Université catholique de Louvain
Faculté des Sciences Appliquées

# Modeling, Simulation and Control of Redundantly Actuated Parallel Manipulators

**Latchezar GANOVSKI**
Mechanical Engineer

Ph.D. Thesis

Members of the Jury:

| | |
|---|---|
| Prof. Paul Fisette (promotor) | Université catholique de Louvain |
| Prof. Jean-Claude Samin (promotor) | Université catholique de Louvain |
| Prof. Joris De Schutter | Katholieke Universiteit Leuven |
| Prof. Pierre Duysinx | Université de Liège |
| Prof. Pierre-Yves Willems | Université catholique de Louvain |
| Prof. Grégoire Winckelmans | Université catholique de Louvain |

Louvain-la-Neuve, Belgium
December 2007

I would like to express my deepest gratitude to:

Paul Fisette, for his friendship, for being of constant help and support to me, for guiding perfectly my work on this thesis from the very beginning – with his precious ideas, advice and remarks, his words of encouragement and motivation, when I needed them the most,

Jean-Claude Samin, for so many fruitful discussions and enlightening remarks, as well as for his valuable supervision of this work,

Pierre-Yves Willems, for being such a magnificent mentor to me, in the scientific and every other sense of this term,

Paul Thibaut, another precious friend and colleague, with whom we spent many hours together, tackling mechatronic design problems,

Thierry Daras, for his kind assistance with respect to some hardware implementation aspects of this thesis,

Guy Campion, Joris De Schutter and Bram Demeulenaere, for their helpful advice on some robot control issues,

All my past and present colleagues at the PRM Division of the UCL, where I have been working for almost nine years, in a wonderful atmosphere of team spirit, union and friendship,

Maxime Raison, Vincent Spinewine and François Nicolas, for being such good office mates,

Paul Fisette and Jean-Claude Samin, once more, as well as Benoît Raucent – for landing me a helping hand in the edition of the text,

All my friends in Belgium and Bulgaria,

All of you, that I have probably forgot to thank here,

And, finally, my wife Evgenia and my stepdaughter Lora, for their love and support, for being my courage and my guiding light.

Thank you all from the bottom of my heart!

Latchezar, December 2007

*Felix qui potuit rerum cognoscere causas.*

# List of symbols

**Subscripts**

| | | |
|---|---|---|
| $n$ | : | corresponds to the number of degrees of freedom of the multibody system; |
| $k$ | : | corresponds to the number of joints of the multibody system; |
| $u$ | : | stands for the independent coordinates of the multibody system; |
| $v$ | : | stands for the dependent coordinates of the multibody system; |
| $a$ | : | stands for the actuated (active) coordinates of the multibody system; |
| $p$ | : | stands for the non-actuated (passive) coordinates of the multibody system; |
| $a_d$ | : | stands for the actuated (active) coordinates, corresponding in number to the degrees of freedom of the multibody system; |
| $a_r$ | : | stands for the actuated (active) coordinates, corresponding in number to the redundant actuators of the multibody system; |
| $c$ | : | stands for the kinematic constraint equations; |
| $m$ | : | stands for the parallel manipulator Jacobian; |

**Superscripts**

| | | |
|---|---|---|
| $n$ | : | stands for "power $n$"; |
| $d$ | : | stands for desired (reference) values of the controlled variables; |

**Operators**

| | | |
|---|---|---|
| $\dot{a}$ | : | first time derivative of $a$; |
| $\ddot{a}$ | : | second time derivative of $a$; |
| $\tilde{a}$ | : | tilde matrix associated with a $(3 \times 1)$ array $a$; |
| $\tilde{\mathbf{a}}$ | : | tilde tensor associated with a vector $\mathbf{a}$; |

**Main symbols**

| | | |
|---|---|---|
| $X$ | : | set of end-effector (absolute) generalized coordinates of the manipulator; |
| $q$ | : | set of joint (relative) generalized coordinates of the manipulator; |
| $J$ | : | Jacobian matrix; |
| $u$ | : | sub-set of independent generalized coordinates of the parallel manipulator; |

| | | |
|---|---|---|
| $v$ | : | sub-set of dependent generalized coordinates of the parallel manipulator; |
| $a$ | : | sub-set of actuated (active) generalized coordinates of the multibody system; |
| $p$ | : | sub-set of non-actuated (passive) generalized coordinates of the multibody system; |
| $W$ | : | end-effector "wrench" - resultant vector of the external torques and forces, applied on the end-effector; |
| $\tau$ | : | vector of torques/forces applied to the manipulator joints; |
| $\Re^n$ | : | Euclidean space of a dimension $n$; |
| $\epsilon^n$ | : | ellipsoid, defined in the n-dimensional Euclidean space $\Re^n$; |
| $\varsigma_u$ | : | sphere of radius equal to 1 (unit sphere); |
| $\Psi$ | : | surface (set of points), defined in the n-dimensional Euclidean space $\Re^n$; |
| $U$ | : | left unitary matrix of the singular value decomposition $J = U\ \Sigma\ V^T$; |
| $\Sigma$ | : | diagonal matrix (with nonnegative elements in decreasing order) of the singular value decomposition $J = U\ \Sigma\ V^T$; |
| $V$ | : | right unitary matrix of the singular value decomposition $J = U\ \Sigma\ V^T$; |
| $\lambda$ | : | vector of Lagrange multipliers; |
| $N_{act}$ | : | number of independent actuators (actuated joints); |
| $d_{AR}$ ($d_{FR}$) | : | number of degrees of actuation (force) redundancy; |
| $d_{eff}$ | : | number of end-effector degrees of freedom that the parallel manipulator; |
| $d_{mbs}$ | : | number of degrees of freedom of the multibody system; |
| $N^{body}$ | : | number of bodies of the multibody system; |
| $N^{joint}$ | : | number of joints of the multibody system; |
| $\{O, \{\hat{\mathbf{I}}\}\}$ | : | orthonormal inertial frame of origin $O$ in the three-dimensional Euclidian space, rigidly fixed to the base (body 0) of the multibody system; |
| $\{\hat{\mathbf{X}}^i\}$ | : | orthonormal reference frame in the three-dimensional Euclidian space, attached to body $i$; |
| $R^i$ | : | $(3 \times 3)$ absolute rotation matrix of $\{\hat{\mathbf{X}}^i\}$ with respect to $\{\hat{\mathbf{I}}\}$; |
| $R^{i,h}$ | : | $(3 \times 3)$ relative rotation matrix between frames $\{\hat{\mathbf{X}}^i\}$ and $\{\hat{\mathbf{X}}^h\}$; |
| $\mathbf{\Omega}^i$ | : | relative angular velocity vector of body $i$ with respect to its parent $h$; |
| $\boldsymbol{\omega}^i$ | : | absolute angular velocity vector of body $i$; |

| | | |
|---|---|---|
| $O^i$, $O'^i$ | : | reference attachment points of joint $i$ on the parent body $h$ and its child $i$, respectively; |
| $\mathbf{z}^i$ | : | relative position vector $\overrightarrow{O^i O'^i}$; |
| $\mathbf{d}^{ik}$ | : | position vector of the reference attachment point $O^k$ with respect to $O'^i$; |
| $\mathbf{d}_z^{ik}$ | : | extended position vector of $O^k$ with respect to $O'^i$; |
| $G^i$ | : | center of mass of body $i$; |
| $\mathbf{d}^{ii}$ | : | position vector of the center of mass $G^i$ with respect to $O'^i$; |
| $\mathbf{d}_z^{ii}$ | : | augmented position vector of $G^i$ with respect to $O^i$; |
| $\mathbf{x}^i$ | : | absolute position vector of the center of mass $G^i$; |
| | | |
| $m^i$ | : | mass of body $i$; |
| $I^i$ | : | inertia matrix of body $i$ with respect to its center of mass $G^i$; |
| $\mathbf{I}^i$ | : | inertia tensor of body $i$ with respect to its center of mass $G^i$; |
| $\mathbf{g}$ | : | gravity vector; |
| $\mathbf{F}^i$ | : | vector of the forces applied to body $i$ by its parent $h$ through joint $i$; |
| $\mathbf{F}^i_{ext}$ | : | resultant vector of the external forces (excluding the gravity), applied to body $i$; |
| $\mathbf{F}^i_{tot}$ | : | resultant vector of all the forces applied to body $i$; |
| $\mathbf{L}^i$ | : | vector of the torques applied to body $i$ by its parent $h$ through joint $i$; |
| $\mathbf{L}^i_{ext}$ | : | resultant vector of the external torques, applied to body $i$; |
| $\mathbf{L}^i_{tot}$ | : | resultant vector of all the torques applied to body $i$; |
| $Q^i$ | : | generalized force associated with the generalized joint coordinate $q^i$; |
| $S_d$ | : | diagonal selection matrix of a hybrid controller; |
| $S_r$ | : | diagonal selection matrix of a redundant hybrid controller; |

**Abbreviations**

| | | |
|---|---|---|
| MBS | : | multibody system; |
| d.o.f. | : | degrees of freedom; |
| RPPM | : | reconfigurable planar parallel manipulator; |
| RAPM | : | redundantly actuated parallel manipulator; |
| PD | : | proportional-derivative; |
| PDFF | : | proportional-derivative plus feed forward; |
| FCTT | : | feed-forward computed-torque. |

# Contents

# Introduction

Only recently, the redundantly actuated manipulators have really interested the scientists of the domain of parallel robotics. The significant advantages in terms of enlarged effective workspace, higher payload ratio and better manipulability of this type of parallel structures with respect to non-redundantly actuated systems will undoubtedly engender more and more applications in different directions: high-precision machining, fault-tolerant manipulators, transport and outer-space applications, surgical operations, etc.

However, to the present day, the number of real applications of redundantly actuated parallel manipulators is still quite limited. This is partially due to the fact that to develop effective strategies for actuation of such type of closed-loop structures is not so obvious. The solutions for the redundant actuator efforts are usually sought via optimization techniques targeting their minimization with respect to energy considerations, actuator performance limits, etc. As another application of the redundant actuation principles, in some recent researches the force redundancy is used to cancel the backlash, existing in the manipulator structure (joints), by bringing the latter into internally loaded state throughout the motion it has to accomplish.

Another major difficulty when dealing with redundant actuation of parallel manipulator arises from their control. The majority of controllers that can be found in most robotic applications, are developed to act over non-redundantly actuated systems, in which the number of actuators corresponds to the number of system degrees of freedom. Therefore, finding control algorithms that are appropriate to parallel structures (closed-loop topologies) subjected to redundant actuation remains a challenging task.

## Framework and main achievements of the present research

The present research aims first of all at modeling the dynamics of redundantly actuated parallel manipulators, using multibody formalisms [1, 2, 3]. The modeling approach is based on relative coordinates, recursive Newton/Euler formulation with Lagrange multipliers, and the coordinate partitioning technique [4], allowing for the construction of compact, computation efficient and numerically stable direct and inverse dynamic models. The developed inverse models allow for finding solutions to the non-redundant or redundant manipulator actuation tasks. These models are robust with respect to singularity problems and lead to torque optimization in cases of actuator redundancy. The direct dynamic models are used either for validation of the computed actuator torque solutions, or for real-time control simulations, used to tune or test different algorithms for non-redundant/redundant actuation control.

Along with the development of specific models, cited above, as another goal (and achievement) of this research work can be considered the original approach for actuation of parallel manipulators that successfully eliminates the effects of the so-called *parallel* (force) singular configurations. The force singularities depend on the actuator configuration and are specific to parallel manipulators. They degrade the manipulator force/torque performances, locally causing uncontrollable motion of its end-effector and incapability to withstand external loads applied.

The proposed approach for modeling and actuation provides a global solution for taking advantage of the actuator redundancy, when parallel manipulators have to accomplish trajectories that contain force singularities. The complete approach algorithm and its corresponding computer implementation in MATLAB/SIMULINK consist in several, consecutive stages:

- Description of the considered manipulator topology, type and performances of the actuator to be used, the joints that can be actuated, the trajectories to be followed in the end-effector space, etc.

- Symbolic generation of compact and computationally efficient kinematic and dynamic models, on the basis of the Newton/Euler recursive algorithm and using a dedicated symbolic generation software [5, 3, 6];

- Solution of the parallel manipulator inverse kinematics, numerically stable with respect to kinematic loop closure problems. Transformation in joint coordinates of the trajectories to be followed;

- Application of a *piecewise actuation* strategy generating *active/passive* coordinate partition sequences to eliminate local force singularity problems. Depending on the trajectory, the manipulator topology, the presence of internal friction forces/torques, the joints available for actuation and the actuator performance specifications, the strategy produces non-redundant or redundant actuation configurations;

- Inverse dynamics solution: pseudoinverse or infinite-norm torque minimization in cases of redundant actuation;

- Direct dynamics real-time integration, validating the obtained actuation solutions. As a second result of the integration process, *independent/dependent* coordinate partition sequences are generated to provide for control simulations that are numerically stable with respect to kinematic loop closure problems;

- Real-time control simulations, applying the obtained solutions for the actuator torques and the piecewise coordinate partitioning sequences. Controller tuning by means of exhaustive control simulations, testing of different control laws in cases of redundantly actuated parallel closed-loop multibody systems.

Finally, the design of a redundantly actuated prototype and the real-time control experiments carried out on it can be cited as another achievements of this work. The experimental results permitted to practically verify the strategies for effective actuation, proposed by the author, and their application to the control of over-actuated parallel manipulators.

As we shall see, the results from simulations and experiments show the advantages of using redundant actuation on parallel manipulators in terms of force singularity elimination. The over-actuation allows for a better control over smooth, continuous manipulator motion in a singularity-free workspace.

# Chapter overview

This research work is organized as follows.

In Chapter 1, a general introduction to the domain of parallel robotics is given with some historical facts and going through some basic robotic terminology, manipulator architecture elements, characteristics, and definitions. Then, in Chapter 2, the principal types of parallel manipulator configurations: assembly modes and poses, are defined, their description as solutions to the direct and inverse kinematic problem is presented, and some common difficulties when finding these solution are commented.

Chapter 3 is dedicated to the robot singular configurations of different kind, degrading the manipulator performances in terms of end-effector velocities and forces/torques. Singularity classifications are given in the chapter, which concern parallel robots and are made using the manipulator kinematic equations or velocity and force ellipsoid definitions. The singularity problematics is followed by an extensive discussion on the redundant actuation of parallel manipulators in Chapter 4. In the latter, a comparison of the force redundancy with the kinematic redundancy of serial robots is given first, then a classification and state of the art outlining the fields of application of the redundant actuation are presented, as well as methods used by the researchers to solve for it.

Chapter 5 describes the principal concepts of the multibody formalism [1, 2, 3], its terminology and the basic notations it employs so as to define tree-like and closed-loop multibody systems. Then, in Chapter 6, the generation of kinematic models of parallel manipulators is discussed, detailing the kinematic loop cutting procedures used, and the difference between the two coordinate partitioning schemes, considered both for the kinematic and the dynamic modeling stage. The closed-loop multibody system dynamics is dealt with in Chapter 7. The dynamic model reduced forms, obtained by using the coordinate partitioning technique [4] and Lagrange multiplier elimination, and the model computation, based on the Newton-Euler recursive formalism, are presented. The modeling is then extended to cases, in which internal joint friction forces/torques are present in the system. A short review of the methods for direct dynamics time integration is then given. At the end of the chapter, one of the contributions of the present research is described, namely, the development of a piecewise time integration procedure that is numerically robust with respect to loop closure problems and allows for stable real-time control simulations over any trajectory in the manipulator workspace.

In Chapter 8, a special *piecewise actuation* approach, proposed in this work, is commented in detail. This approach targets the elimination of force singularities that exist in the manipulator workspace, and produces non-redundant or redundant actuator solutions for the treated parallel manipulators. In cases of actuator redundancy, two practically oriented torque optimization solutions are presented – a *pseudoinverse-equivalent* solution, and an *infinity-norm torque minimization* solution respecting the actuator torque performance limits. At the end of the chapter, a computer implementation of the piecewise actuation algorithm that makes use of the efficiently generated symbolic dynamic models, is proposed.

Chapter 9 is devoted to the control of redundantly actuated parallel manipulators, demonstrating the advantage of applying the force redundancy by widely used control schemes. A short review of the latter and some basic methods for controller tuning are followed by a description of the control applications of the present work and their corresponding computer implementations. Control simulation results for two studied closed-loop multibody systems: a four-bar mechanism and a planar parallel manipulator, are systematized and commented at the end of this chapter.

Finally, a prototype, created at the Center for Research in Mechatronics of the UCL, and used for experimental validation of the obtained theoretical results, is described in Chapter 10 in terms of mechanical, hardware and software design solutions adopted. The chapter presents as well the experimental results, obtained from the real-time prototype control.

In the very end of this work, conclusions are drawn on the applicability of the redundant actuation to the control of closed-loop multibody systems, before summarizing the contributions of this work and giving some perspective directions.

# Part I

# Parallel Robots

# Chapter 1

# Definitions and Classification

## 1.1 Some History About Robots

One of the first documented evidences of a sophisticated mechanism is that of the *Clepsydra* (or water clock), which was made in 250 B.C. It was created by Ctesibius of Alexandria – a Greek physicist and inventor, and can be considered as one of the first primitive robots ever made in the human history.

The swiss inventors Pierre and Henri-Louis Jacquet-Droz created some of the most complicated automatons of this period. In 1774 their *Automatic Scribe* was announced to the society. The mechanized boy-like automaton could draw and write any message up to 40 characters long. A robot-woman playing a piano was another one of their great inventions. A century later – in 1892, Seward Babbitt created a motorized crane with gripper to remove ingots from a furnace.

The word "robot" was introduced for the first time in 1921 by the czhech writer Karel Capek in his play R.U.R. – *Rossum's Universal Robots*. The word comes from the Czech "robota", which means tedious labor. The plot was simple: robotic workers – "mechanical men" – rebel against their masters and assume control of the world after slaughtering them, i.e. man makes robot then robot kills man!

Seventeen years later, a parallel mechanism for automated spray painting was designed by Willard L. Pollard. This extraordinary invention (Figure 1.1) represented in fact a real three-branched parallel robot intended for spray painting, but unfortunately was never built. It was the son Willard Pollard Jr.. in fact, who succeeded in getting a patent[1] in 1942 for this ingenious mechanism, that he has been a co-designer of.

In 1941, the science fiction writer Isaac Asimov used for the first time the word "robotics" to describe the technology of robots, and predicted the rise of a powerful robot industry. A year later Asimov wrote "Runaround", a story about robots, which contained the "Three Laws of Robotics". This story was later included in Asimov's famous book "I, Robot" [7].

In 1946 George Devol patented a general purpose playback device for controlling ma-

[1] US Patent No. 2,286,571

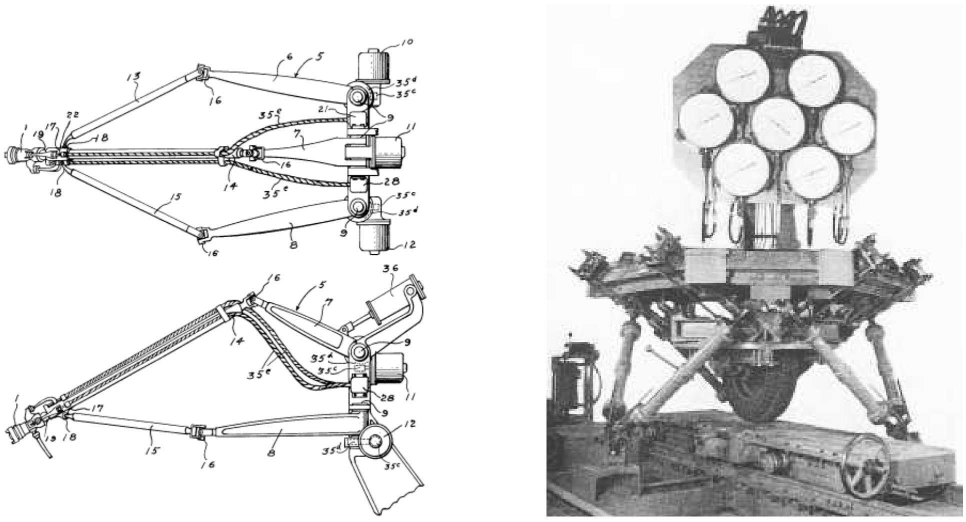


Figure 1.1: Left: The first industrial parallel robot, patented in 1942. Right: The first octahedral hexapod or the original Gough platform in 1954 (Proc. IMechE, 1965-66).

chines. The device used a magnetic process recorder. In the same year the computer emerged for the first time. The american scientists J. Presper Eckert and John Mauchly built the first large electronic computer, called the *Eniac*, at the University of Pennsylvania.

