisbury, and David L. Brock. Contact sensing from force measurements. *International Journal of Robotics Research*, 12:249–262, 1993.

[65] T. Tsujimura and T. Yabuta. Object detection by tactile sensing method employing force/torque information. *IEEE Transactions on Robotics and Automation*, 5: 444 – 450, 1989.

[66] R. S. Fearing and T. O. Binford. Using a cylindrical tactile sensor for determining curvature. *IEEE Transactions on Robotics and Automation*, 7:806 – 817, 1991.

[67] Robert D. Howe. Tactile sensing and control of robotic manipulation. *Journal of Advanced robotics*, 8(3):245–261, 1994.

[68] Joris De Schutter, Herman Bruyninckx, Stefan Dutré, Jan De Geeter, Jayantha Katupitiya, Sabine Demey, and Tine Lefebvre. Estimating first-order geometric parameters and monitoring contact transitions during force-controlled compliant motion. *I. J. Robotic Res.*, 18(12):1161–1184, 1999.

[69] Tine Lefebvre, Herman Bruyninckx, and Joris De Schutter. Polyhedral contact formation modeling and identification for autonomous compliant motion. *IEEE Trans. Robotics and Automation*, 19(1):26–41, 2003.

[70] Nicolas Sommer, Miao Li, and Aude Billard. Bimanual compliant tactile exploration for grasping unknown objects. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, pages 6400–6407, 2014.

[71] Mårten Björkman, Yasemin Bekiroglu, Virgile Hogman, and Danica Kragic. Enhancing visual perception of shape through tactile glances. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, November 3-7, 2013*, pages 3180–3186. IEEE, 2013.

[72] Claudio Zito, Marek Sewer Kopicki, Rustam Stolkin, Christoph Borst, Florian Schmidt, Máximo A. Roa, and Jeremy L. Wyatt. Sequential trajectory re-planning with tactile information gain for dexterous grasping under object-pose uncertainty. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4013–4020, 2013.

[73] Christian Gehring, C. Dario Bellicoso, Stelian Coros, Michael Blösch, Peter Fankhauser, Marco Hutter, and Roland Siegwart. Dynamic trotting on slopes for quadrupedal robots. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5129–5135, 2015.

[74] T Yoshikawa and A Sudou. Dynamic Hybrid Position/Force Control of Robot Manipulators: On-Line Estimation of Unknown Constraint. *Robotics and Automation, IEEE Transactions on*, 1990.

[75] Hsiu-Chin Lin, Matthew Howard, and Sethu Vijayakumar. Learning null space projections. In *ICRA*, pages 2613–2619. IEEE, 2015.

[76] Roger Y. Tsai. An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, FL*, pages 364–374, 1986.

[77] Danica Kragic and H. Christensen. Survey on Visual Servoing for Manipulation. Technical report, Centre for Autonomous Systems, Numerical Analysis and Computer Science,, 2002.

[78] Lee E. Weiss, Arthur C. Sanderson, and Charles P. Neuman. Dynamic sensor-based control of robots with visual feedback. *IEEE J. Rob. Autom.*, 3(5):404–417, 1987.

[79] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, 1992.

[80] Seth Hutchinson, Greg Hager, and Peter Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12:651–670, 1996.

[81] F. Chaumette and S. Hutchinson. Visual servo control, part i: Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90, December 2006.

[82] François Chaumette and S. Hutchinson. Visual servo control, Part II: Advanced approaches. *IEEE Robotics and Automation Magazine*, 14(1):109–118, 2007.

[83] Markus Vincze. Dynamics and system performance of visual servoing. In *IEEE Int. Conf. on Robotics and Automation*, pages 644–649. IEEE, 2000.

[84] William J. Wilson, Carol C. Williams Hulls, and G. S. Bell. Relative end-effector control using cartesian position based visual servoing. *IEEE T. Robotics and Automation*, 12(5):684–696, 1996.

[85] A. Castano and S. A. Hutchinson. Visual compliance: Task-directed visual servo control. *IEEE Trans. on Robotics and Automation*, 10(3):334–342, June 1994.

[86] Ezio Malis, François Chaumette, and Sylvie Boudet. 2-1/2-d visual servoing. *IEEE Trans. on Robotics and Automation*, 15:238–250, 1999.

[87] T. Drummond and R. Cipolla. Visual tracking and control using lie algebras. In *In Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 652–657. IEEE Computer Society Press, 1999.