In 1947 another parallel mechanism was invented - the infamous variable-length-strut octahedral hexapod (Figure 1.1). It was designed by Dr. Eric Gough – a distinguished automotive engineer at Dunlop Rubber Co., Birmingham, England, in response to the needs of a machine that would permit to determine the properties of tires under combined loads.

Among the early robots (1940's - 50's) were Grey Walter's *Elsie the tortoise*, also called "Machina speculatrix", and *Shakey* – a small, unstable box on wheels, that used memory and logical reasoning to solve problems and navigate in its environment. It was developed at the Stanford Research Institute (SRI) in Palo Alto, California in the 1960s (Figure 1.2). These robots were probably the first representatives of a future robotics sub-domain called *mobile robotics.*

The first industrial robot patent was granted to George Devol Jr, in 1954. Devol named the control system of his robots "unimation". Then he sold his idea to Joseph Engelberger, who transformed his machines into what he called Unimates. Engelberger started the first robotics company – "Unimation", and has been often called the "father of robotics". Yet, during the sixties the robotics was not seen by the industrial companies in the USA and Europe as capable of bringing significant benefits. The idea of large introduction of robots in the industry was firstly embraced by the japans, which quickly created a huge market segment in Japan.

During the seventies thousands of Unimates and Pumas (another well known Unimation industrial robot family) entered the human workplaces in the plants and factories. They influenced different industries, especially the automotive and the aircraft industry,

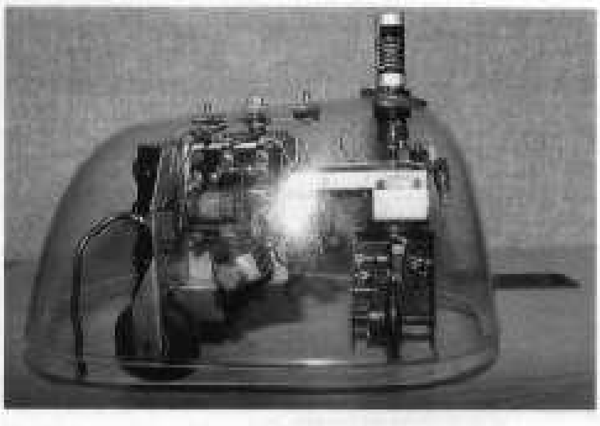
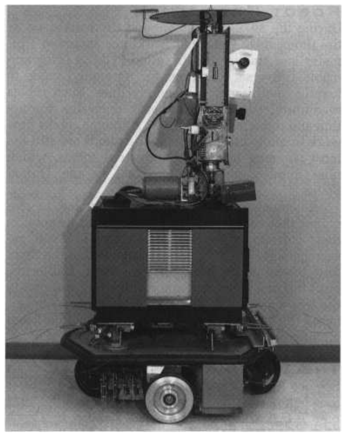



Figure 1.2: Left: Grey Walter's tortoise, restored recently by Owen Holland and fully operational (from Arkin, 1998). Right: Shakey the robot (from Wickelgren, 1996).

remarkably improving the effectiveness in terms of production rate and precision of different kinds of operations - riveting, arc-welding, spray painting, part and module assembly, etc. In the eighties the industrial robotics knew its highs and lows, and the excitement over the incoming "next industrial revolution", brought by the intentions of extensively using robots, progressively calmed down.

Nevertheless, during the last thirty years the robot manipulators found their places in warehouses, laboratories, hospitals, harmful environments, even outer space. Numerous applications appeared. The majority of the industrial manipulators were and presently still are of the so called *serial* morphology – a term that we will precise in Section 1.2.1. Many of the serial manipulators are of special class, called *anthropomorphic*. The term is derived from two greek words: $\alpha\nu\theta\rho\omega\pi o\varsigma$ (anthropos), meaning *human*, and $\mu o\rho\varphi\eta$ (morphe), meaning *shape* or *form*, i.e. the morphology of these manipulators is a human-like, usually resembling the human arm.

In the late eighties, demands started appearing for robots possessing lower inertia and high robustness, motion rapidity and precision, along with capability of manipulating bigger loads. This pushed further the research and development in terms of novel robot morphologies with improved functional characteristics. Bit by bit, the parallel robotics drew bigger interest, to become today a central robotic research domain with multiple application segments: from machining operations to surgery assistance and vehicle, aircraft and spacecraft simulators. Many scientists put the milestones to this direction, creating novel parallel robot types. Starting back from the first parallel manipulators, created during the period 1945-1970, we could cite here some famous ones, such as:

- The *Gough platform* - a parallel manipulator (Figure 1.1), created in 1947 by Eric Gough. The motion simulator of Klaus Cappel that he came up with in 1962 as a solution to the request of the Franklin Institute Research Laboratories in Philadelphia to improve an existing conventional vibration system (Figure 1.3) and the platform of D. Stewart [8] he proposed to use in a flight simulator (Figure 1.4) in 1965, can be mentioned here as well. These manipulators gave birth to the well-known class of parallel octahedral hexapods, called also *hexapod positioners* (Figure 1.4).

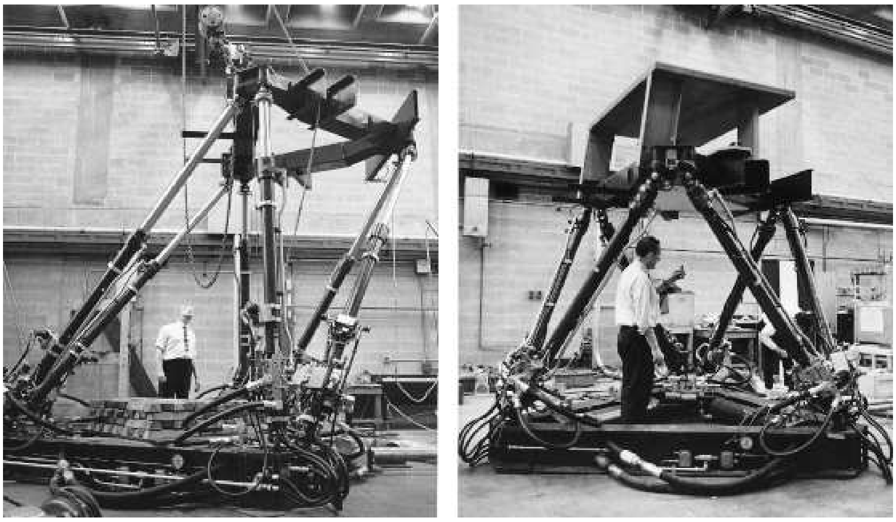


Figure 1.3: The flight simulator of Klaus Cappel, based on an octahedral hexapod (courtesy of Klaus Cappel)

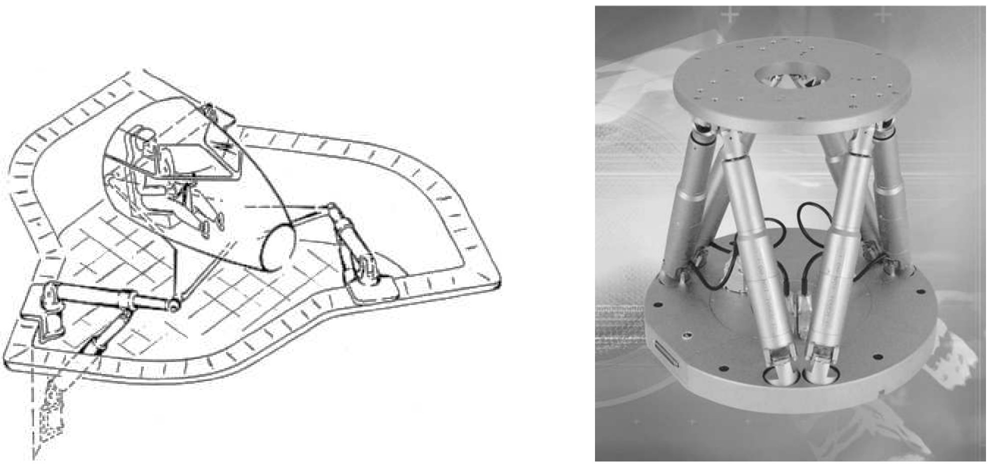


Figure 1.4: Left: Schematic drawing of the platform, proposed by D. Stewart [8]. Right: Contemporary hexapod manipulator (image courtesy of PI (Physik Instrumente) GmbH and Co. KG.).

- The *Delta robot* family. The ingenious idea in the early 80's of Raymond Clavel [9], professor at the Ecole Polytechnique Fédérale de Lausanne, of using light parallelograms as constitutive elements of the legs of a parallel robot (Figure 1.5), gave birth to the Deltas. The use of base-mounted actuators and low-mass elements allows the manipulator to achieve accelerations of up to 50 $g$ in experimental environments and 12 $g$ in industrial applications, which makes it convenient for pick-and-place operations of light objects (from 10 gr to 1 kg) at high speeds. Nowadays, the

Delta robots find multiple applications, including industrial machining operations like drilling, etc.

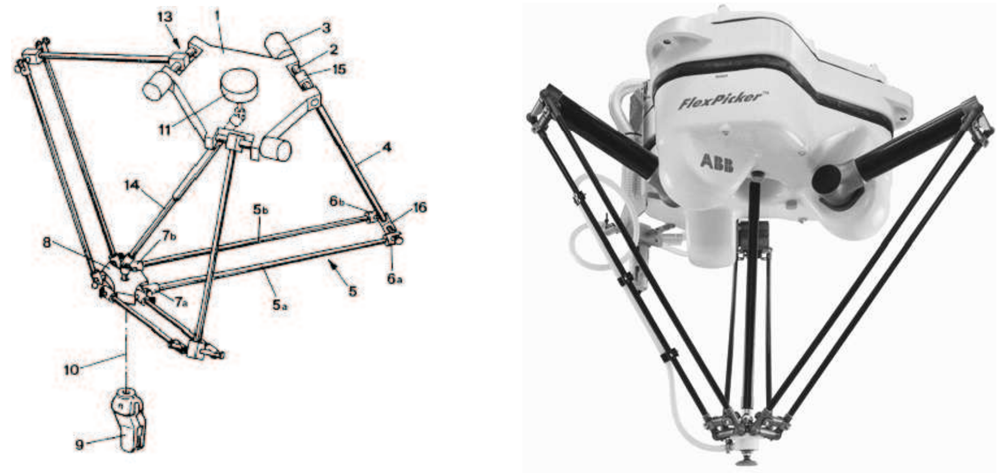



Figure 1.5: Left: Schematic drawing of the Delta robot (from R. Clavel US patent [10]). Right: ABB Flexible Automation's IRB 340 FlexPicker (courtesy of ABB Flexible Automation)

- The *left hand* parallel manipulator (Figure 1.6), developed in 1986 at the INRIA (Institut National de Recherche en Informatique et en Automatique), intended to serve as a dexterity enhancer for another manipulators, mostly of serial type.

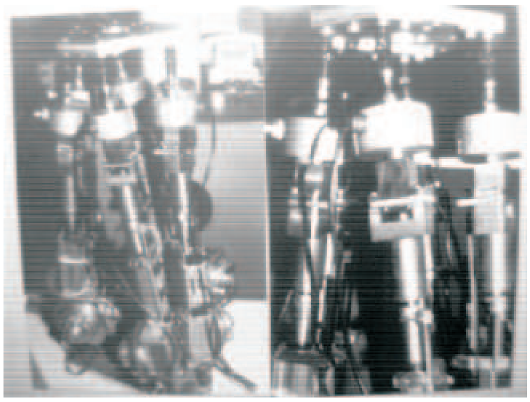


Figure 1.6: The Left Hand robot, developed in 1986 at the INRIA (image from [11])

- The *articulated truss* structures, e.g. the Logabex LX4 robot of the Logabex company (Figure 1.7) or the UTIAS Trussarm, designed at the University of Toronto. These robots consist of piled up identical parallel mechanisms and possess large workspace, and good ratio "load capacity/manipulator mass'. The Logabex robot, for example, is a series of left-hand parallel robots.

- The *Agile Eye* (a spherical parallel mechanism) developed by Gosselin and Hamel [12] in the Robotics Laboratory at Laval University, Canada (Figure 1.8) and prin-

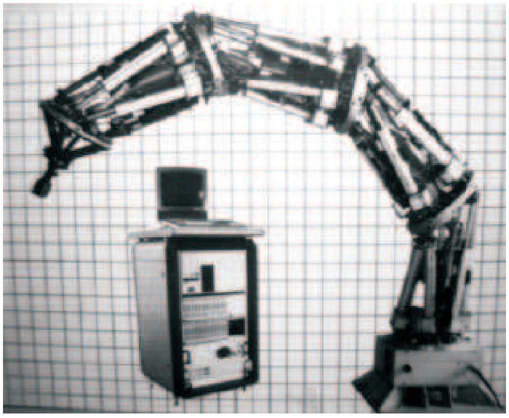


Figure 1.7: The Logabex robot LX4 (image from [11], taken by kind permission of the Logabex company)

cipally destined to fast video camera orientation tasks. Because of its low inertia and inherent stiffness, the mechanism can achieve angular velocities superior to 1000 deg/sec and angular accelerations greater than 20000 deg/sec$^2$, largely outperforming the human eye. Since patented in 1993, the Agile Eye gained popularity, giving birth to some simpler, yet very effective mechanisms.

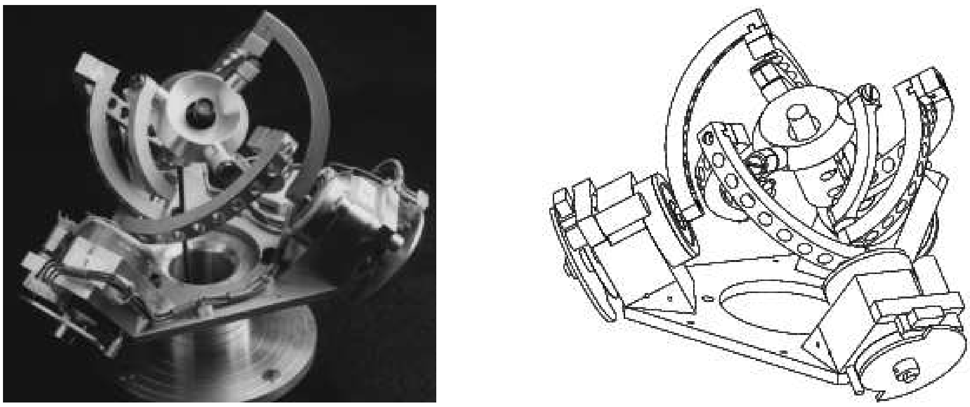


Figure 1.8: The Agile Eye parallel robot (images courtesy of Robotics Laboratory at University of Laval, Canada)

These are, of course, only some examples of widely known parallel manipulator inventions. There are many others, subject to mentioning, review or extensive analysis, that we shall not give here, as this would go beyond the scope of the present work. Let us though conclude this section with an indication of a very good and thorough parallel architecture history and classification that the reader could find in [11].

## 1.2 General terminology

Before we proceed with the description of some basic manipulator topologies (architectures), examples of which we shall use further in this work, let us recall a few fundamental definitions from the domain of mechanics and robotics. We address them here with the intention to recall, on the one hand, at least some of the scientists, whose contributions in this direction are significant, and to gradually build, on the other hand, a sufficient theoretical basis for the comprehension of the concepts of this work.

For any rigid body, which can move in the space, its motion can always be described as a combination of at most six basic motions - three independent translations and three independent rotations along/around three mutually orthogonal axes. These motions are also known as *degrees of freedom* (D.O.F.) of the rigid body and are represented for every time instant by a set of independent variables (coordinates), completely defining the body's position and orientation with respect to an inertial reference frame.

A convenient set of coordinates is usually chosen for a particular problem in mechanics. If several rigid bodies that form a mechanical system (e.g. a robot) are treated, then scalar variables are needed to define the configuration of each one of them. These variables are known as *generalized coordinates*, as they can be of different kind and serve to obtain a complete description of the system configuration. *Cartesian* coordinates and *spherical* (polar) coordinates are examples of generalized coordinates.

In the universal language of mechanics, the basic constitutive element of every chain (mechanism, manipulator, etc.) is a ***body***, often considered rigid. The bodies are interconnected by ***joints*** – simple mechanical devices, allowing relative motions between the connected elements.

According to the terminology of the *International Federation for the Promotion of Mechanism and Machine Science* (IFToMM) and the document ISO 8373:1994 of the *International Standart Organization*, the following definitions of principal interest to our work can be given:

- A ***kinematic chain*** is an assemblage of links and joints (IFToMM).
- A ***mechanism*** is a system of bodies designed to convert motions of, and forces on, one or several bodies into constrained motions of, and forces on, other bodies (IFToMM).
- A ***machine*** is a mechanical system that performs a specific task, such as forming of material, and the transference and transformation of motion and force (IFToMM).
- A ***manipulator*** is a machine, the mechanism of which usually consists of a series of segments, jointed or sliding relative to one another, for the purpose of grasping and/or moving objects (pieces or tools) usually in several degrees of freedom. It may be controlled by an operator, a programmable electronic controller, or any logic system (ISO8373).

In order to avoid the confusion between the terms *link*, *segment* and *joint*, and be compliant with the terminology of the *multibody formalism* that will be used in this work, we shall rely on *bodies* and *joints.*

A definition that explicits the difference between *manipulator* and *robot*, is given by the French standard document Norme Française E 61-100:

- A ***robot*** (or *industrial robot*) is an *automated manipulator*, which features position control; it shall be multi-usage, flexible and re-programmable. The device shall be able to accomplish various programmable displacement paths which means to position and orient materials, parts, tools, instruments or specialized equipments.

In the pages to follow, *manipulator* will be used more often than *robot.* The only reason is that, even if the second term obviously comprises the first one, the use of *robot* in a scientific context is generally less common.

### 1.2.1 Manipulator architecture elements and characteristics

As we mentioned previously, while during the 70ies and the 80ies most of the developments and industrial applications concerned serial manipulators, in the last twenty years broader research has been carried out in the field of parallel robotics, because of numerous new applications that require higher payload ratio[2], together with high rigidity, rapidity and very high precision. Table 1.2.1 sums up the characteristics of serial and parallel manipulators.

| | Serial Manipulators | Parallel Manipulators |
|---|---|---|
| Workspace | Larger | Smaller |
| Compliance | Higher | Low |
| Payload ratio | Limited | High |
| Precision | Limited | Very high |
| Rapidity | Limited | Very High |
| Repeatability | Good | Very good |

Table 1.1: Comparison between serial and parallel manipulator characteristics

In every robotic arm or leg of a serial structure, the first moving body of the kinematic chain is connected to another, fixed body, called *base* or *fixed platform*, whereas the last one - to a terminal moving body, called *wrist*, *tool plate* or *flange*, usually carrying an application-specific tool - *end-effector.* The term *end-effector degrees of freedom* is very often used in robotics, meaning in fact the degrees of freedom at the tool-plate, because the tool itself is assumed a body, rigidly fixed to it.

[2]The *payload ratio* is defined as the mass of the manipulated load divided by the mass of the structure

Two classical examples of serial robots are the widely known and used SCARA (selective compliant articulated robot arm) robot, developed by Sankyo and IBM Corp. in 1975, and the PUMA560 manipulator of Unimation (Figure 1.9). Designed more that 25 years ago, today still they demonstrate precision and payload ratio that are quite acceptable for industrial serial manipulators. The SCARA robot is typically used for sorting and stocking, and can perform different machining operations like drilling, cutting and welding, but it's end-effector is of limited mobility (restrained orientation). The PUMA robot itself is of an anthropomorphic structure and much more flexible in terms of end-effector position/orientation combinations.