[88] A. Fox and S. Hutchinson. Exploiting visual constraints in the synthesis of uncertainty-tolerant motion plans. *IEEE Trans. on Robotics and Automation*, 11:56–71, 1995.

[89] F. Chaumette, P. Rives, and B. Espiau. Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing. In *IEEE Int Conf on Robotics and Automation, ICRA'91*, volume 3, pages 2248–2253, Sacramento, California, April 1991.

[90] Warren E. Dixon, Erkan Zergeroglu, Yongchun Fang, and Darren M. Dawson. Object tracking by a robot manipulator: A robust cooperative visual servoing approach. In *ICRA*, pages 211–216. IEEE, 2002.

[91] Tej Dallej, Nicolas Andreff, and Philippe Martinet. Image-based visual servoing of the i4r parallel robot without proprioceptive sensors. In *ICRA*, pages 1709–1714. IEEE, 2007.

[92] Nicolas Andreff, Tej Dallej, and Philippe Martinet. Image-based visual servoing of a gough&stewart parallel manipulator using leg observations. *Int. J. Rob. Res.*, 26(7):677–687, July 2007. ISSN 0278-3649.

[93] Y. Hu, Roy Eagleson, and M. A. Goodale. Human visual servoing for reaching and grasping: The role of 3-d geometric features. In *ICRA*, pages 3209–3216. IEEE Robotics and Automation Society, 1999.

[94] Nikolaos P. Papanikolopoulos, Bradley J. Nelson, and Pradeep K. Khosla. Six degree-of-freedom hand/eye visual tracking with uncertain parameters. *IEEE T. Robotics and Automation*, 11(5):725–732, 1995.

[95] Andrew Comport, D. Kragic, E. Marchand, and François Chaumette. Robust real-time visual tracking: Comparison, theoretical analysis and performance evaluation. In *IEEE Int. Conf. on Robotics and Automation, ICRA'05*, pages 2852–2857, Barcelona, Spain, France, 2005. URL `http://hal.inria.fr/inria-00351885`.

[96] Eric Marchand, François Chaumette, Fabien Spindler, and Michel Perrier. Controlling an uninstrumented manipulator by visual servoing. *I. J. Robotic Res.*, 21 (7):635–648, 2002.

[97] Kevin Nickels and Seth Hutchinson. Model-based tracking of complex articulated objects. 17(1):28–36, 2001.

[98] E. Marchand and F. Chaumette. A new formulation for non-linear camera calibration using virtual visual servoing. Technical Report 1366, IRISA, January 2001.

[99] E. Marchand and F. Chaumette. Virtual visual servoing: A framework for real-time augmented reality. In G. Drettakis and H.-P. Seidel, editors, *EUROGRAPHICS 2002 Conference Proceeding*, volume 21(3) of *Computer Graphics Forum*, pages 289–298, September 2002.

[100] X. Gratal, J. Romero, and D. Kragic. Virtual visual servoing for real-time robot pose estimation. In *International Federation of Automatic Control World Congress, IFAC*, 2011.

[101] Xavi Gratal, Christian Smith, Mårten Björkman, and Danica Kragic. Integrating 3d features and virtual visual servoing for hand-eye and humanoid robot pose estimation. In *IEEE-RAS International Conference on Humanoid Robots*, pages 240–245, 2013.

[102] Muriel Pressigout and Éric Marchand. Model-free augmented reality by virtual visual servoing. In *ICPR (2)*, pages 887–890, 2004.

[103] Andrew I. Comport, Éric Marchand, Muriel Pressigout, and François Chaumette. Real-time markerless tracking for augmented reality: The virtual visual servoing framework. *IEEE Trans. Vis. Comput. Graph.*, 12(4):615–628, 2006.

[104] David G. Lowe. Local feature view clustering for 3d object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 682–688. Springer, 2001.

[105] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich. Viewpoint-aware object detection and continuous pose estimation. *Image and Vision Computing*, 2012.

[106] S. Savarese and L. Fei-Fei. 3d generic object categorization, localization and pose estimation. In *IEEE International Conference on Computer Vision.*, Rio de Janeiro, Brazil, October 2007.

[107] Min Sun, Gary Bradski, Bing-Xin Xu, and Silvio Savarese. Depth-encoded hough voting for joint object detection and shape recovery. In *Proceedings of the 11th European conference on Computer vision: Part V*, ECCV'10, pages 658–671. Springer-Verlag, 2010.

[108] Bastian Leibe, Ales Leonardis, and Bernt Schiele. Combined object categorization and segmentation with an implicit shape model. In *In ECCV workshop on statistical learning in computer vision*, pages 17–32, 2004.

[109] Y. Hel-Or and M. Werman. Absolute orientation from uncertain point data: a unified approach. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR '92., 1992 IEEE Computer Society Conference on*, pages 77–82, 1992. doi: 10.1109/CVPR.1992.223224.

[110] Liang Mei, Jingen Liu, Alfred O. Hero III, and Silvio Savarese. Robust object pose estimation via statistical manifold modeling. In *ICCV*, pages 967–974. IEEE, 2011.

[111] Nadia Payet and Sinisa Todorovic. From contours to 3d object detection and pose estimation. In *ICCV*, pages 983–990. IEEE, 2011.

[112] Hongsheng Li, Tian Shen, and Xiaolei Huang. Global optimization for alignment of generalized shapes. In *CVPR*, 2009.

[113] L-Z Liao L-W Zhao, S-W Luo. 3d object recognition and pose estimation using kernel pca. In *Int. Conf. on Machine learning & Cybernetics*. IEEE, 2004.

[114] Yacov Hel-Or and Michael Werman. Model based pose estimation of articulated and constrained objects. In *ECCV*, pages 262–273, 1994.

[115] I. Jane Mulligan, Alan K. Mackworth, and Peter D. Lawrence. A model-based vision system for manipulator position sensing. Technical report, Vancouver, BC, Canada, Canada, 1989.

[116] Ali Shahrokni, Tom Drummond, Francois Fleuret, and Pascal Fua. Classification-based probabilistic modeling of texture transition for fast line search tracking and delineation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 (3):570–576, 2009. ISSN 0162-8828. doi: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2008.236.

[117] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Real-time detection and tracking for augmented reality on mobile phones. *IEEE Trans. Vis. Comput. Graph.*, 16(3):355–368, 2010.

[118] Ahmet Denker and Tuğrul Adigüzel. Vision based robotic interception in industrial manipulation tasks. 1(12):388 – 395, 2007.

[119] J. K. Aggarwal and Q. Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73:428–440, 1999.

[120] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Commun. ACM*, 56(1):116–124, January 2013. ISSN 0001-0782. doi: 10.1145/2398356.2398381.

[121] Odest Chadwicke Jenkins, German González Serrano, and Matthew Maverick Loper. Tracking human motion and actions for interactive robots. In Cynthia Breazeal, Alan C. Schultz, Terry Fong, and Sara B. Kiesler, editors, *HRI*, pages 365–372. ACM, 2007. ISBN 978-1-59593-617-2.

[122] Michael A. Goodrich and Dan R. Olsen Jr. Seven principles of efficient human robot interaction. In *SMC*, pages 3942–3948. IEEE, 2003. ISBN 0-7803-7952-7.

[123] D. Kulić and E. A. Croft. Estimating intent for human-robot interaction. In *in IEEE Int. Conference on Advanced Robotics*, page 810815, 2003.

[124] Dana Kulić and Elizabeth A. Croft. Safe planning for human-robot interaction. *J. Robotic Syst.*, 22(7):383–396, July 2005.

[125] Dana Kulić and Elizabeth A. Croft. Affective state estimation for human-robot interaction. *IEEE Transactions on Robotics*, 23(5):991–1000, 2007.

[126] Takayuki Kanda, Hiroshi Ishiguro, Michita Imai, and Tetsuo Ono. Body movement analysis of human-robot interaction. In Georg Gottlob and Toby Walsh, editors, *IJCAI*, pages 177–182. Morgan Kaufmann, 2003.

[127] Yasser F. O. Mohammad, Toyoaki Nishida, and Shogo Okada. Unsupervised simultaneous learning of gestures, actions and their associations for human-robot interaction. In *IROS*, pages 2537–2544. IEEE, 2009.

[128] Tom Drummond and Roberto Cipolla. Real-time visual tracking of complex structures. 24(7):932–946, 2002.

[129] Andrew I Comport, Éric Marchand, and François Chaumette. Complex articulated object tracking. In *Articulated Motion and Deformable Objects*, pages 189–201. Springer, 2004.