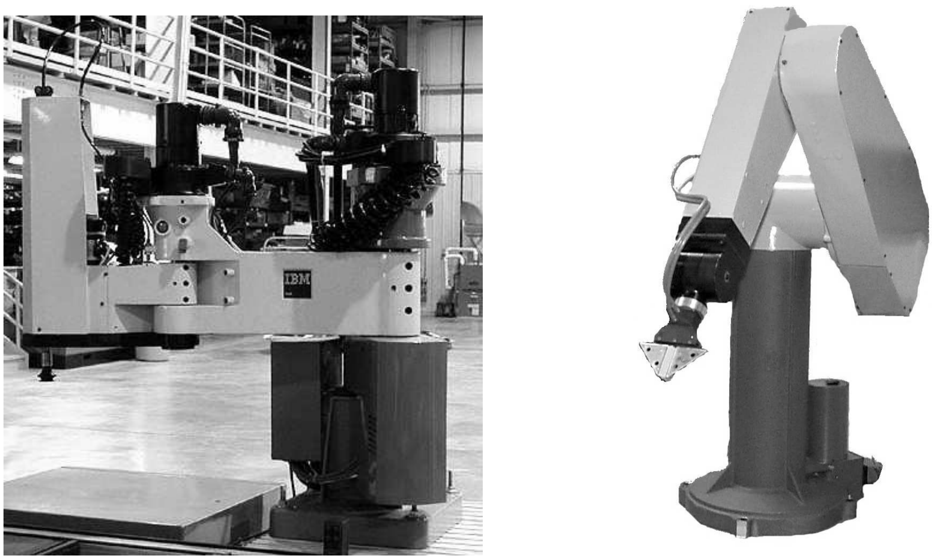



Figure 1.9: Typical industrial serial robots. Left: IBM SCARA robot. Right: Unimation PUMA 560 robot.

Different definitions of the term *serial kinematic chain* can be found in the literature, most of them being quite similar and describing it as a set of rigid bodies interconnected *in series* by joints. An example of such definition can be found in [11]: "A *serial manipulator* is defined as simple kinematic chain, for which all the connection degrees are equal to 2, except for the base and the end-effector". But this definition requires the definition of two other terms:

- *connection degree* of a manipulator link – the number of rigid bodies[3] attached to it by a joint,
- *simple kinematic chains* – kinematic chains, in which each member possesses a connection degree less or equal to 2.

As regards the different manipulator topologies and basic constitutive elements, in our work we shall rather use the definitions, based on the *multibody formalism* terminology

[3]Here again, we would like to note that "rigid body" should be understood under the term "link"!

[2], which, while being more general than those of the field of robotics, remain nevertheless straightforward and clear when dealing with the latter.

### 1.2.2 Parallel manipulator definitions

As the present work concerns parallel manipulators, in this section more attention is paid to definitions related to them.

Two examples of parallel topologies are given in Figure 1.10. When parallel manipulators are dealt with, the tool plate is very often called *moving platform*, usually of lesser dimensions than the fixed one.

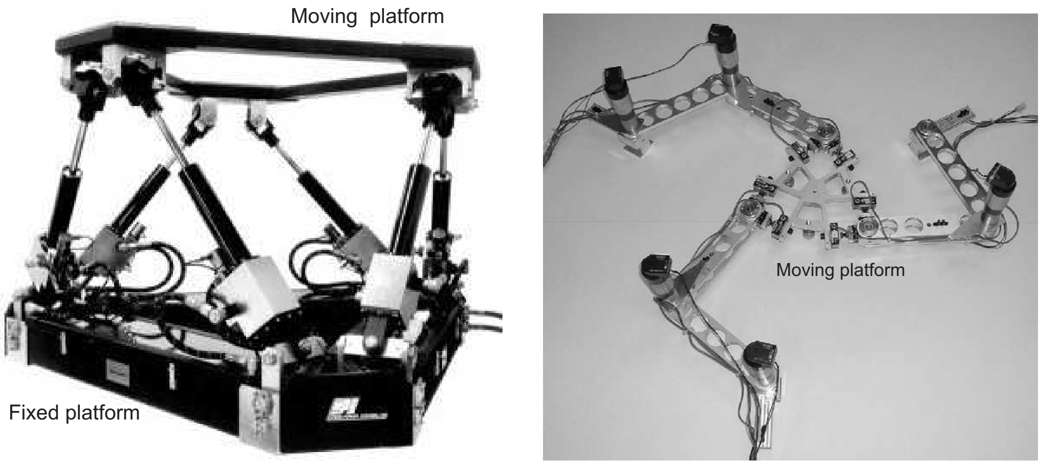



Figure 1.10: Two parallel robots. Left: 3-dimensional Motion base by Hydra-Power Systems, Inc. Right: Planar parallel robot prototype (image courtesy of F. Frimani and R. Podhorodeski of Robotics and Mechanisms Laboratory at the University of Victoria, Canada).

Merlet defined comprehensively in [11] the notions of *generalized parallel manipulator* and *parallel robot*:

*Definition 1:* A ***generalized parallel manipulator*** is a closed-loop mechanism whose moving platform is linked to the base by several independent chains.

*Definition 2:* A ***parallel robot*** is made up of an end-effector with n-degrees of freedom, and of fixed base, linked together by at least two independent kinematic chains.

The definitions of *fully parallel manipulator* and *light parallel manipulator* that can also be found in [11] are useful for the classification and analysis of parallel robots:

*Definition 3:* A ***fully parallel manipulator*** is a closed-loop mechanism, the number of chains of which is strictly equal to the moving platform number of degrees of freedom

(Figure 1.11, left and right).

<u>*Definition 4:*</u> A ***light parallel manipulator*** is a parallel manipulator, all of the actuators of which are fixed to its base (Figure 1.11, right).

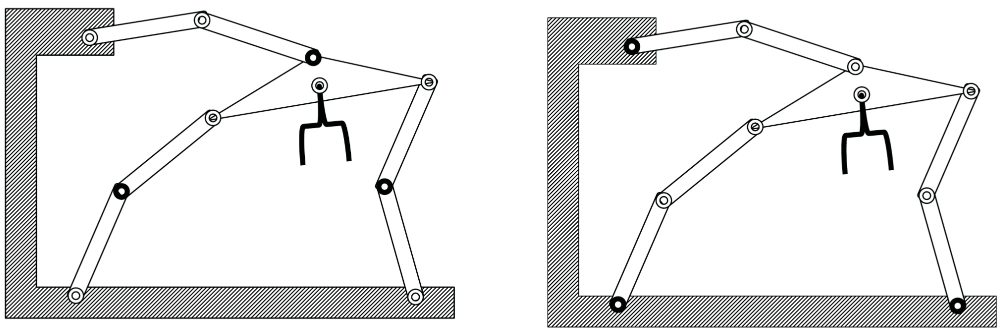


Figure 1.11: Example of planar manipulator schematics. Left: A planar fully parallel manipulator. Right: The same manipulator in a "light" version (the black joints show the actuator locations).

The light parallel manipulators show better kinematic and dynamic performances, due to lack of actuator mass and inertia influence on the moving parts of the mechanism. As we shall see later, the prototype discussed in this work is an example of a light parallel manipulator as well.

Some of the scientists have amused themselves in finding formulas that characterize from different points of view parallel manipulators, giving numerical values for their parallelism, mobility, etc. The formula of Gosselin, for example, that must be fulfilled for fully parallel manipulators, can be found in the literature [11]:

$$p(n-6) = -6 \tag{1.1}$$

Here $p$ is the number of chains of the parallel manipulator and $n$ is the number of rigid bodies in each chain. In order for this formula to be correct, for each chain the base must be counted as a first body and the moving platform – as a terminal body.

Quite often the different characteristic formulas are specific to particular sets of manipulator architectures (number of chains, type of joints, ...) and, what is more, differ in the manner of body counting (see the example above). Therefore, when it comes to defining manipulator topologies, number of degrees of freedom, etc. we shall hold only to the definitions and formulas, based on the *multibody formalism* we already mentioned. This formalism allows for a generalized, clear and compact description of any *multibody* system, i.e. any mechanical system, composed of *rigid bodies* connected by *joints*. We shall implement the multibody formalism in the modeling and analysis of kinematics and dynamics of parallel manipulators. In Part II, we shall discuss its conventions and hypotheses.

Concluding this chapter, we would finally note the existence of *hybrid manipulators* (see Figure 1.12 for example), the topology of which is a assembled sequence of a serial and a parallel structure. The latter is often called *active wrist* in this particular robot topology case.

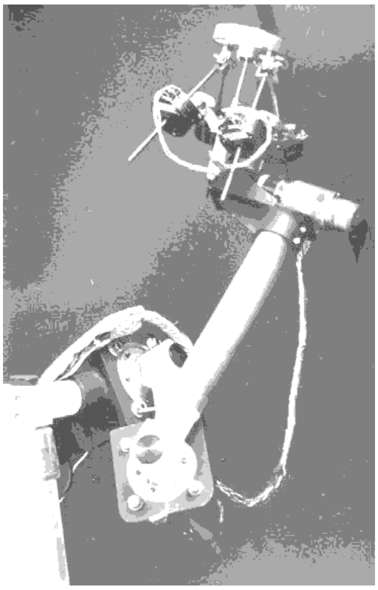


Figure 1.12: McGill-IRIS C3 hybrid manipulator

# Chapter 2

# Parallel Manipulator Configurations

In this chapter we shall briefly review the two main types of configurations for parallel manipulators: *assembly modes* and *poses*. In order to define them, a description of manipulator *direct* and *inverse kinematic problem* will be introduced.

The direct and inverse kinematic problems concern important aspects of the robotics: manipulator design, geometry and performance optimizations, etc. In the literature they can also be found as *forward* and *backward kinematics*. In this introductory chapter we shall limit their description to a manipulator position (configuration) level. More detailed analysis and modeling of the kinematics, extended to velocity and acceleration levels, will be presented in Chapter 6.

Let $X = [X_1 \ ... \ X_n]^T$ be the set of absolute generalized coordinates, describing the position and the orientation of the parallel manipulator moving platform, and $q = [q_1 \ ... \ q_k]^T$ – the set of joint generalized coordinates of the manipulator. Because of the existing closed-loop constraints (detailed later in Section 6.2) over the mechanism topology, the mapping between $X$ and $q$ is feasible through a sub-set $q_u = [q_{u_1} \ ... \ q_{u_n}]^T$, which contains the so-called *independent* joint generalized coordinates and corresponds in dimension to $X$. This sub-set is determined using the *coordinate partitioning* method [4], to which we shall refer systematically throughout this work, giving more detail on it in Section 3.2 and mostly in Section 6.2.2. The number $n$ represents the number of degrees of freedom of the moving platform (the end-effector).

The two sets of coordinates are mutually dependent, their dependency can be expressed mathematically by nonlinear equations of the form:

$$F(X, q_u) = 0 \tag{2.1}$$

This form is similar to the one, used for serial manipulators, in which all the joint coordinates can be considered independent ($q_u \equiv q$). It is used as a starting point of the manipulator kinematic modeling. As parallel robots are in the scope of the present work and the analysis of kinematics is limited to manipulator positions (configurations), we shall describe the two kinematic problems and define the two basic types of configuration for parallel manipulators.

## 2.1 Inverse kinematic problem

The inverse kinematic problem consists in finding the possible solutions for the set $q_u$, given the position and the orientation of the moving platform $X$. In the domain of parallel robotics these solutions are also known as ***poses*** of the parallel manipulator (Figure 2.2).

The inverse kinematic problem of parallel manipulators depends on the manipulator geometry and the type of joints. A unique or multiple poses could exist. For example, if a planar fully parallel manipulator with prismatic joints (representing translational actuators) (Figure 2.1) is considered, there is only one possible set of values for the coordinates $q_u$ (one manipulator *pose*) that gives the desired $X$, whereas for the manipulator of Figure 2.2 there exist four different sets of values and thus four different poses.

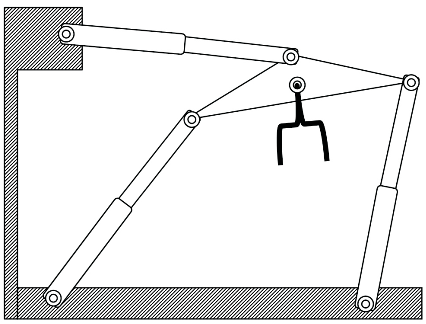


Figure 2.1: A planar fully parallel manipulator with translational actuators possesses a unique solution to its inverse kinematics

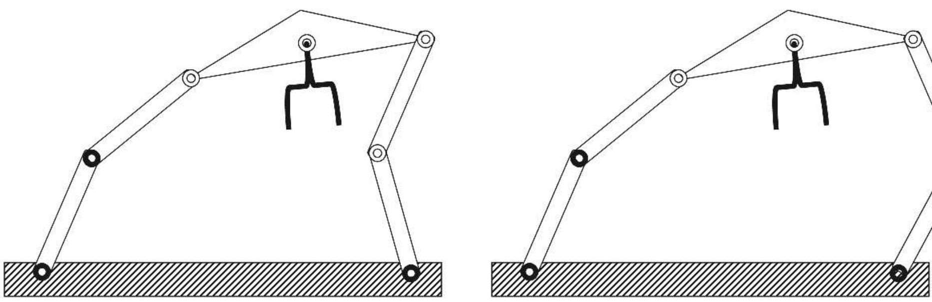


Figure 2.2: Different poses of a parallel manipulator: the black joints correspond to $q_u$.

An example of an analytical method for solving the inverse position kinematics of parallel manipulators, proposed by Merlet in [11], consists in expressing the vectors $\overrightarrow{A_i\,B_i}$, formed by the origins $A_i$ and the extremities $B_i$ of each manipulator leg $i$, as functions of the moving platform generalized coordinates. This approach eventually leads to an explicit form of the inverse kinematic problem:

$$q_u = g(X) \tag{2.2}$$

The solution could become difficult in case of spatial parallel manipulators with complex leg topology. But the legs usually are identical, which partially simplifies the inverse kinematic task.

## 2.2 Direct kinematic problem

The parallel manipulator direct kinematic problem consists in finding the possible solutions for the position and the orientation of the moving platform $X$, given a pose $q_u$. It can be found in the literature in the following explicit form:

$$X = f(q_u) \tag{2.3}$$

In the domain of parallel robotics the solutions to this problem are also known as ***assembly modes*** of the manipulator (Figure 2.3).

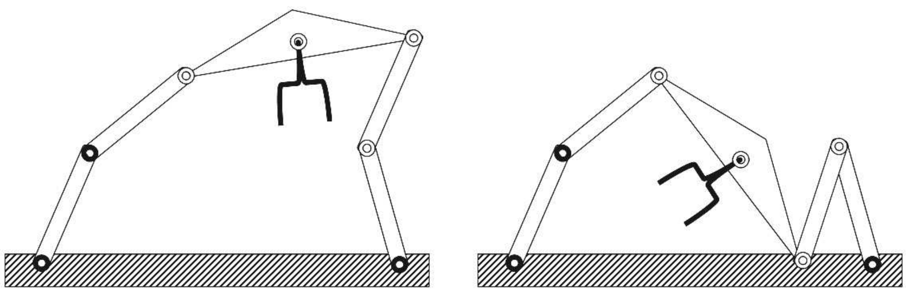


Figure 2.3: Different assembly modes of a planar parallel manipulator: the black joints correspond to $q_u$ and the pose is the same for both cases.

When for given parallel manipulator topology this solution is not unique (see Figure 2.3), to express analytically the platform coordinates $X$ as functions of $q_u$ can become quite a fastidious task. Different methods (analytical and numerical) for finding the possible solutions exist. Quite often they are not advantageous in terms of computational efficiency and thus - less suitable for use in real-time applications. Special attention has thus been paid to computational algorithms that comply with the time constraints imposed by real-time robotic applications.

### 2.2.1 Maximal number of direct kinematic problem solutions

Initially, the efforts were concentrated on finding all the direct kinematic solutions for any parallel manipulator, but as this task is quite demanding, methods were developed concerning families of parallel manipulators or particular parallel structures. For example, Dietmaier showed that for a *Gough* platform 40 assembly modes (i.e. direct model solutions) exist [13].

A systematic approach is the *polynomial direct kinematics*, proposed by Gosselin [14] and discussed by Merlet in [11], which consists in transforming the direct kinematic problem into solving the polynomial characteristic equation of the inverse kinematic equations.

### 2.2.2 A possible solution from a given manipulator configuration

In cases when not all of the direct kinematics solutions are needed, one of them could be found using numerical methods (e.g. the Newton-Raphson algorithm) or optimization techniques, for instance. The solution could be problematic, however, if the configuration is *singular* or nearly singular. The *singular configurations* are a major problem in robotics that has been treated in numerous research works, including the present one.

As parallel manipulators following trajectories with singularities are in the scope of this work, we shall devote the next chapter to the description and the classification of these particular configurations. We shall propose in Part II a robust technique for solving the mechanism kinematics in any of its existing configurations, singular or not.

# Chapter 3

# Singularities

*Singular configurations* (or singularities) in mechanics are configurations at which a given mechanism adopts an unpredictable behavior. This behavior is usually indicated by drastic changes in the mechanism performances, e.g. a velocity of a point or sustained external force at this point, going to infinity or zero. As shown further in this chapter, when dealing with parallel robots, the singularities could lie within the manipulator workspace or at its boundaries and lead to a local loss or "gain" of end-effector degrees of freedom. The uncontrollable behavior and performance deterioration due to singularities is undesirable for most practical applications.

The correct description and detection of singular configurations allow for their analysis. This is useful not only for predicting the MBS behavior, but for developing remedies in order to attenuate or eliminate their negative effects as well, e.g. following specific design rules, task planning, actuation and control strategies.

In this work, special attention is paid to the:

- Development of parallel manipulator kinematic and dynamic *modeling approach*, *robust* with respect to singularity *numerical* problems,
- Development of special parallel manipulator *actuation strategy* and subsequent *control applications*, *robust* with respect to singularity *physical* problems.

## 3.1 Problematics

Identifying the singular configurations is often a problem that is difficult to treat analytically. Many of the approaches in this direction are based on the analysis of the so-called *manipulator Jacobian* – the matrix that gives the mapping between the joint and end-effector velocities (see Section 3.2). As this matrix loses rank at the singularities, they are identified as corresponding to the roots of its determinant[1] [15]. This approach is practically sufficient for serial manipulators, but as we shall see, deeper analysis is required

[1]Provided that the Jacobian is a square matrix, which is valid for serial manipulators with up to 6 end-effector d.o.f. and a corresponding number of coordinates $q_u$.

for parallel manipulators, because singularities of different types exist. Furthermore, the difficulties in finding the roots of the Jacobian determinant for complex topologies are increased by the fact that the kinematic equations are nonlinear. With respect to this, methods based on screw theory and line geometry [16, 17], Grassman geometry [11] and differential geometry [18, 19] have been developed.

As the singular configurations of a parallel manipulator not only define the manipulator workspace limits, but can also appear during the manipulator motion within its workspace, their prior knowledge and analysis have preoccupied many scientists. For example, Gosselin and Wang defined the singularity loci for planar parallel manipulators [20]. Mohammadi Daniali, Zsombor-Murray and Angeles derived Jacobian matrices for two classes of planar parallel manipulators and identified from them three types of singularities [21]. Yang, Chen, Lin and Angeles analyzed the singularities of three-legged parallel robots, using a product-of-exponentials formula [22]. Tchon found the singularities of an Euler wrist [23]. Husty and Karger gave an overview of the self-motions and architectural singularities of Stewart-Gough platforms [24]. Choudhury and Ghosal studied kinematic and force singularities in parallel manipulators and closed-loop mechanisms and their relationship with the accessibility and control of such mechanisms [25]. Basu and Ghosal found a common tangent geometric conditions, useful for finding singularities of multi-loop platform-type spatial mechanisms [26].

Some of the researchers in this domain have been working on different remedies of *avoiding* singularities: specific mechanism synthesis [27, 28], proper trajectory generation modifications [29, 30], variational approaches [31], differential-form approaches [32], task pose optimization [33], etc.

We shall see that in the redundant actuation strategy proposed in this work, the singularity detection plays an important role as well. But it is not done as a process, separate from the mechanism kinematics and dynamics computation, and is not used to modify the manipulator task trajectories or to reduce the workspace, trying to avoid singularities. Thus, the *a priori* knowledge of the singular configurations that the parallel manipulators could exhibit is not an issue of this work. The proximity of singularities that could appear, when a given trajectory is followed, is in fact "sensed" by numerical means and/or via evaluation of actuator (joint effort) limits. This procedure is used in conjunction with special actuation strategy, developed in order to *pass through* these problematic configurations, preserving the manipulator motion and force capabilities.

As we already mentioned in the beginning of this chapter, the singular configurations in parallel manipulators provoke local "loss" or "gain" of end-effector degrees of freedom. It is sometimes said that for these specific (or *singular*) configurations the mechanism behavior "bifurcates" (takes an unpredictable form), hence the singularities are sometimes called "bifurcation configurations" instead.

In this research work and for parallel manipulators we shall distinguish two degree-of-freedom characteristics:

*Definition 5:* ***End-effector degrees of freedom*** $d_{eff}$ are degrees of freedom that the parallel manipulator can perform at its moving platform, the number of which can locally change, depending on the type of manipulator singularity, and can not exceed 6 (in case of spatial manipulator).

*Definition 6:* ***Multibody system (MBS) degrees of freedom (manipulator degrees of mobility)*** are global degrees of freedom of the parallel manipulator multibody structure, the number of which is constant for a given manipulator, depends on its architecture (number and type of independent actuated joints) and can be found using the following formula:

$$d_{mbs} = N^{joint} - m \tag{3.1}$$

where $N^{joint}$ is the number of system 1-d.o.f. joints and $m$ is the number of independent kinematic constraints, resulting for example from the closed loops in the multibody mechanical system. $d_{mbs}$ is positive for any moving mechanism, and *can exceed* the 6 degrees of freedom characterizing the spatial motion of the end-effector.

For serial manipulators we will obviously obtain for the second of the above definitions: $d_{mbs} = N^{joint}$.

## 3.2 Singularity classification through kinematics

Because of the constraints existing over the generalized velocities $\dot{q}$ in case of parallel manipulator topology, the mapping between $\dot{X}$ and $\dot{q}$ is feasible only over a sub-set of $\dot{q}$, the size of which corresponds to the number of manipulator end-effector degrees of freedom. A feasible sub-set is that of the so-called *independent velocities* $\dot{q}_u$, obtained by applying the *coordinate partitioning technique*[2] [4] we already mentioned. According to it, the vectors of joint generalized coordinates, their velocities and accelerations can be *partitioned* (divided) into vectors of $u$ *independent* and $v$ *dependent* variables:

$$q = \begin{bmatrix} q_u \\ q_v \end{bmatrix} ; \ \dot{q} = \begin{bmatrix} \dot{q_u} \\ \dot{q_v} \end{bmatrix} ; \ \ddot{q} = \begin{bmatrix} \ddot{q_u} \\ \ddot{q_v} \end{bmatrix} \tag{3.2}$$

Let $h(q) = 0$ be the system of holonomic constraint equations over the set of generalized joint coordinates $q$, arising for instance from the MBS kinematic loops. The constraint equations over the generalized joint velocities $\dot{q}$ can then be obtained by deriving $h(q) = 0$ with respect to time:

$$\frac{\partial h(q)}{\partial t} = 0 \ \Rightarrow \ J_c \ \dot{q} = 0 \tag{3.3}$$

with $J_c$ being the *constraint Jacobian* of the parallel manipulator.

If we derive with respect to time the system of equations $F(X, q) = 0$, similar to (2.3), but describing the kinematic model of a serial manipulator, we obtain the mapping between the joint and end-effector velocities, corresponding to the direct kinematic problem at velocity level:

$$\dot{X} = J(q) \ \dot{q} \tag{3.4}$$

[2]See Chapter 6 for more details.

In case of a parallel manipulator, we apply the partitioning $\{u, v\}$ to the Jacobian matrices $J$ and $J_c$, as well as to the constraint equations at velocity level, obtaining:

$$J = \begin{bmatrix} J_u & J_v \end{bmatrix}; \; J_c = \begin{bmatrix} J_{c_u} & J_{c_v} \end{bmatrix} \tag{3.5}$$

$$\dot{X} = J \; \dot{q} = J_u \; \dot{q_u} + J_v \; \dot{q_v} \tag{3.6}$$

and

$$J_c \; \dot{q} = 0 \quad \Longrightarrow \quad J_{c_u} \dot{q_u} + J_{c_v} \dot{q_v} = 0 \quad \Longrightarrow \quad \dot{q_v} = B_{vu} \; \dot{q_u} \tag{3.7}$$

where $B_{vu} = -J_{c_v}^{-1} J_{c_u}$, assuming that $J_{c_v}$ is of full rank for independent constraints.

Further, we obtain:

$$\dot{X} = (J_u + J_v \, B_{vu}) \;\; \dot{q_u} = J_m \; \dot{q_u} \tag{3.8}$$

in which the so-called *manipulator Jacobian* is defined as:

$$J_m \;\triangleq\; J_u + J_v \; B_{vu} = J_u - J_v \; J_{c_v}^{-1} J_{c_u} \tag{3.9}$$

It can be written in the form:

$$J_m(q_u) = \begin{bmatrix} J_{m_1}(q_u) & J_{m_2}(q_u) & ... & J_{m_n}(q_u) \end{bmatrix}^T \tag{3.10}$$

every line of which contains the partial derivatives of the functions $f_i(q_u)$ of (2.3) with respect to the set of independent generalized coordinates $q_u$:

$$J_{m_i}(q_u) \;\triangleq\; \begin{pmatrix} \frac{\partial f_i(q_u)}{\partial q_{u_1}} & \frac{\partial f_i(q_u)}{\partial q_{u_2}} & ... & \frac{\partial f_i(q_u)}{\partial q_{u_n}} \end{pmatrix} \tag{3.11}$$

On the other hand, for cases of parallel manipulator topology the system of equations (2.1) can be transformed after deriving it with respect to time as follows:

$$F(X, q_u) = 0 \quad \Longrightarrow \quad J_x \; \dot{X} + J_{q_u} \; \dot{q_u} = 0 \tag{3.12}$$

If we compare equations 3.8 and 3.12, we obtain $J_m = -J_x^{-1} J_{q_u}$, provided that $J_x$ is non singular.

Analyzing the two Jacobian sub-matrices in equation 3.12 gives the following principal types of singular configurations:

- **Serial singularities** - arise when $J_{q_u}$ becomes singular. This case corresponds to *a non-zero* joint velocity vector $\dot{q}_u$ (for example, $\dot{q} = [\; 0 \; 0 \; \dot{q}_{u3}]$ in Figure 3.1), *for which the end-effector does not (locally) move* ($\dot{X} = 0$). When the manipulator is in such configuration, it looses end-effector degrees of freedom. Conversely, it can bear infinite external forces or moments in certain directions (see Section 3.3.2). If a given configuration is a serial-type singularity, this sometimes corresponds to a manipulator workspace limit.

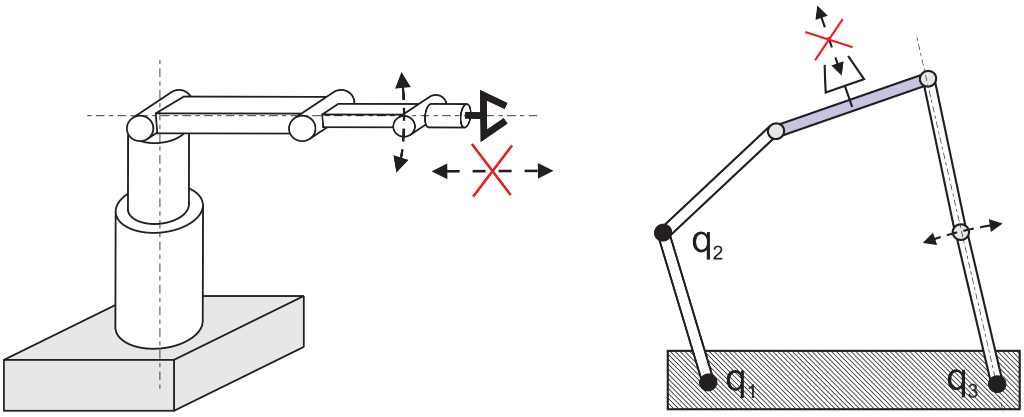



Figure 3.1: Comparison of a serial robot singularity (left) with serial-type singularity of a planar parallel robot (right). The alignment of two bodies in the serial robot chain and in the serial chain of one of the parallel robot legs locally cancels one end-effector degree of freedom ($q_1, q_2$ and $q_3$ - independent actuated variables).

- **Parallel singularities** - arise when $J_x$ becomes singular. In this case, *a non-zero vector* $\dot{X}$ *of end-effector velocities can be found, for which the actuated joint velocities are zero* (see eq.(3.12) and Figure 3.2). Around such configurations, the end-effector can have an infinitesimal motion without any change in the parallel manipulator legs. As a result some of the end-effector degrees of freedom become uncontrollable ("gain" of end-effector d.o.f. or end-effector *self motion*) and the manipulator can not withstand external loads in spite of the fact that all the actuators are locked. The parallel singularities are also known as *self-motion singularities*.

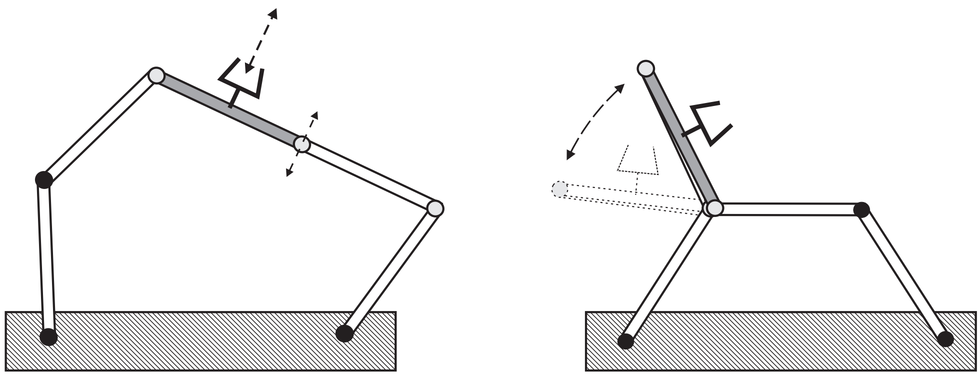


Figure 3.2: Parallel-type singularities of a planar parallel manipulator. Left: a local infinitesimal self motion of the end-effector. Right: a local end-effector macro self motion.

- **Mixed singularities** - appear when both $J_{q_u}$ and $J_x$ become singular (Figure 3.3, right). For such configurations, neither the end-effector moves when joint velocities exist, nor can some of its degrees of freedom be controlled.

The above singularities correspond to the physical configuration of the mechanism, whatever the choice of the end-effector coordinates $X$.

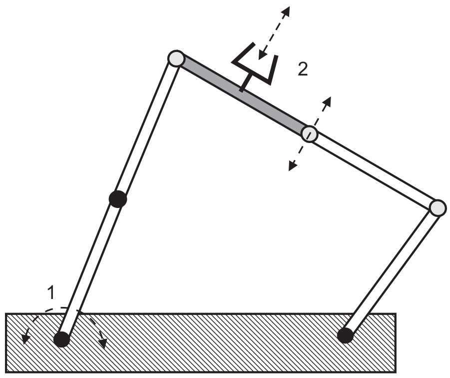



Figure 3.3: Mixed-type singularity of a planar parallel manipulator (actuated joints filled in black): a mix of actuated joint motion without end-effector motion (No.1 in the figure) and uncontrollable end-effector motion (No.2 in the figure).

- **Mathematical singularities**[3] – depend on the way the kinematic relations between $X$ and $q$ are formulated: the type of generalized coordinates used, the geometrical description of the kinematic constraints. These singularities do not necessarily have a physical meaning and can be eliminated by an appropriate mathematical modeling reformulation. For example, if a spherical joint exists as an element of a parallel manipulator leg serial chain[4] and its three degrees of freedom are represented as a sequence of three successive rotations about mutually orthogonal axes (known in mechanics as *Tait-Bryan* or *nautical rotational* angles (see [1, 3] for details)), this leads to cases of singularities in the input-output rotation matrix when the intermediate angle equals $\pi/2$, even if the real physical joint does not introduce any physical singularity problems.

A dual way of detecting the previous singular configurations is considering the manipulator static force/torque balance equation [11]. In a static situation ($\dot{X} = 0, \dot{q}_u = 0$), we can write the following system of equations:

$$W^T \dot{X} + \tau_u^T \dot{q}_u = 0 \tag{3.13}$$

whatever $W$ and $\tau_u$, where $W$ is the vector of external forces and torques applied to the end-effector, often called *wrench*, and $\tau_u$ is the vector of joint efforts acting on the independent joints $q_u$. It is straightforward to express $W$ in terms of $\tau_u$, using the definition of $J_m$ from (3.12):

$$W^T \dot{X} + \tau_u^T (J_{q_u}^{-1} J_x) \dot{X} = 0 \quad \Rightarrow \quad W = (J_x^T J_{q_u}^{-T}) \tau_u \tag{3.14}$$

which gives the well-known form of the static force balance equations:

$$W = (J_m^{-1})^T \tau_u = J_m^{-T} \ \tau_u \tag{3.15}$$

[3]also known as *representational singularities* [34].

[4]Spherical joints are often used as anchor joints for the moving platform in 3D parallel robots.

valid for a well-conditioned $J_{q_u}$. This system of equations is linear in terms of $\tau_u$ and admits a unique solution, except if $J_m^{-1}$ becomes singular. Then the manipulator will not be in equilibrium and its structure will exhibit infinitesimal motions (parallel singularity). In other words, in order the manipulator to be able to withstand a given wrench $W$ in a nearness of a singular configuration, the joint forces would have to be very large, which could even lead to a structural breakdown.

## 3.3 Velocity and force ellipsoids

A meaningful way to describe the robot singularities is to use the concept of *velocity* and *force ellipsoids* [35], which quantify the manipulator kinematic performances and static force capabilities, respectively, in different directions of the task space.

In the language of mathematics, an *ellipsoid* $\epsilon^n$ is a geometric surface, defined in a sub-set of the n-dimensional Euclidean space $\Re^n$, all plane sections of which are either ellipses or circles. For the particular, common case of 3-dimensional Euclidean space $\Re^3$, it can be redefined as a surface $\epsilon^3$ (Figure 3.4), consisting of points $(x, y, z) \in \Re^3$ such that:

$$\epsilon^3 : \left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 + \left(\frac{z}{c}\right)^2 = 1 \tag{3.16}$$

for some constants $a, b, c > 0$.

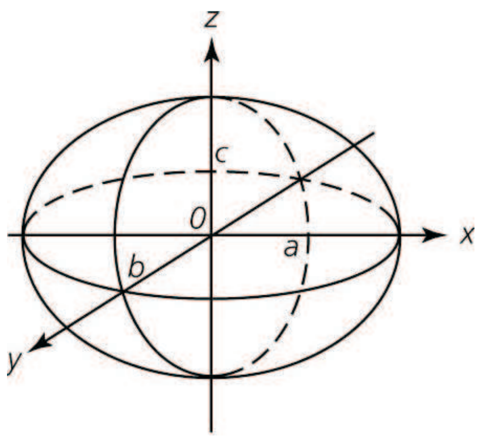



Figure 3.4: Ellipsoid in the 3-dimensional space.

A unit sphere $\varsigma_u$ can be straightforwardly obtained from the ellipsoid definition (3.16), if we take $a = b = c = 1$:

$$\epsilon^3 \equiv \varsigma_u : x^2 + y^2 + z^2 = 1 \ , \quad \|\varsigma_u\| = \sqrt{x^2 + y^2 + z^2} = 1 \tag{3.17}$$

Hence, the ellipsoid $\epsilon^3$, being a surface defined by a set of points $\Psi$ that obey (3.16), can be considered as an image of the unit sphere $\varsigma_u$, obtained by means of a linear mapping $f_m$:

$$f_m : \varsigma_u \mapsto \epsilon^3 \ \Rightarrow \ \epsilon^3 = \{\Psi : \Psi = f_m \varsigma_u, \|\varsigma_u\| = 1\} \ , \quad f_m = diag(\frac{1}{a}, \frac{1}{b}, \frac{1}{c}) \tag{3.18}$$

### 3.3.1 Velocity ellipsoid

For *serial* manipulators with revolute joints, the *velocity ellipsoid* $\epsilon_v$ can be defined as the image of a unit joint space velocity sphere, mapped into the task space by the manipulator Jacobian, i.e.:

$$\epsilon_v = \left\{ \dot{X} : \dot{X} = J\ \dot{q}, \|\dot{q}\| = 1 \right\} \tag{3.19}$$

Performing the singular value decomposition (SVD) of $J$:

$$J = U\ \Sigma\ V^T \tag{3.20}$$

the velocity ellipsoid becomes:

$$\epsilon_v = \left\{ \dot{X} : \dot{X} = U\ \Sigma\ V^T \dot{q}, \|\dot{q}\| = 1 \right\} \tag{3.21}$$

Its principal axes are defined by the columns $u_i$ of the left unitary matrix $U$. The lengths of these axes are given by the singular values $\sigma_i$, that form the diagonal matrix $\Sigma$. If the rank of $J$ is less than its maximal value at a certain configuration, then $J$ is singular and the ellipsoid is said to be "degenerate". In this case one or more of the ellipsoid axes have zero length and the ellipsoid will have zero volume. The degenerate ellipsoid means an end-effector local motion "incapability" in one (or more) of the principal axes directions (see the serial manipulator configuration in Figure 3.1 for instance).

In case of parallel manipulators with revolute joinyt, the velocity ellipsoid is redefined as an image of a unit sphere formed by the subset of *independent* joint velocities:

$$\epsilon_v = \left\{ \dot{X} : \dot{X} = J\ \dot{q}, J_c\ \dot{q} = 0, \|\dot{q_u}\| = 1 \right\} \tag{3.22}$$

A *constrained velocity ellipsoid* is then written in the following manner:

$$\epsilon_{v_c} = \left\{ \dot{X} : \dot{X} = J_m\ \dot{q_u}, \|\dot{q_u}\| = 1 \right\} \tag{3.23}$$

assuming that $J_{c_v}$ is of full rank.

### 3.3.2 Force ellipsoid

The static force balance (equation (3.15)), to which the notion of *force ellipsoid* refers, can be considered as a dual to the direct kinematics (3.4) of the manipulator. It considers the balance between the joint forces/torques $\tau$ that have to be applied to the manipulator joints in order to withstand an external wrench $W$, applied to the end-effector. For static cases the existing internal friction force/torques and gravity components are often neglected.

Analogically, as a dual to the velocity ellipsoid, the *force ellipsoid* $\epsilon_f$ can then be defined as the end-effector force/torque space image of a unit sphere in the joint effort space [35]. For a serial manipulator, it is written as:

$$\epsilon_f = \left\{ W : J^T W = \tau, \|\tau\| = 1 \right\} \tag{3.24}$$

Applying again the SVD to $J$, we obtain:

$$\tau = V\ \Sigma^T\ U^T\ W \tag{3.25}$$

Assuming that $J$ is square[5] and the system configuration is not singular, the force ellipsoid definition then becomes:

$$\epsilon_f = \left\{W : W = U\ \Sigma^{-1} V^T \tau, \|\tau\| = 1\right\} \tag{3.26}$$

Comparing (3.21) and (3.26) helps to see that the principal axes of the force ellipsoid are the same as those of the velocity ellipsoid, but their lengths are reciprocal. Recalling the velocity ellipsoid degeneracy singularities, revealed by one (or more) diagonal entries of $\Sigma$ being zero, would imply that in such configurations the force ellipsoid is infinite along the corresponding directions, found in $U$. In such singularity the mechanism can bear external efforts $W$ of very large amplitudes, so in spite of the fact that end-effector motions are canceled, such singularities (equivalent to serial-type ones of the classification of Section 3.2) can be advantageous in terms of static effort balance.

For parallel manipulators, to the vector $\tau$ have to be added the components of the constraint (joint) forces/torques that maintain the existing kinematic loops of the parallel topology closed. If we recall the well-known semi-explicit form of the direct dynamics of such manipulators (discussed in detail in Section 7.4):

$$M(q)\ddot{q} + c(q, \dot{q}, g, ...) + J^T W = \tau + J_c^T \lambda \tag{3.27}$$

in which:

- $M$ is the symmetric positive definite generalized mass matrix of the system,
- $c$ is a vector that contains the Coriolis, centrifugal and gravity terms as well as the contribution of external resultant forces acting on the manipulator constitutive bodies, except on its end-effector,
- $g$ is the gravity vector,
- $\lambda \in \Re^m$ represent the vector of *Lagrange multipliers*, associated with the *explicit* constraints over the manipulator topology,

and consider a static case ($\dot{q} = 0, \ddot{q} = 0, c(q, \dot{q}, g, ...) = 0$), for which the internal friction forces/torques are neglected, then the static force balance equations of the manipulator will read:

$$\tau = J^T\ W - J_c^T \lambda \tag{3.28}$$

The vector of *Lagrange multipliers* associated with the constraints corresponds to the components of the constraint forces in the joints, maintaining the manipulator structure closed.

[5]meaning that the number of actuators equals the number of end-effector d.o.f.