[130] Andrew I Comport, Éric Marchand, and François Chaumette. Kinematic sets for real-time robust articulated object tracking. *Image Vision Comput.*, 25(3): 374–391, 2007.

[131] Stefano Pellegrini, Konrad Schindler, and Daniele Nardi. A generalisation of the icp algorithm for articulated bodies. In *Proc. British machine vision conference*, September 2008.

[132] Karl Pauwels, Vladimir Ivan, Eduardo Ros, and Sethu Vijayakumar. Real-time object pose recognition and tracking with an imprecisely calibrated moving rgb-d camera. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 2733–2740, 2014.

[133] Matthew Klingensmith, Thomas Galluzzo, Christopher Dellin, Moslem Kazemi, J Andrew Bagnell, and Nancy Pollard. Closed-loop servoing using real-time markerless arm tracking. In *IEEE Int. Conf. Robot. Autom. (Humanoids workshop)*, 2013.

[134] Karl Pauwels and Danica Kragic. Simtrack: A simulation-based framework for scalable real-time object pose detection and tracking. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[135] Nadia B Figueroa, Florian Schmidt, Haider Ali, and Nikolaos Mavridis. Joint origin identification of articulated robots with marker-based multi-camera optical tracking systems. *Robot. Auton. syst.*, 61(6):580–592, 2013.

[136] Tanner Schmidt, Richard Newcombe, and Dieter Fox. DART: Dense Articulated Real-Time Tracking. In *proc. Robotics: Science and Systems*, Berkeley, USA, 2014.

[137] J. Bohg, J. Romero, A. Herzog, and S. Schaal. Robot arm pose estimation through pixel-wise part classification. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 3143–3150, 2014.

[138] Felix Widmaier, Daniel Kappler, Stefan Schaal, and Jeannette Bohg. Robot arm pose estimation by pixel-wise regression of joint angles. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2016*. IEEE, May 2016.

[139] Dennis Park and Deva Ramanan. Articulated pose estimation with tiny synthetic videos. In *Proc. IEEE Int. Conf. Comput. Vision. Pattern Recog. Workshops*, pages 58–66, 2015.

[140] Stephan Gammeter, Andreas Ess, Tobias Jäggli, Konrad Schindler, Bastian Leibe, and Luc Van Gool. Articulated multi-body tracking under egomotion. In *Proc. European Conf. Comp. Vision*, pages 816–830. Springer, 2008.

[141] B. Houska, H.J. Ferreau, and M. Diehl. ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

[142] R. Calandra, S. Ivaldi, M. Deisenroth, and J. Peters. Learning torque control in presence of contacts using tactile sensing from robot skin. In *15th IEEE-RAS International Conference on Humanoid Robots*, pages 690–695, November 2015.

[143] Andrew I Comport, Eric Marchand, Muriel Pressigout, and Francois Chaumette. Real-time markerless tracking for augmented reality: the virtual visual servoing framework. 12(4):615–628, 2006.

[144] Sounkalo Dembélé, Nadine Le Fort-Piat, Naresh Marturi, and Brahim Tamadazte. Gluing free assembly of an advanced 3d structure using visual servoing. In *Micromechanics and Microsystems Europe Workshop*, pages 6–12, 2012.

[145] E. Marchand, F. Spindler, and F. Chaumette. Visp for visual servoing: a generic software platform with a wide class of robot control skills. 12(4):40–52, 2005.

[146] Steven G. Johnson. The nlopt nonlinear-optimization package. http://ab-initio.mit.edu/nlopt.

[147] Damien Jade Duff, Thomas Mörwald, Rustam Stolkin, and Jeremy Wyatt. Physical simulation for monocular 3d model based tracking. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5218–5225. IEEE, 2011.

[148] Jean-Philippe Saut, Serena Ivaldi, Anis Sahbani, and Philippe Bidaud. Grasping objects localized from uncertain point cloud data. *Robotics and Autonomous Systems*, 62(12):1742 – 1754, December 2014.

[149] Alessandro De Luca and Raffaella Mattone. Sensorless robot collision detection and hybrid force/motion control. In *ICRA*, pages 999–1004. IEEE, 2005.

[150] Lucas Manuelli and Russ Tedrake. Localizing external contact using proprioceptive sensors: The contact particle filter. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[151] Éric Marchand, Fabien Spindler, and François Chaumette. Visp for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics & Automation Magazine*, 12(4):40–52, 2005.