Applying to (3.28) the coordinate partitioning into *independent* and *dependent* coordinates , we obtain:

$$\begin{aligned} \tau_u &= J_u^T W - J_{c_u}^T \lambda \\ \tau_v &= J_v^T W - J_{c_v}^T \lambda \end{aligned} \tag{3.29}$$

Considering the joints, corresponding to the dependent coordinates as passive (or non-actuated, i.e. $\tau_v = 0$), we can substitute $\lambda$ from the second set of equations into the first one (assuming that $J_{c_v}$ is non singular), thus obtaining:

$$\tau_u = \left(J_u^T + B_{vu}^T J_v^T\right) W = J_m^T W \tag{3.30}$$

Hence, the constrained force ellipsoid can be written as:

$$\epsilon_{f_c} = \left\{W : W = J_m^{-T} \tau_u, \|\tau_u\| = 1\right\} \tag{3.31}$$

## 3.4 Singularity classification through velocity and force ellipsoids

Taking into account the duality of the velocity and force ellipsoids of parallel manipulators, we can distinguish (as already proposed in [35, 36]) two main classes of singularities[6]:

- **Velocity singularities**[7] – corresponding to those configurations of the mechanism, at which $J_m$ loses rank. At such configurations the velocity ellipsoid degenerates and the force ellipsoid is infinite along one or more of its principal axes. The mechanism can bear infinite loads, but finite joint motion does not produce any task motion. This kind of singularity can occur for both serial and parallel manipulators. It corresponds to the serial-type singularities, defined earlier in this chapter (see Section 3.2 and Figure 3.1).

- **Force singularities**[8] – corresponding to those configurations of the mechanism, at which $J_{c_v}$ loses rank. At these configurations the force ellipsoid becomes degenerate and the velocity ellipsoid is infinite along one or more of its principal axes. As a result, the task frame can move even if all the active joints are locked. This type of singularity is specific to parallel robotic structures. It corresponds to the parallel-type singularities, defined earlier in this chapter (see Section 3.2 and Figure 3.2).

In conclusion to this chapter's matter, we would mention the extensive singularity classification, established by Zlatanov, Fenton and Benhabib [37]. The correspondence between the general types of singularity we already described here and those of the mentioned

[6]This distinction can also be made on the basis of the definition (3.9) of $J_m$

[7]Named "mechanically advantageous singularities" in [36]

[8]Named "mechanically unstable singularities" in [36]

researchers is not difficult to notice. In terms of example, let us compare the singular configurations of a 6-d.o.f. parallel manipulator, depicted in Figure 3.5, with those of Figures 3.1 and 3.2.

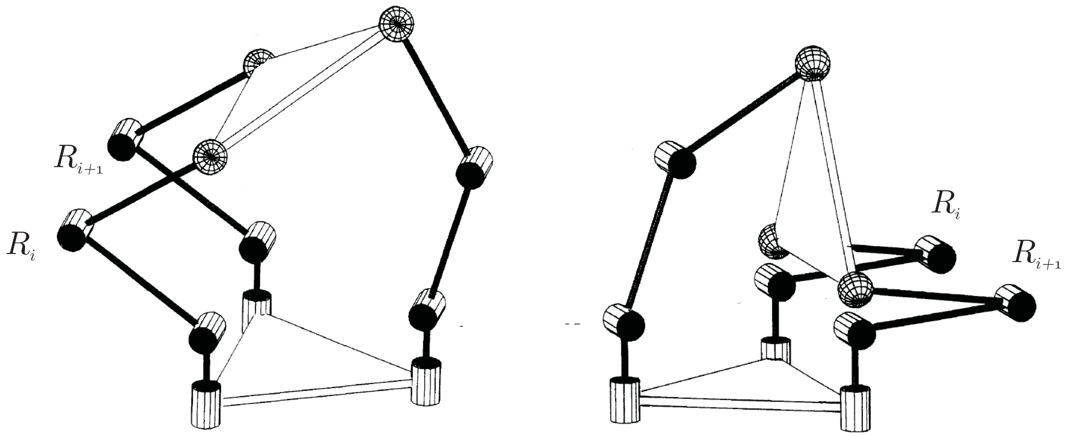



Figure 3.5: Singular configurations of a 6-d.o.f. parallel manipulator (according to Zlatanov, Fenton and Benhabib [37]). Parallel singularity (left) and serial singularity (right) appear when the indicated revolute joints $R_i$ and $R_{i+1}$ are passive (non actuated).

# Chapter 4

# Redundant actuation

The application of *redundant actuation* in order to eliminate force singularities in the workspace of parallel manipulators when they follow desired task trajectories, can be considered as a core of the present research work. A special actuation strategy on the basis of force singularity detection was developed by the author for this purpose (see Chapter 8).

In this chapter, definitions and principal classification of the redundant actuation will be discussed at first. Then, a state of the art of redundant actuation applications will be presented. Finally, existing redundancy solution methods will be briefly reviewed.

## 4.1 Definitions and Classification

The redundancy in robotics is of significant importance, because of its advantages when augmented task versatility and manipulator performances are sought. To our knowledge, this subject has not drawn significant scientific attention till the early 90s, with the interest growing mostly during the last decade.

Non-redundant robots, be they serial or parallel, perform well over a certain range of task operations, corresponding to the limitations of their structural and actuation characteristics. The latter define the manipulator performances in terms of load handling, precision and rapidity, as well as the workspace aspect and volume. If, for instance, avoidance of obstacles in the workspace is necessary for a successful task accomplishment, depending on the robot architecture and the obstacle location this avoidance could be proved impossible.

Redundant manipulators, for their part, possess "additional inputs" that offer means for improving the above-mentioned performances and increasing the manipulator versatility. In particular, redundancy in parallel manipulators recently attracted research interest, revealing such possibilities as: active modulation of the manipulator internal pre-load state ("internal stiffness"), fault tolerance, reconfigurability, singularity elimination, joint backlash (clearance) elimination. The aspects of redundancy in parallel manipulators is in the main scope of this work as well, its principles being advantageously exploited. We shall proceed with its definition.

### 4.1.1 Definitions

When considering serial manipulators, a *kinematic redundancy* is defined. It consists in adding to the manipulator serial structure new bodies with their corresponding joints and actuators, in order to increase the multibody system degrees of freedom (manipulator mobility), discussed in the previous chapter, to an extent greater than the number of end-effector degrees of freedom. An example of a serial manipulator with redundant kinematics is shown in Figure 4.1.

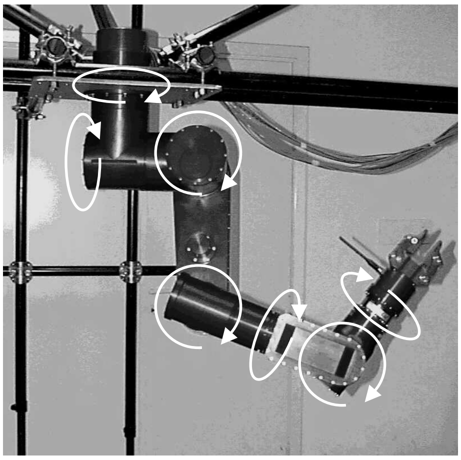


Figure 4.1: Kinematically redundant 7-d.o.f. serial manipulator, developed as a part of AMADEUS HUMAN COMPUTER INTERFACE at University of Genoa, Italy

With respect to this, the following definition can be formulated:

*Definition 7:* A ***kinematic redundancy*** of a serial manipulator is the difference, represented by a positive integer, between the number of multibody system degrees of freedom and the number of end-effector degrees of freedom.

It is worth mentioning that a non-redundant serial robot could be considered redundant with respect to a given task or set of tasks. For example, if a task consists in sliding a cube on a plane, maintaining contact between one of its faces and the plane surface, this would allow the object to exhibit three degrees of freedom at most (two translations and one rotation) and require the same three d.o.f. from the end-effector. If a PUMA robot is used for this task, it would be redundant with respect to it, because its architecture assures 6 d.o.f. at the end-effector. But we will limit the notion of kinematic redundancy to the definition stated above, as to our opinion this characteristic should reflect the presence of an "excess" in the manipulator architecture with maximum 6 (3 for the planar case) d.o.f., and not possible task d.o.f. reduction.

The kinematic redundancy allows for achieving various improvements, e.g. increasing the robot workspace volume, avoiding obstacles in it, increasing the manipulator dexterity, its fault tolerances, etc. An interesting example of highly redundant serial chains are the so called *snake robots* (Figure 4.2), often categorized as *hyper-redundant*, i.e. possessing more than 10 multibody system d.o.f. Hybrid snake robots exist as well. The one shown

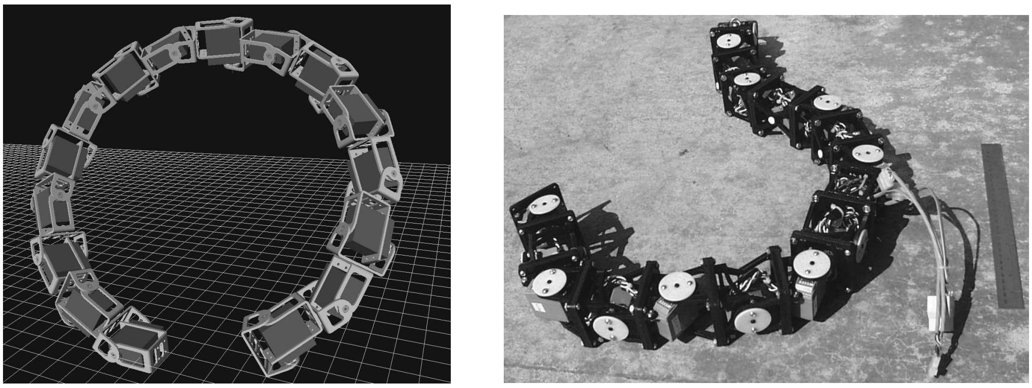


Figure 4.2: Hyper-redundant serial chain *snake* robot. Left: 3-D virtual reality model; Right: a developed prototype

here is a serial-chain micro-robot (Figure 4.3 - left), every element of which is a miniature parallel mechanism itself (Figure 4.3 - right). As another example of hyper-redundant

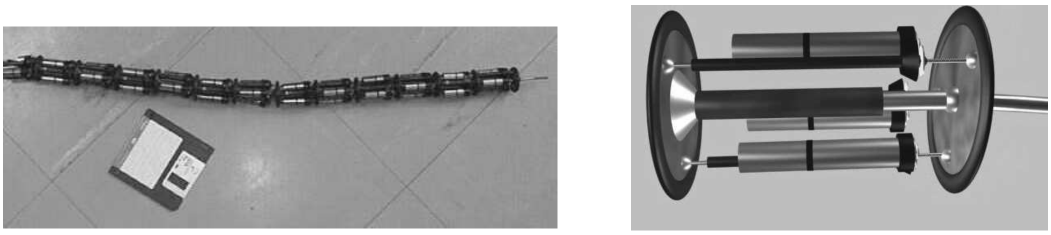


Figure 4.3: Hybrid snake robot, designed in the frame of a Copernicus project at the University of Metz (France).

structures can be cited the *articulated truss* structures, mentioned in Chapter 1 (Figure 1.7).

It is important to note that this type of redundancy does not cancel serial singularities [38]. As already pointed in the previous chapter, the latter provoke local end-effector d.o.f. loss, but the end-effector degrees of freedom are not influenced by kinematic redundancy. It is the kinematic redundancy, on the contrary, that locally increases in case of singular configuration. The redundant kinematics changes the workspace volume, but not the singularity manifold dimension (see Figure 4.4).

In contrast with the kinematic redundancy, the redundancy in *parallel* manipulators concerns mainly their static and dynamic performances, influencing their force capabilities, manipulability and control under external and internal (joint friction, joint damping) loads. It can be used to eliminate force (parallel-type) singularities, present in the

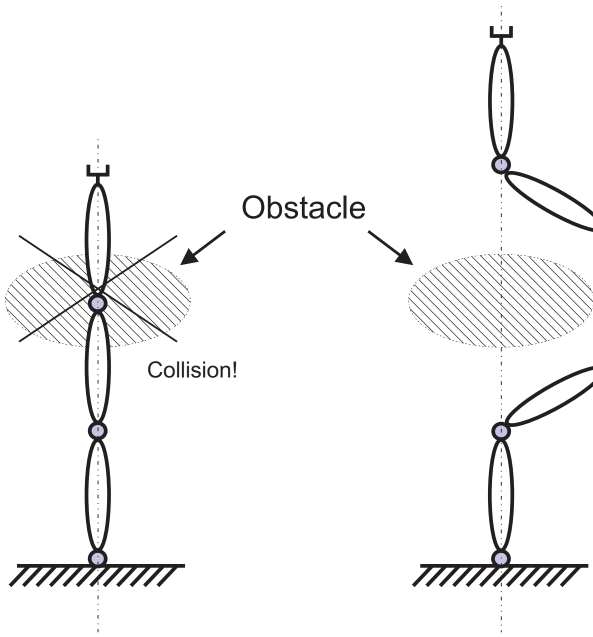



Figure 4.4: Kinematic redundancy and serial singularities. The 3-d.o.f. planar serial robot (3 revolute joints of mutually parallel axes) to the left has the same singularity for the configuration shown, as its redundant 5-d.o.f. analogue (5 revolute joints of parallel axes) to the right. However, the latter has larger workspace, allowing obstacle avoidance, for example.

workspace. This type of redundancy is often named ***redundant actuation*** or ***actuation redundancy***:

Definition 8: A ***degree of actuation (force) redundancy*** $d_{AR}$ ***(***$d_{FR}$***)*** of a *parallel* manipulator is the difference, represented by a positive integer, between the number of its actuators (actuated joins) and the number of its MBS degrees of freedom.

According to this definition, the degree of force redundancy can be calculated as follows:

$$d_{AR} = N_{act} - d_{mbs} \tag{4.1}$$

where $N_{act}$ is the number of independent actuators (actuated joints) and $d_{mbs}$ – the number of degrees of freedom of the multibody system, defined in Section 3.1. $d_{AR}$ is positive if redundant actuation is present, zero if the manipulator is not redundantly actuated and negative, when it is under-actuated. Under-actuated manipulators exist (e.g. the under-actuated grasping applications of [39], allowing the robotic hand to adjust itself to an irregularly shaped object without complex control strategies and sensors), but are not in the scope of this research work, therefore we shall not mention them further.

A typical example of redundant actuation, existing in the nature, is the human arm, which has a total of 29 muscles (actuators), but can perform "only" 6 degrees of freedom at its "end-effector" – the human hand, giving rise to a degree of actuation redundancy, equal to 23! The strong ”excess” of actuation in the arm serves to augment its force performances and workspace, so the nature says all.

Sometimes the role of *active wrist* in hybrid manipulators is played by redundantly actuated parallel mechanisms. Here we could cite the hybrid manipulator of the McGill university in Canada (Figure 1.12), the wrist of which is a spatial 4-d.o.f. parallel manipulator that transforms into 3-d.o.f. redundantly actuated one, when the sliding of the central rod is locked (see Figure 4.5, left).

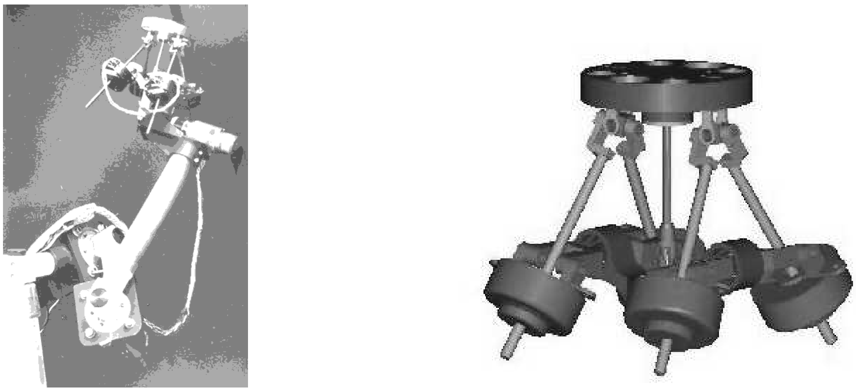


Figure 4.5: The redundantly actuated *active wrist* of the IRIS C3 hybrid manipulator (McGill University - Canada).

### 4.1.2 Redundant actuation classification

According to [40], the redundant actuation ($d_{AR} > 0$) can be divided into three main categories:

1. Actuation of some of the passive joints in the mechanism parallel branches (Figure 4.6, right).

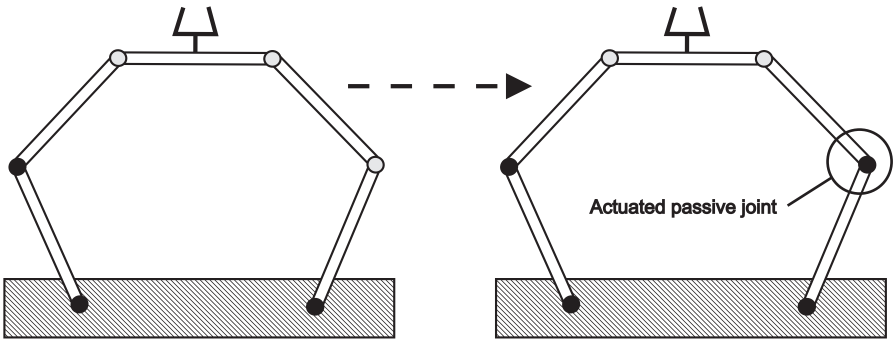



Figure 4.6: Redundant actuation of a 3-d.o.f. planar parallel manipulator (actuated joints in black). The redundancy is achieved actuating a passive joint in the parallel branch.

2. Addition of more parallel branches with actuators in order for the total number of the latter to exceed the minimal one, required for proper actuation of the mechanism (Figure 4.7, left)

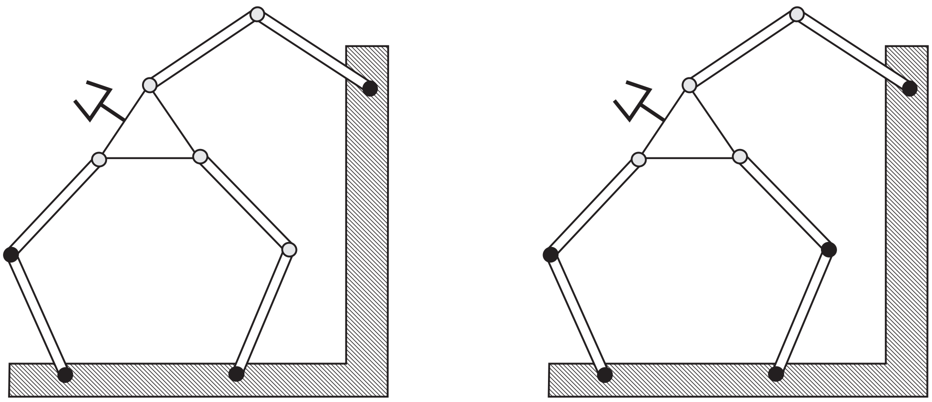


Figure 4.7: Redundantly actuated 3-d.o.f. planar parallel manipulator. Left: the redundancy is achieved adding a parallel branch with a motor; Right: redundancy of mixed type, obtained actuating passive joints and adding additional parallel branch (actuated joints in black)

3. Redundant actuation that is a mix of the previous two types (Figure 4.7, right)

It is easily understandable that if force redundancy by means of adding more parallel branches is used, it would tend to reduce the manipulator workspace due to the additional kinematic constraints added by every new parallel branch. Furthermore, from an "economic" point of view, more resources would be required to add a new branch with its actuation to an existing parallel structure, than just to add an actuator on a passive joint, suitable (accessible) for actuation. That is why, in this research work redundant actuation of the first type: actuation of available passive joints of a predefined parallel manipulator topology, was chosen as a basis of the proposed actuation strategy that we shall reveal later.

# 4.2 State of the Art

## 4.2.1 Force redundancy applications

Currently, several fields of application of redundant actuation in parallel manipulators can be enumerated:

- *Internal stiffness modulation*

In general, the principal goal of this group of applications is to take profit from force redundancy in order to actively manage the manipulator internal stiffness and have it reacting conveniently to external loads applied. For example, Yi and Freeman [41] investigated and applied the antagonistic internal load properties, inherent to redundantly actuated closed-loop mechanisms, demonstrating how the mechanism motion, its effective stiffness and internal load state can be controlled independently, using the actuator redundancy. They proposed three different applications of this type of control, stressing on the optimal placement of the required actuation inputs with respect to joint/torque limits. Kock and Schumacher in [42] examined a 2-d.o.f. parallel manipulator with actuation redundancy for high-speed and stiffness-controlled operation. They outlined the force redundancy advantages, performing a force transmission analysis and developing a novel control scheme that guarantees a lower bound of the end-effector stiffness. Chakarov [43] investigated parallel manipulators, consisting of a basic anthropomorphic kinematic chain and parallel chains with redundant linear actuators. He proposed approaches for specification of a desired compliance along a given direction and of a biggest compliance in the operational space, and developed a scheme for stiffness control on their basis.

- *Fault tolerant manipulators*

In this group of applications, the redundancy is simply used to compensate for the possible failure of some of the principal manipulator actuators. In connection with this aspect, Notash and Huang proposed in [44] an extensive parallel manipulator failure analysis at a component level (link and joint failures), subsystem level (branch failure), and system level (parallel device failure). In addition to that, they discussed the failure of parallel manipulators as a result of the loss of degree of freedom, loss of actuation, loss of constraint, special configurations (singularities), as well as branch interference, and defined two criteria for identification of optimum fault-tolerant configurations of parallel manipulators with simple redundant actuation. Chen et al. [45] looked at the dynamic fault-tolerance ability of redundantly actuated robots, proposing a Weighted Null-space Algorithm with Torque Adjustment (WATA) in order to increase it, and a joint torque redistribution scheme to carry out fault-tolerant control.

- *Robot cooperative work and multi-finger grasping*

Two serial manipulators, handling a common payload, can be thought of as one redundantly actuated parallel manipulator. The cooperative manipulator operation was commented by Zheng and Luh in [46], for example. They proposed algorithms for solving this specific case of redundant actuation.

The multi-fingered grasping [47] can be considered as an example of redundant actuation, as long as the manipulated object remains grasped. The grasping can be viewed as a particular, reduced-scale cooperative object manipulation, every finger of the grasping device being a mini-manipulator, usually of serial topology. This situation is of even more complex nature than the classical robot cooperative work, because the object handling is generally ensured by friction forces between the fingertips and the manipulated object. These friction components are created by forces of pressure (exerted by the fingertips) that are normal to the contact plane and must be precisely controlled. Hence, additional unilateral contact constraints have to be taken into account when solving the actuator redundancy problem. Of course, independently of the grasping, the fingers themselves could be redundant manipulators – take [48] for example.

- *Reconfigurable manipulators*

This is a relatively new issue, as most of the researchers were till now interested basically in kinematically redundant serial and non-redundant parallel reconfigurable robots [49, 50], concentrating on the design of modular robots that could reach reconfigurability only if physically relocating the actuators and/or changing the manipulator topology [51]. Just recently – in 2004 – Fisher et al. [52] presented the design of a reconfigurable planar parallel manipulator (RPPM) (see Figure 1.10, right), designed to act as a testbed manipulator for theories on redundant actuation in parallel robotics. The manipulator can reconfigure into three different revolute-joint mechanism types: a 2-branch 2-d.o.f. 5-bar mechanism, a 2-branch 3-d.o.f. 6-bar mechanism and a 3-branch 3-d.o.f. 8-bar mechanism (Figure 4.8). The authors included some criteria and constraints of the design and commented on how their designed prototype relates to the latter.

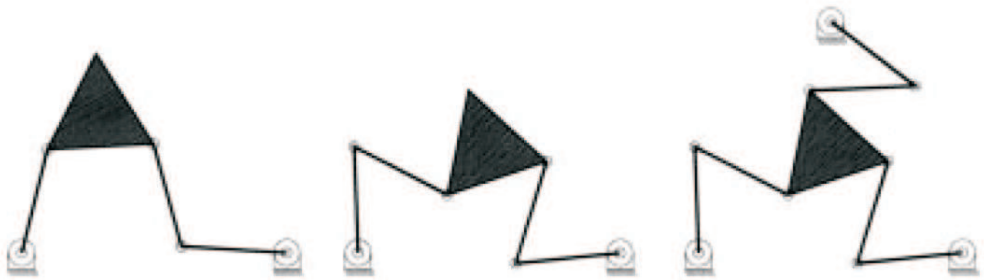


Figure 4.8: The three desired mechanisms, obtained by reconfiguration of the RPPM of Firmani and Podhorodeski of Figure 1.10 (image taken from [52]). The base and the moving platform are counted as "bars".

- *Singularity elimination*

Possible elimination of singular configurations is one of the main advantages of having redundant actuation on parallel manipulators. During the last decade some researches, the author of this work included, concentrated on this problematics. The redundant actuation allows for elimination of singularities present in the parallel manipulator workspace. As already mentioned in Chapter 3, the singularities locally degrade the robot external load handling capabilities (output effort performances) and could lead to mechanism locking, actuator saturation and even physical damage. A thorough analysis of the redundancy in parallel manipulators was given by Dasgupta and Mruthyundjaya in [38], where they studied it as a series-parallel dual concept of kinematic redundancy in serial manipulators, describing its implications in the kinematics and dynamics of parallel robots. They demonstrated the actuation redundancy effective utilization in reduction and elimination of what they called *static singularities*[1], showing on numerical studies of two parallel manipulators that an addition of a single degree of redundancy (one parallel branch with additional actuator) can reduce singularities drastically and improve the quality of the workspace to

[1]which correspond to the *force singularities*, defined in Section 3.3

a great extent, the cost being only its slight reduction due to the additional kinematic constraints added via the new leg in the parallel structure. In the previous chapter we discussed the fact that the singularities in parallel manipulators are actuator-dependent. Firmani and Podhorodeski used in [53] the concept of reciprocal screw quantities and kinematic geometry to determine the feasibility of the existence of force singularities of planar parallel manipulators. They showed that one order of infinity of force singularities is eliminated for each added actuator and considered different actuator location redundant schemes, investigating the singularity modifications depending on them. Krut et al. [54] performed an analysis of the velocity isotropy of parallel mechanisms with actuation redundancy, emphasizing the convenience of the latter for an improvement of their velocity performance indexes, i.e. its positive effect on the reduction of the degeneracies caused by velocity singularities in parallel mechanisms. They mentioned the duality between the velocities and forces in parallel manipulators, and discussed on the relevancy of using the Jacobian matrix condition number as a quality index for velocity and force isotropy, respectively.

The strategy of force singularity elimination by suitably choosing the actuator locations is used by the author of the present research work, the first stages of which were presented in in [55]. By eliminating the force singularities, one could achieve better workspace exploit and augment the robot force performances, when external loads have to be managed in the presence of joint friction. As mentioned earlier in Section 4.1.2, redundancy is achieved by means of actuating accessible passive joints of the existing structure, and not changing the manipulator architecture by incorporating extra legs (parallel branches). This approach cancels the problem of possible reduction of the manipulator workspace due to the additional kinematic loops that would be introduced. In [36], we developed a methodology for application of trajectory-dependent sequences of non-redundant actuations that locally eliminate actuator singularities and eventually lead to redundancy[2]. We commented as well on the control of redundantly actuated parallel mechanisms, testing different control algorithms that effectively use the chosen force redundancy solutions schemes. These developments will be presented in detail in this work as well, because they constitute a significant part of its theoretical and practical achievement.

- *Joint clearance (backlash) elimination*

This aspect represents one of the latest redundant actuation research directions. The joint clearances provoke instable mechanism operation and a lack of control precision. The main idea behind the effective usage of the redundant actuators in this case is to counterbalance the additional parasite degrees of freedom that appear (in a sudden manner) in the joints, causing impulse disturbances and instant losses of assembly rigidity of the manipulator.

Many researchers have worked on the aspects of clearance modeling and mechanical analysis. For example, Dubowsky and Freudenstein [56] performed in 1971 a dynamic analysis of mechanical systems with clearances. Sarkar, Ellis and Moore commented in

[2]due to the physical impossibility of changing actuator places during motion.

[57] on backlash detection in geared mechanisms. They showed that, by modeling backlash as microscopic impact, its presence can be detected and possibly measured using only simple sensors. Bauchau and Rodriguez presented in [58] a generalized, versatile approach for analysis and modeling of unilateral contact conditions involving backlash, freeplay and friction. Garcia Orden gave in [59] a methodology for the study of typical smooth joint clearances in multibody systems, proving with numerical applications that it is very stable in long-term simulations with relatively large time step sizes and, hence, promising in terms of efficiency and robustness for the numerical analysis of real joints with clearances. Schwab, Meijaard and Meijers [60] studied the dynamic response of mechanisms and machines affected by revolute joint clearance, developing an impact model with a procedure to estimate the maximum contact force during impact and showing how the compliance of the links or lubrication of the joint smooths the peak values of the contact forces. Zhu and Ting [61] worked on the uncertainty analysis of planar and spatial robots with joint clearances. They established, based on the probability theory, a general probability density function of the endpoint of planar robots, calculating distribution functions of the robot endpoint that provide a convenient way to obtain probability values of the joint clearance and position repeatability for a desired type of tolerance zone. Parenti-Castelli and Venanzi commented the effects of joint clearance on the end effector pose accuracy of serial and parallel manipulators [62], reporting simulation results of several serial and parallel manipulators with clearance in the joints and comparing their positioning accuracy.

Only recently, some researchers concentrated on the application of redundant actuation in order to cope with backlash problems. Chen and Yao [63] worked on systematic methodology for the drive train design of redundant-drive backlash-free robotic mechanisms (RBR mechanisms). An interesting research work is that of Muller [64]. Giving a general formulation for the dynamics of redundantly actuated parallel manipulators, he derived an explicit solution in terms of a single pre-stress parameter for the special case of simple redundancy (one redundant actuator) and proposed a computational efficient open-loop pre-stress control that eliminates the mechanism backlash. The author proved the control algorithm simplicity and demonstrated its applicability in real-time on a planar 4RRR manipulator.

Finally, without pretending to cover all the possible cases that could exist, let us cite two more fields of application of redundant actuation analysis, modeling and resolution, concerning the domain of *biomechanics*:

- *Muscle force determination*

The animal and human bodies are highly redundantly actuated, as already mentioned in the example of the human arm, given in the previous section. The efforts of the researchers are concentrated mostly on the dynamic analysis and quantification of muscle efforts [65, 66, 67] using experimentally measured kinematic data and optimization techniques to determine them, in spite of the fact that some have worked on methods for solution of the redundancy problem in biomechanics mainly, even targeting computational efficiency [68].

- *Legged locomotion*

The aspects of legged locomotion (Figure 4.9) have gained significant interest and concentration of research effort, because of their importance with respect to applications in medicine, biomechanics and robotics. During certain phases of the gate – the *double stand phases* (both feet are in contact with the ground), namely, the structure, formed by the legs can be considered as a redundantly actuated closed-loop mechanism. The research

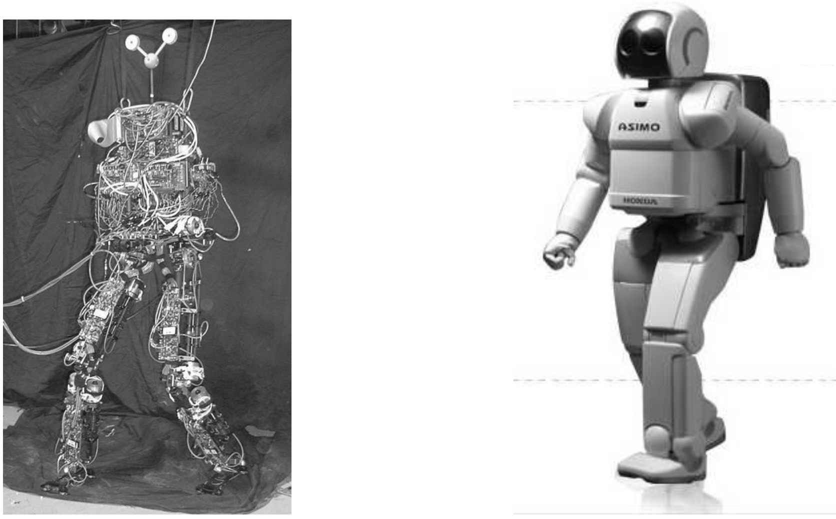



Figure 4.9: Legged locomotion robotic examples. Right: the famous ASIMO robot of Honda Motor Corporation (image courtesy of Honda Motor Co., Ltd.)

results concerning the previous item find direct application in this latter research direction as well. The synergy of different sciences: biology, medicine, physics, mechanics and electronics, contributes a great extend to the analysis and modeling of the legged locomotion phenomena, thus opening a multitude of perspectives in terms of future walking robot and vehicle design, human prosthetics design, etc.

### 4.2.2 Redundancy resolution schemes

The researchers that deal with finding a solution for the redundant actuation, often resort to optimization techniques. For example, when treating the problem of two serial manipulators, handling a common payload in [46], Zheng and Luh proposed two methods for resolving the force redundancy. The first one is based on the well-known pseudoinverse (2-norm minimization), the second - on non-linear programming algorithm, considering the maximum torque limits of the actuators (infinite-norm minimization)[3]. In [69], Nakamura and Ghoudossi used active/passive coordinate partitioning in order to reduce the manipulator plant dynamics and eliminate the Lagrange multipliers, corresponding to the generalized constraint forces. Using this plant dynamic model, they proposed a method for

[3]The same resolution scheme comparison is used in the present research work

finding the joint torques of closed-loop manipulators based on torques of equivalent open-loop structures. Moreover, they parametrized the torque solution for cases of actuation redundancy. As we shall see later, their achievements were recently successfully exploited by other research teams for the purposes of redundantly actuated parallel manipulator control. Tao and Luh [70] considered the similar problem of two redundant serial manipulators that manipulate a payload in cooperation, and minimized the squares of the joint torques to find a redundancy solution. Gonzalez et Sreenivasan [34] treated the example of two 6-d.o.f. coupled serial manipulators, deriving a minimum norm solution for this redundant actuation problem. Nahon and Angeles [71] used quadratic programming (QP) with inequality constraints in order to solve for the torques of hand fingers, grasping an object, thus minimizing the internal forces. Buttolo and Hannaford [72] looked at a linear programming problem (LP) and applied a Simplex algorithm to solve the redundant actuation of a parallel-structure based haptic device. They compared the results with that of a pseudoinverse solution. Kerr et al. [47] used the internal stiffness properties of redundant parallel manipulators to obtain a solution that minimizes the potential energy of the system, demonstrating that this approach is equivalent to a weighted pseudoinverse solution. Merlet [73] proposed two cost functions for the redundancy solution via optimization: one minimizing the joint rates and one minimizing the actuator efforts. The proposed methodology was again based on the pseudoinverse solution. Kim and Choi [74] developed analytical methods for the description of manipulator kineto-static capabilities, based on eigenvalue treatment. Using [47] they demonstrated the possible extension of their solutions to redundantly actuated parallel manipulators. Lee et al. proposed a method for managing impact disturbances on parallel manipulators by means of redundant actuation [75]. They compared three redundancy solutions: a minimum torque norm solution, a minimum torque rate solution and a solution that respects the torque limits, drawing the conclusion that the third one gives best results. Cheng et al. defined in [76] a transmission ratio between the torque input and output of parallel manipulators, demonstrating that the force redundancy lead to more uniform and symmetric force output. Nokleby et al. in [40] derived an analytical methodology using scaling factors to determine the force capabilities of parallel manipulators, presenting in addition an optimization-based method for generation of these capabilities. They showed that the redundant actuation allows for significant improvement of the latter and that using optimization-based solutions for the redundancy makes better use of the maximum force/torque performances of the actuators. Dasgupta [77] drew the reader's attention to the need of effective redundancy solution schemes, discussing and comparing some simple solution examples of different norm on the basis of their performance with respect to actuator limits and other constraints. Ding et al. [78] considered a comprehensive dynamic performance index (CDPI), representing the maximum value of relative joint driving torques, as a measure of the dynamic merit of redundant manipulators. They solved the force redundancy through CDPI optimization by linear programming, incorporating the stability condition into the dynamic optimization algorithm to eliminate the instability problem for long trajectories of the end-effector. Further, they commented the computational burden that CDPI bares and show that it can be handled easily using recurrent neural networks. Zhang presented in [79] an interesting approach for inverse-free QP formulation of the infinity-norm torque minimization problem (minimum-effort redundancy solution) for redundant manipulators, testing it on an industrial PUMA-560 robot.

As we shall see, the reformulation of the QP inverse-free minimum-effort redundancy solution for the case of redundantly actuated parallel manipulators is not problematic. This reformulation was performed by the author of this work and applied to two benchmark parallel multibody systems, in order to compare the solution with a minimum two-norm one (using the *pseudoinverse* solution approach). The results showed the advantages of the infinite-norm QP problem, when optimal distribution of the actuator loads that respects their performance limits is needed. All this will be commented in detail later in the work.

# Part II

# Multibody Formalism

# Motivation

Starting from the simplest mechanical devices like slider-cranks, four-bar mechanisms, going through more complex mechanical systems, e.g. weaving machines, bicycles, road and railway vehicles, robots and spacecrafts – we can find that they are all constituted of a number of rigid or flexible components, connected in such a way that relative motions between them can exist. These mechanical systems are known as *multibody systems*. We find numerous examples of them in our everyday life (see Figure 4.10).

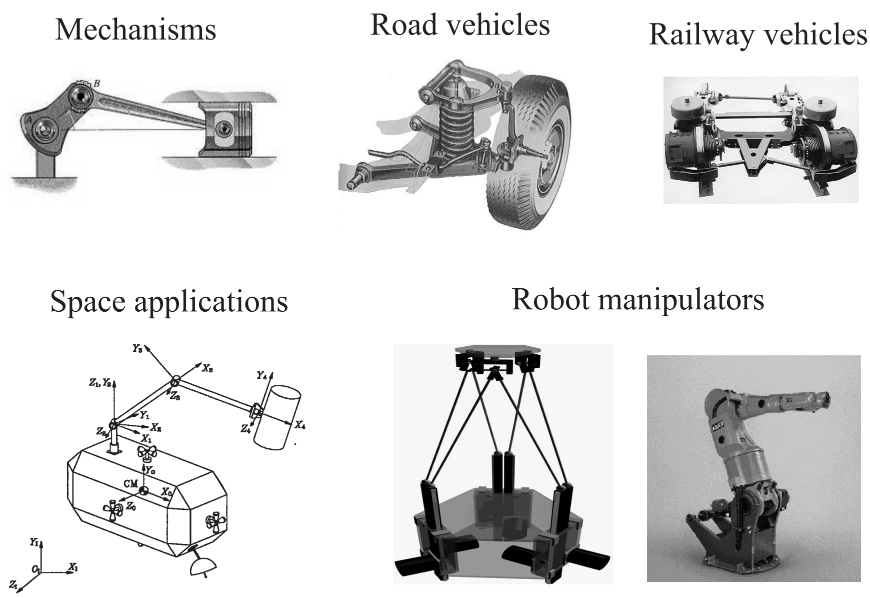



Figure 4.10: Examples of multibody systems

Nowadays, the latest developments in multibody dynamics have made possible the in-depth analysis and advanced modeling of systems of high complexity that include not only mechanical, but also electrical, hydraulic and pneumatic, and even control systems. In our work, however, we concentrate our vision on mechanical systems, constituted of rigid bodies, and basically deal with modeling, analysis and control of multibody systems in robotics – parallel manipulators, in particular.

In Chapter 1 we already mentioned our choice of following the multibody formalism principles, briefly revealing its advantages when dealing with multibody mechanical systems, namely, its convenience for a description of systems of varying complexity, using comprehensive terminology and hypotheses of acceptable simplification level, different types of generalized coordinates and conventions of readable notation. In this part, a thorough presentation of the MBS formalism definitions, conventions and hypotheses, as well as the modeling techniques that it provides for description of multibody system kinematics and dynamics, shall be given. All these theoretical aspects are of primordial importance for the developments of the present work. The principles of Newton-Euler recursive computation scheme (Chapter 7) – a precious tool, for example, to compute the robot inverse dynamics for a given trajectory, or the direct dynamics so as to validate the solution via time simulation (Chapter 9), will be described. The coordinate partitioning technique [4], which lies also at the root of our approach, will be detailed and generalized in this part to parallel robots. Finally, the MBS formalisms and the techniques mentioned above will be used to develop robust closed-loop multibody dynamic models and time integration procedures that shall be subsequently used in the control of redundantly actuated parallel manipulators.

# Chapter 5

# Multibody formalism terminology

## 5.1 Basic concepts

A system composed of a set of $N$ rigid[1] *bodies* interconnected by mechanical devices, called *joints*, can be defined as a *multibody system (MBS)* [3] (see Figure 4.10).

Multibody formalisms consider the joints as massless components of the system, connecting two constitutive bodies at reference *attachment points* (Figure 5.1), in a way that permits relative motion between them. At least one body is connected to an inertial (fixed) body, called *base*[2].

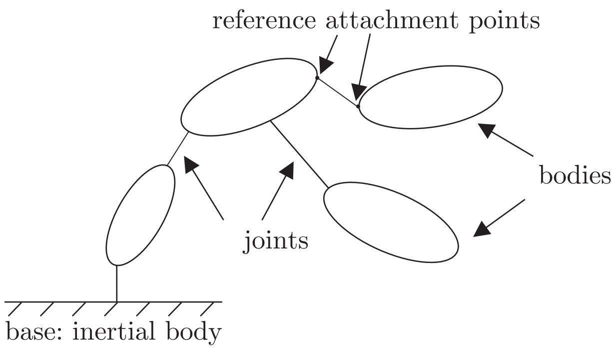



Figure 5.1: Multibody system representation

In order to develop the theoretical basis of multibody kinematics and dynamics, it is necessary to introduce fundamental concepts, such as *kinematic chain*, *body filiation* in a chain, *tree-like* and *closed-loop structure*, *kinematic constraint*, etc. We shall start with considering *tree-like structures*. i.e. structures without kinematic loops in them. Figure 5.2 gives a representation of a tree-like and a closed-loop structure in terms of comparison.

[1] or even flexible

[2] the fixed platform of a manipulator, for example.

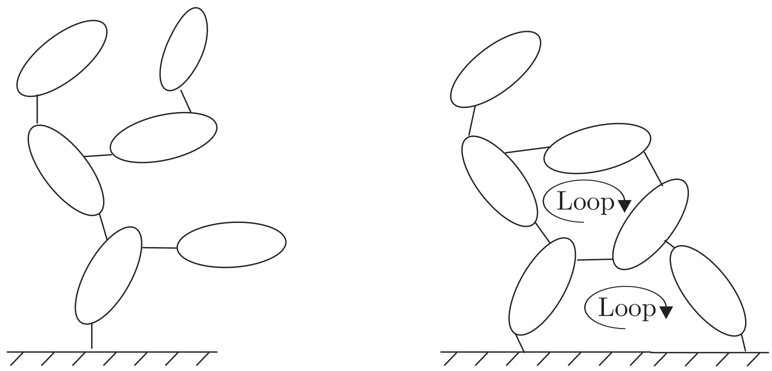



Figure 5.2: Tree-like structure and closed-loop structure

### Tree-like MBS characteristics

Tree-like multibody mechanical systems possess the following specific characteristics:

- the number of rigid bodies (without the inertial) is equal to the number of joints in the system
- the path from one rigid body of the system to any other body is unique

The following elements (taken from [3]) characterize any tree-like topology:

- *Base body*: the body of the MBS that is inertially fixed.
- *Leaf*: any terminal body of the tree-like structure (ex: $l$, $p$ and $o$ in Figure 5.3).
- *Kinematic chain*: an ordered set of *consecutive* bodies, starting from the base and going to the leaves. For instance, $\{i, j, k\}$ in Figure 5.3 is a kinematic chain, while $\{i, k, j\}$ and $\{m, j, k\}$ are not.
- *Ancestor*: body $i$ is an ancestor of body $k$ if it belongs to the kinematic chain starting from the base and going to $k$ ($k$ excluded).
- *Descendant*: if body $i$ is an ancestor of body $k$, then body $k$ is said to be a *descendant* of body $i$.
- *Parent*: body $i$ is the *parent* of body $j$ if it is the direct ancestor of $j$. In a tree-like structure each body has only one parent, contrary to *closed-loop* structures (see Fig. 5.2).
- *Child*: body $j$ is called *child* of body $i$, if body $i$ is the parent of body $j$. Analogically, the joints connecting body $i$ to its child bodies will be called *child joints* of body $i$. In a tree-like structure, one parent may have more than one child.

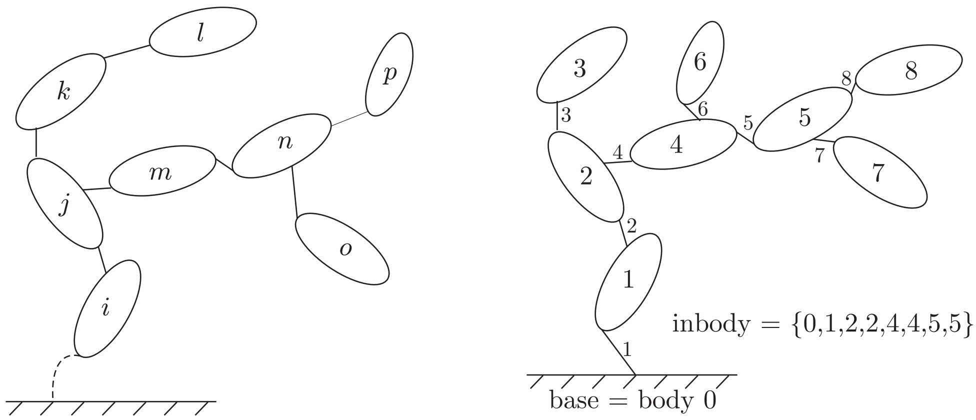



Figure 5.3: Left: Tree-like body filiation. Right: Numbering of bodies and joints in a tree-like structure.

## 5.2 Tree-like MBS topology

Certain MBS topology conventions and definitions are adopted, mostly for convenience when dealing with computer implementations, such as the one underlying the ROBOTRAN symbolic generation software [3]:

### Conventions

1. Bodies are numbered in ascending order, starting from the base (index 0) and going to the leaf bodies (as illustrated in Figure 5.3).

2. A joint, preceding a given body, receives its index.

### Definitions

- A topological vector *inbody* is defined (giving the information on the structure filiation) as the vector whose $i^{th}$ element contains the index of the parent body $i$.

Any tree-like structure can be described by means of the topology conventions and elements given. For example, a *linear tree-like structure* can be defined, as being a multibody structure, in which the number of bodies equals the number of joints and every parent body has only one child (Figure 5.4). This type of structure is very common when the legs (branches) of a parallel manipulator are considered, for instance to describe the kinematics of their attachment points with the moving platform. The closed-loop structures, on their hand, are structures in which the number of bodies is greater that the number of joints.

## 5.3 Kinematics: basic notations and definitions

### Notations

We shall first introduce the notations of vectors and tensors that will be used in this work.

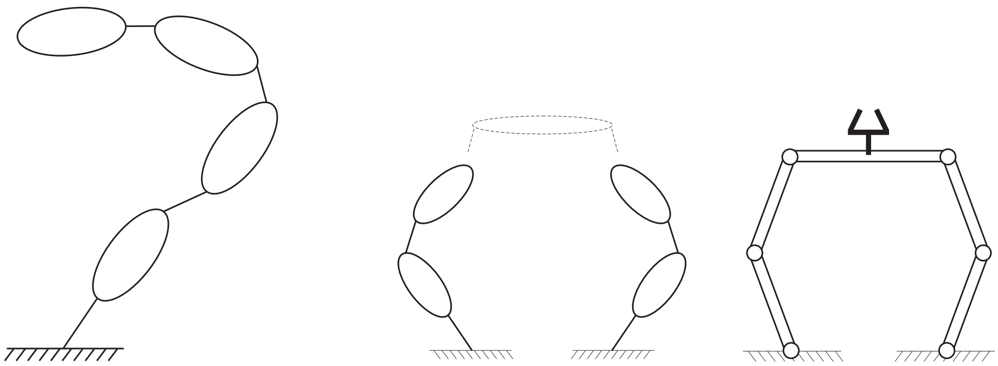


Figure 5.4: Linear tree-like multibody structure. Left: principal topology. Right: an example – the legs of a planar parallel manipulator considered as linear tree-like (serial) chains.

Let $\{\hat{\mathbf{X}}^i\}$ be an orthonormal frame. Writing the frame $\{\hat{\mathbf{X}}^i\}$ under the form of a $(3 \times 1)$ column matrix of *unit vectors* $\hat{\mathbf{X}}^i$ gives:

$$[\hat{\mathbf{X}}^i] \triangleq \begin{pmatrix} \hat{\mathbf{X}}^i_x \\ \hat{\mathbf{X}}^i_y \\ \hat{\mathbf{X}}^i_z \end{pmatrix} \tag{5.1}$$

A vector $\mathbf{a}$ is described in the given frame $\{\hat{\mathbf{X}}^i\}$ by its three components $a_x, a_y, a_z$:

$$\mathbf{a} = a_x\,\hat{\mathbf{X}}^i_x + a_y\,\hat{\mathbf{X}}^i_y + a_z\,\hat{\mathbf{X}}^i_z \tag{5.2}$$

Applying classical matrix multiplication rules, we can write the vector $\mathbf{a}$ in the following concise form:

$$\mathbf{a} = [\hat{\mathbf{X}}^i]^T\; a \tag{5.3}$$

where $a \triangleq (a_x \;\; a_y \;\; a_z)^t$.

The same notation holds for a tensor $\mathbf{T}$ of order 2 (ex: the inertia tensor of a body):

$$\mathbf{T} = [\hat{\mathbf{X}}^i]^T\, T\, [\hat{\mathbf{X}}^i] \text{ with } T = \begin{pmatrix} T_{xx} & T_{xy} & T_{xz} \\ T_{yx} & T_{yy} & T_{yz} \\ T_{zx} & T_{zy} & T_{zz} \end{pmatrix}$$

### Definitions

Now the following quantities, related to body $i$, its parent $h$ and children $j$ and $k$, can be defined on the basis of Figure 5.3:

- $O^i$ and $O'^i$ – the reference attachment points of joint $i$ on the parent body $h$ and its child $i$, respectively,
- $\mathbf{z}^i$ – the relative position vector $\overrightarrow{O^i\,O'^i}$, representing the relative displacement in joint $i$,

- $\mathbf{d}^{ik}$ – the position vector (with respect to $O'^i$) of the reference attachment point $O^k$ of joint $k$ that connects body $i$ with its child body $k$. This vector is fixed on body $i$ and represents its contribution to the kinematic chain, linking body $k$ to the inertial frame $\{\hat{\mathbf{I}}\}$ of body 0 (the base).

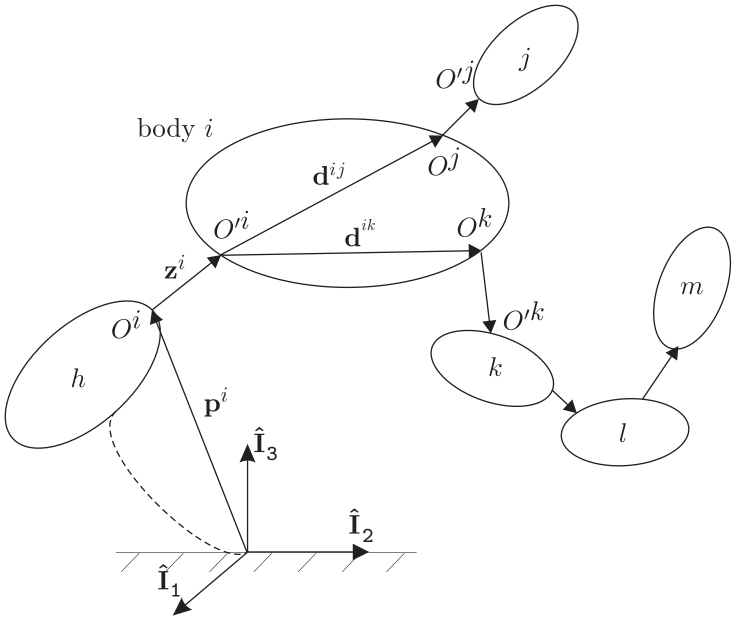



Figure 5.5: Basic kinematic notations: attachment points and vectors

The following augmented vectors are defined as well for convenience in case of recursive formalism computations (see Section 7.3):

- $\mathbf{d}_z^{ij} \triangleq \mathbf{z}^i + \mathbf{d}^{ij}$, $\mathbf{d}_z^{ik} \triangleq \mathbf{z}^i + \mathbf{d}^{ik}$ – the *extended position vectors* of joints $j$ and $k$, respectively (Figure 5.6).

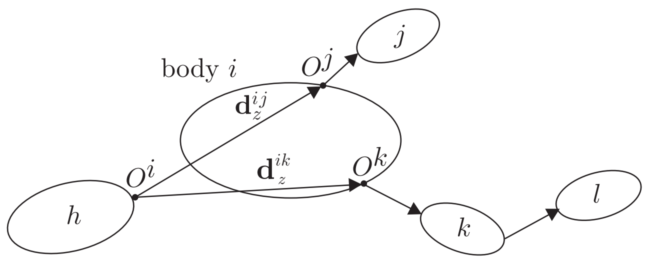



Figure 5.6: Basic kinematic notations: frames, bodies and joints

The orientation of the different bodies in given kinematic chain and their angular velocities can be described using the following quantities of Figure 5.7:

- $\{O, \{\hat{\mathbf{I}}\}\}$ – the orthonormal inertial frame, rigidly fixed to the base (body 0),
- $\{\hat{\mathbf{X}}^i\}$ – an orthonormal moving frame, rigidly attached to body $i$. Expressed in this frame, the components of the vectors $\mathbf{d}^{ij} = [\hat{\mathbf{X}}^i]^T \, d^{ij}$ and $\mathbf{d}^{ik} = [\hat{\mathbf{X}}^i]^T \, d^{ik}$ have constant components,

- $R^{i,h}$ – the $(3 \times 3)$ rotation matrix, defining the relative orientation of the frame $\{\hat{\mathbf{X}}^i\}$ (body $i$) with respect to frame $\{\hat{\mathbf{X}}^h\}$ (parent body $h$), i.e. $[\hat{\mathbf{X}}^i] = R^{i,h}\ [\hat{\mathbf{X}}^h]$,
- $R^i$ – the rotation matrix, defining the absolute orientation of the frame $\{\hat{\mathbf{X}}^i\}$ (body $i$) with respect to the inertial frame $\{\hat{\mathbf{I}}\}$ (fixed body 0), i.e. $[\hat{\mathbf{X}}^i] = R^i\ [\hat{\mathbf{I}}]$,
- $\boldsymbol{\Omega}^i$ – the relative angular velocity vector of body $i$ with respect to its parent $h$,
- $\boldsymbol{\omega}^i = [\hat{\mathbf{X}}^i]^T\ \omega^i = \sum_{h \leq i} \boldsymbol{\Omega}^h$ – the absolute angular velocity vector of the fixed to body $i$ frame $\{\hat{\mathbf{X}}^i\}$

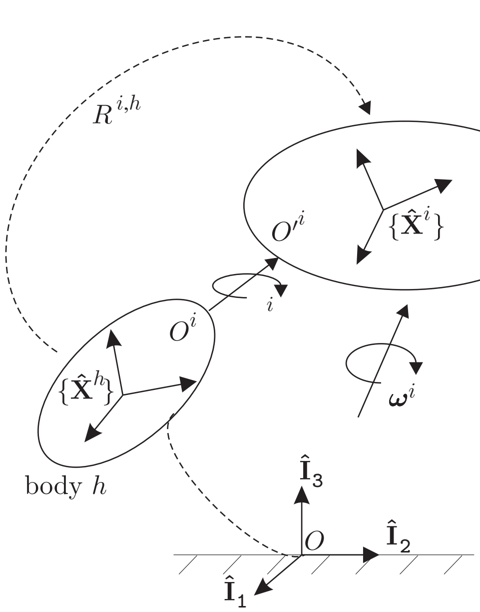



Figure 5.7: Basic kinematic notations: frames, bodies and joints

The components $\omega^i$ are related to the rotation matrix according to the following expression [3], which involves the *skew-symmetric* matrix associated with $\omega^i$:

$$\tilde{\omega}^i = \begin{pmatrix} 0 & -\omega_3^i & \omega_2^i \\ \omega_3^i & 0 & -\omega_1^i \\ -\omega_2^i & \omega_1^i & 0 \end{pmatrix} \triangleq R^i \dot{R}^{iT} \tag{5.4}$$

related to the rule of performing a vector cross product: $\mathbf{v} \times \mathbf{w} = [\hat{\mathbf{X}}^i]^T \tilde{v} w$ for any two vectors $\mathbf{v}$ and $\mathbf{w}$, the components of which are expressed in a given frame $\{\hat{\mathbf{X}}^i\}$.

As only rigid bodies are considered in the present work, knowing the position of one reference point $Q^i$ per body $i$ and its orientation suffice to completely determine the configuration of the multibody system.

### Joint modeling hypotheses

Joints represent simple mechanical connection devices such as telescopic arms, hinges, universal joints, bearings, etc. The relative motions between bodies are in fact allowed by the joints and correspond to the mechanical nature of the latter, i.e. to their *internal relative degrees of freedom.*

In fact, every existing joint can be modeled as an appropriate succession of two most basic, one-d.o.f. joints: *prismatic* and *revolute* (Figure 5.8). Therefore, instead using an exhaustive library of real joints, we shall use this fact (as in [3]) to simplify the modeling procedure by formulating the following modeling hypothesis:

*In a mutlibody system (MBS) the connections between its constitutive bodies are realized by 1-d.o.f. prismatic or revolute joints.*

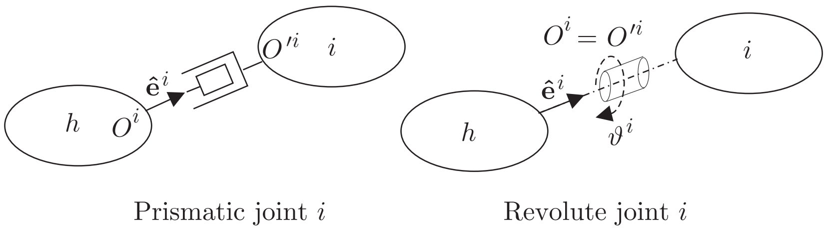



Figure 5.8: Elementary joints

Each joint $i$ connects body $i$ to its parent $h$ at anchor reference points $O'^i$ and $O^i$, respectively. The joint attachment points are distinct for a prismatic joint and identical for a revolute one (as depicted in Figure 5.8). The variables that represent the relative motion in the joints (translational or angular displacement) are used as *joint generalized coordinates q*.

## Joint kinematics

Let us introduce the joint unit vector $\hat{\mathbf{e}}^i$, aligned with the joint axis according to Figure 5.8 and having constant components in both $\{\hat{\mathbf{X}}^i\}$ and $\{\hat{\mathbf{X}}^h\}$. For a joint $i$ of an MBS, the generalized coordinate $q^i$ represents:

- if joint $i$ is prismatic, the amplitude of the relative displacement $\overrightarrow{O^i\,O'^i}$ along the unit vector $\hat{\mathbf{e}}^i$. The relative position of the attachment point $O'^i$ with respect to the the attachment point $O^i$ is given by a vector $\mathbf{z}^i$, such that:

$$\mathbf{z}^i = \overrightarrow{O^i\,O'^i} = q^i\hat{\mathbf{e}}^i \text{ (and } R^{i,h} = E) \tag{5.5}$$

  which means that $O'^i$ coincides with $O^i$ when $q^i = 0$.

- if joint $i$ is revolute, the amplitude of the relative rotation angle $\vartheta^i$ of body $i$ with respect to its parent body, around the unit vector $\hat{\mathbf{e}}^i$. The relative orientation of the frame $\{\hat{\mathbf{X}}^i\}$, fixed on body $i$, with respect to the frame $\{\hat{\mathbf{X}}^h\}$, fixed on body $h$, is given by a rotation matrix $R^{i,h}$, depending on $q^i \equiv \vartheta^i$:

$$[\hat{\mathbf{X}}^i] = R^{i,h}(q^i)[\hat{\mathbf{X}}^h] \text{ (and } \mathbf{z}^i = 0) \tag{5.6}$$

  where the joint relative angle $\vartheta^i$ is the only time-dependent quantity in $R^{i,h}(q^i)$.

Now, the relative translational and angular velocities of body $i$ with respect to its parent $h$ can be defined:

$$\begin{array}{lll} \overset{\circ}{\mathbf{z}}{}^i = \dot{q}^i \hat{\mathbf{e}}^i & , \quad \mathbf{\Omega}^i = 0 & \text{if joint } i \text{ is prismatic} \\ \mathbf{\Omega}^i = \dot{q}^i \hat{\mathbf{e}}^i & , \quad \overset{\circ}{\mathbf{z}}{}^i = 0 & \text{if joint } i \text{ is revolute} \end{array}$$

For computer recursive implementation, it is convenient to define two vectors $\boldsymbol{\varphi}^i$ and $\boldsymbol{\psi}^i$ for each joint $i$ ($i = 1, ..., N^{body}$) such as:

$$\begin{array}{llll} \boldsymbol{\varphi}^i \triangleq 0 & \text{and} & \boldsymbol{\psi}^i \triangleq \hat{\mathbf{e}}^i & \text{if joint } i \text{ is prismatic} \\ \boldsymbol{\varphi}^i \triangleq \hat{\mathbf{e}}^i & \text{and} & \boldsymbol{\psi}^i \triangleq 0 & \text{if joint } i \text{ is revolute} \end{array} \tag{5.7}$$

which allows us to rewrite the velocities in a more compact form:

$$\overset{\circ}{\mathbf{z}}{}^i = \dot{q}^i \boldsymbol{\psi}^i \text{ and } \mathbf{\Omega}^i = \dot{q}^i \boldsymbol{\varphi}^i \tag{5.8}$$

and the corresponding relative accelerations are:

$$\overset{\circ\circ}{\mathbf{z}}{}^i = \ddot{q}^i \boldsymbol{\psi}^i \text{ and } \overset{\circ}{\mathbf{\Omega}}{}^i = \ddot{q}^i \boldsymbol{\varphi}^i \tag{5.9}$$

## 5.4 Dynamics: basic notations and definitions

According to Figure 5.9 here follow the main notations and definitions that will be used in this study to model the dynamics of multibody systems:

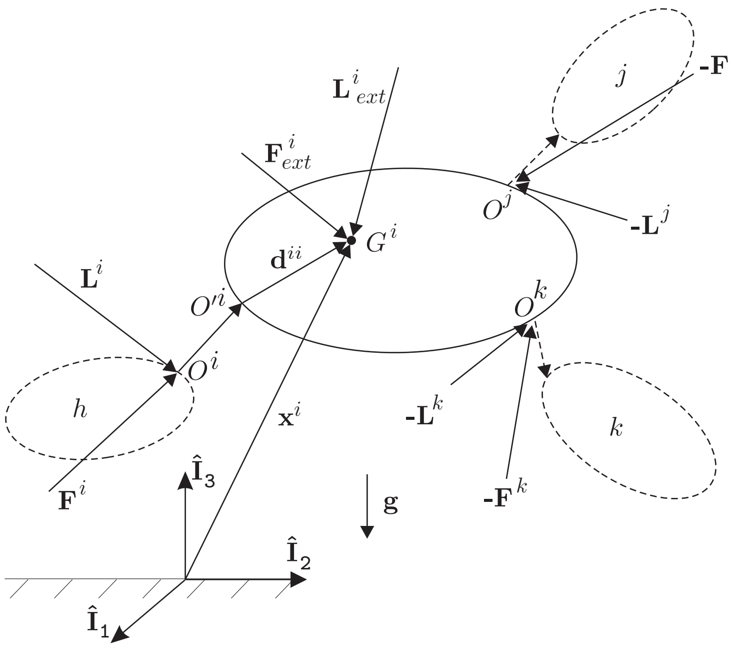



Figure 5.9: Dynamic notations

- $m^i$ and $\mathbf{I}^i = [\hat{\mathbf{X}}^i]^T I^i [\hat{\mathbf{X}}^i]$ – the mass of body $i$ and its inertia tensor with respect to its center of mass $G^i$.

- $\mathbf{d}^{ii} = [\hat{\mathbf{X}}^i]^T d^{ii}$ – the position vector of the center of mass $G^i$ with respect to $O'^i$. The components $d^{ii}$ and $I^i$ are constant in the body $i$ fixed frame $\{\hat{\mathbf{X}}^i\}$,

- $\mathbf{x}^i = [\hat{\mathbf{I}}]^T x^i$ – the absolute position vector of the center of mass $G^i$,

- $\mathbf{g} = [\hat{\mathbf{I}}]^T g$ – the gravity vector,

- $\mathbf{F}^i = [\hat{\mathbf{X}}^i]^T F^i$, $\mathbf{L}^i = [\hat{\mathbf{X}}^i]^T L^i$ – the resultant force and torque, respectively, applied to body $i$ by its parent $h$ through joint $i$ and evaluated at point $O^i$. According to the *Newton's third law*, reactions $-\mathbf{F}^i$ and $-\mathbf{L}^i$ are applied on body $h$. Similarly (Figure 5.9), body $i$ receives reaction forces and torques $-\mathbf{F}^j$, $-\mathbf{F}^k$, $-\mathbf{L}^j$ and $-\mathbf{L}^k$ through its child body joints,

- $\mathbf{F}^i_{ext} = [\hat{\mathbf{X}}^i]^T F^i_{ext}$, $\mathbf{L}^i_{ext} = [\hat{\mathbf{X}}^i]^T L^i_{ext}$ – the external loads acting on body $i$ (excluding the gravity as well as the internal forces and torques transmitted by the joints) in the form of an equivalent resultant force applied at the body center of mass $G^i$ and resultant torque.

For the sake of convenience the following *extended mass center position vector* is defined as well:

- $\mathbf{d}^{ii}_z = \mathbf{z}^i + \mathbf{d}^{ii}$, representing the position vector of the mass center $G^i$ with respect to $O^i$.

The total resulting force and torque applied on body $i$ can now be calculated as:

$$\mathbf{F}^i_{tot} = \mathbf{F}^i - \sum_{j\in\bar{i}} \mathbf{F}^j + \mathbf{F}^i_{ext} + m^i\mathbf{g} \tag{5.10}$$

$$\mathbf{L}^i_{tot} = \mathbf{L}^i - \sum_{j\in\bar{i}} (\mathbf{L}^j + (\tilde{\mathbf{d}}^{ij} + \tilde{\mathbf{d}}^{ii})\mathbf{F}^j) + \mathbf{L}^i_{ext} + \tilde{\mathbf{d}}^{ii}_z\mathbf{F}^i \tag{5.11}$$

where $\sum_{j\in\bar{i}}$ represents the summation over all the child bodies of body $i$.

# Chapter 6

# Multibody Kinematics

In this chapter, tree-like direct and closed-loop inverse MBS kinematics will be reviewed. In order to solve the inverse kinematic problem, the *coordinate partitioning* method, mentioned in Section 3.3, will be detailed and generalized for systems that reconfigure during motion.

## 6.1 Forward sub-chain kinematics

It can be of interest for various reasons to obtain the symbolic expression of the forward kinematics of any MBS *sub-chain*, i.e. the position, the orientation, the linear/angular velocities and the linear/angular accelerations of a given body with respect to another one (the inertial base, in particular). For instance, sub-chain forward kinematics can be useful for:

- the expression of a new constraint on the system,
- the introduction of a specific force, whose constitutive equation requires the computation of the kinematics of a given point,
- the computation of a specific result, such as the absolute position or acceleration of a point (e.g. the absolute deviation of a robot tool with respect to a prescribed tool trajectory).

In our formalism, sub-chain kinematics can be generated for tree-like structures only. However, as we shall see in the next section, closed-loop structures are cut in different manners in order to restore a tree-like topology, hence the direct kinematic computation formalism can be used for any topology, be it tree-like or not.

Let us have a sub-chain (Figure 6.1) from body $i$ to body $j$ of a given tree-like structure, i.e. a parallel robot leg structure from the base to the moving platform. For any point $P$ of body $j$, the following direct kinematic elements can be generated[1]:

[1] All of them are supposed to be *scleronomic*, i.e. not depending explicitly on the time $t$.

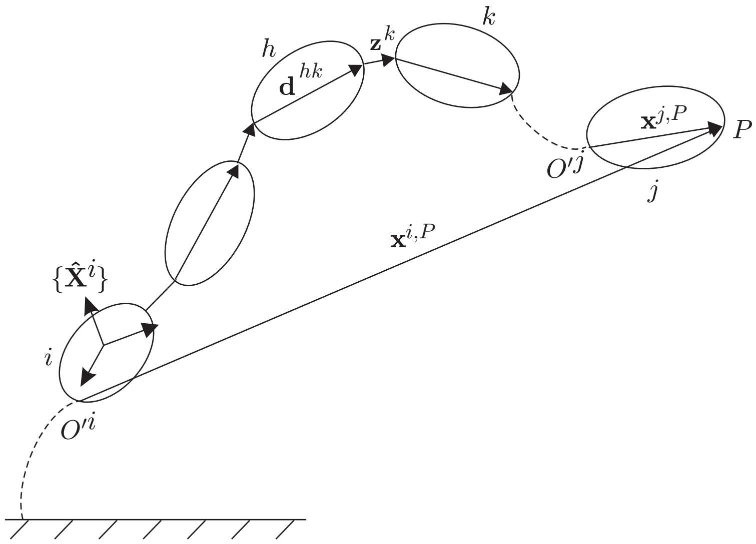



Figure 6.1: Sub-chain kinematics

- the (relative) position vector $\overrightarrow{O'^i\,P}$ of point $P$ with respect to point $O'^i$, expressed in the frame $\{\hat{\mathbf{X}}^i\}$: $\mathbf{x}^{i,P} = [\hat{\mathbf{X}}^i]^T x^{i,P}(q)$,
- the (relative) orientation (rotation) matrix $R^{j,i}$ of body frame $\{\hat{\mathbf{X}}^j\}$ with respect to body frame $\{\hat{\mathbf{X}}^i\}$, defined by: $[\hat{\mathbf{X}}^j] = R^{j,i}(q)[\hat{\mathbf{X}}^i]$,
- the *relative* velocity vector $\overset{\circ}{\mathbf{x}}^{i,P}$, defined by: $\overset{\circ}{\mathbf{x}}^{i,P} = [\hat{\mathbf{X}}^i]^T \dot{x}^{i,P}(\dot{q}, q)$
- the *relative* angular velocity vector $\boldsymbol{\omega}^{j,i}$ associated with the rotation matrix $R^{j,i}$: $\boldsymbol{\omega}^{j,i} = [\hat{\mathbf{X}}^i]^T \omega^{j,i}(\dot{q}, q)$,
- the corresponding acceleration vectors: $\overset{\circ\circ}{\mathbf{x}}^{i,P} = [\hat{\mathbf{X}}^i]^T \ddot{x}^{i,P}(\ddot{q}, \dot{q}, q)$, $\overset{\circ}{\boldsymbol{\omega}}^{j,i} = [\hat{\mathbf{X}}^i]^T \dot{\omega}^{j,i}(\ddot{q}, \dot{q}, q)$.

A recursive computation formalism is used to obtain the expressions given above. As in [3], it holds two particularities:

- the pair $\{O'^i, \{\hat{\mathbf{X}}^i\}\}$ plays the role of local inertial frame (instead of $\{O, \{\hat{\mathbf{I}}\}\}$) for computing the time derivatives,
- in order to save computational effort and time in the evaluation of the Jacobian, resorting from the generation of relative velocities, the translational part of the kinematic equations (Step 4 of the algorithm hereunder) is performed backwards.

Considering the kinematic chain of Figure 6.1 that links the point $P$ to body $i$ and having all the notations in correspondence with the definitions given in Sections 5.2 and 5.3, the sequential steps of the recursive algorithm are as follows[2]:

1. Determine the set of joint indices, belonging to the kinematic chain, by backward recursion of the vectorial *inbody*. Denote this set by: $\{i+1 : j\}$.

[2]For conciseness, the algorithm is presented in its vector form.

2. Compute the rotation matrix $R^{i,k}$ from the elementary joint rotation matrices $R^{h,k}$:

   *Initialization*:

   $R^{i,i} = E$ (unitary matrix)

   *Forward recursion*:

   For $k = i + 1 : j$

   $$h = inbody(k)$$
   $$R^{i,k} = R^{i,h} R^{h,k}$$

   end

3. Compute by a forward recursion the angular velocity and acceleration vectors:

   *Initialization*:

   $\boldsymbol{\omega}^{i,i} = \mathring{\boldsymbol{\omega}}^{i,i} = 0$

   *Forward recursion*:

   For $k = i + 1 : j$

   $$h = inbody(k)$$
   $$\boldsymbol{\omega}^{k,i} = \boldsymbol{\omega}^{h,i} + \boldsymbol{\varphi}^k \dot{q}^k$$
   $$\mathring{\boldsymbol{\omega}}^{k,i} = \mathring{\boldsymbol{\omega}}^{h,i} + \tilde{\boldsymbol{\omega}}^{k,i} \boldsymbol{\varphi}^k \dot{q}^k + \boldsymbol{\varphi}^k \ddot{q}^k$$

   end

4. Compute by a backward recursion the position, velocity and acceleration vectors:

   *Initialization*:

   $k = j$ , $\mathbf{x}^{k,P} = \mathbf{x}^{j,P}$

   $\mathring{\mathbf{x}}^{k,P} = \tilde{\boldsymbol{\omega}}^{j,i} \mathbf{x}^{j,P}$

   $\overset{\circ\circ}{\mathbf{x}}^{k,P} = \tilde{\mathring{\boldsymbol{\omega}}}^{j,i} \mathbf{x}^{j,P} + \tilde{\boldsymbol{\omega}}^{j,i} \tilde{\boldsymbol{\omega}}^{j,i} \mathbf{x}^{j,P}$

   *Backward recursion*[3]:

   while $k \neq i$

   $$h = inbody(k)$$
   $$\mathbf{x}^{h,P} = \mathbf{x}^{k,P} + \mathbf{d}^{hk} + \mathbf{z}^k$$
   $$\mathring{\mathbf{x}}^{h,P} = \mathring{\mathbf{x}}^{k,P} + \tilde{\boldsymbol{\omega}}^{h,i} (\mathbf{d}^{hk} + \mathbf{z}^k) + \boldsymbol{\psi}^k \dot{q}^k$$
   $$\overset{\circ\circ}{\mathbf{x}}^{h,P} = \overset{\circ\circ}{\mathbf{x}}^{k,P} + \tilde{\mathring{\boldsymbol{\omega}}}^{h,i} (\mathbf{d}^{hk} + \mathbf{z}^k) + \tilde{\boldsymbol{\omega}}^{h,i} \tilde{\boldsymbol{\omega}}^{h,i} (\mathbf{d}^{hk} + \mathbf{z}^k) + 2 \tilde{\boldsymbol{\omega}}^{h,i} \boldsymbol{\psi}^k \dot{q}^k + \boldsymbol{\psi}^k \ddot{q}^k$$
   $$k = h$$

   end

5. Compute the Jacobian matrices

[3] Computing the derivatives, we rely on the fact that if joint $k$ is prismatic: $\boldsymbol{\omega}^{h,i} = \boldsymbol{\omega}^{k,i}$, $\mathbf{z}^k \neq 0$ and if it is revolute: $\boldsymbol{\omega}^{h,i} \neq \boldsymbol{\omega}^{k,i}$, $\mathbf{z}^k = 0$.

It will be shown in the next section that the coefficients of the generalized joint velocities $\dot{q}$ in the above expressions of velocity vectors play the role of *pseudo-gradients* for the rotation constraints, associated with a type of loop closure. A vector form, called *vector Jacobian* [80] containing the contributions of the sub-chain kinematics to the (pseudo) Jacobian is obtained as follows:

$$\begin{pmatrix} \overset{\circ}{\mathbf{x}}{}^{i,P} \\ \boldsymbol{\omega}^{j,i} \end{pmatrix} = \begin{pmatrix} \mathbf{J}_t^{i,P} \\ \mathbf{J}_r^{i,P} \end{pmatrix} \dot{q} = \mathbf{J}^{i,P} \dot{q} \tag{6.1}$$

After few manipulations the $k^{th}$ column $\mathbf{J}^{i,P}_{(k)}$ of this vector matrix, i.e. the coefficients of $\dot{q}^k$, is found to be:

$$\mathbf{J}^{i,P}_{(k)} = \begin{pmatrix} \boldsymbol{\psi}^k - \tilde{\mathbf{x}}^{k,P} \boldsymbol{\varphi}^k \\ \boldsymbol{\varphi}^k \end{pmatrix} \tag{6.2}$$

if $k$ belongs to the kinematic chain $\{i+1 : j\}$, and equal to a zero column-vector otherwise.

The latter expression shows that $\mathbf{J}^{i,P}$ can be computed within the backward recursive algorithm of point 4.

## 6.2 Closed-loop multibody systems

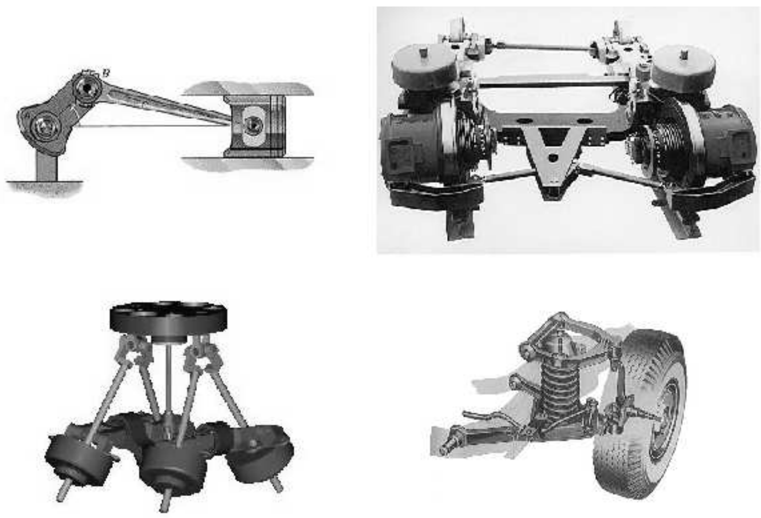


Figure 6.2: Example of closed-loop multibody systems

The previous section introduced the necessary theoretical tools to deal with tree-like multibody systems. In numerous real case, however, the systems that have to be treated do not possess a tree-like structure, but a closed-loop one (see ex. in Figure 6.2: vehicle suspensions, railway boogies, parallel robots, etc.), for which a particular procedure involving loop cuts must be performed, as depicted in Figure 6.3.

The latter shows a MBS topology that is not a tree-like one. This topology contains several *loops*, formed of bodies and joints. As these loops correspond to physical, really

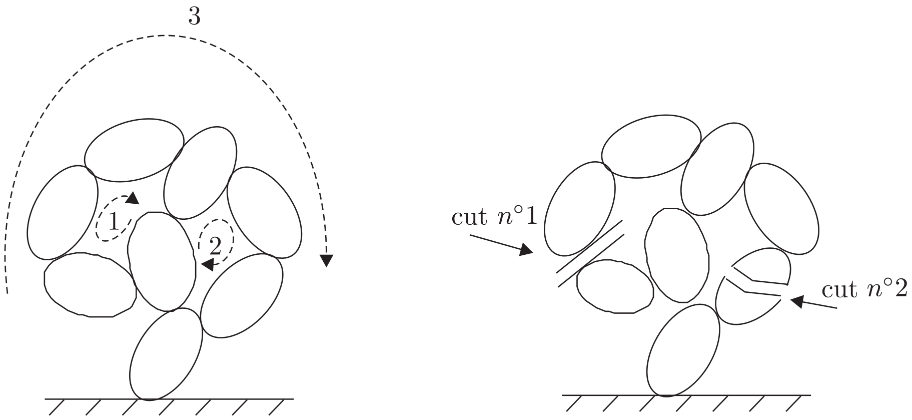



Figure 6.3: a) Closed-loop system b) Corresponding open-loop (tree-like) system

existing connections between bodies, during any motion of the mechanical system they must remain *closed*, and this has to be achieved accurately in both the kinematic and dynamic model, independently from the numerical process that will be used.

In contrast to tree-like structures, the kinematic chain connecting a body of a given loop to the base is not unique, hence the *filiation* concept [3] used for tree-like topologies is not directly applicable. On the other hand, this concept is useful when using relative joint coordinates to compute the direct kinematics of any body.

To deal with closed-loop MBS, as in [3] we shall define an *equivalent spanning tree*, covering all the bodies of the closed-loop system. Such a tree can be obtained by *virtually cutting* the closed loops of the real MBS in two ways: either by cutting bodies, or by disregarding some joints. This procedure, illustrated in Figure 6.3b, leads to restoring a tree-like structure, derived from the initial closed-loop one. Once the spanning tree is obtained, the bodies and joints can be numbered according to the tree-like filiation concepts.

Two principal issues concerning the spanning tree definition arise:

- *Number of necessary cuts.* For closed-loop systems, the number of joints $N^{joint}$ is greater than the number of bodies $N^{body}$. The difference $N^{cut} \triangleq N^{joint} - N^{body}$ is called *cyclomatic number* of the topology. Knowing that for a tree-like topology $N^{joint}$ is equal to $N^{body}$ and the spanning tree must cover all bodies, it is obvious that the number of cuts necessary for obtaining the tree equals $N^{cut}$ and corresponds to the number of independent loops of the MBS. For the case of Figure 6.3, $N^{cut} = 10 - 8 = 2$.

- *Number of loop conditions.* The closure of loop No. 1 in Figure 6.3a does not provide the closure of loop No. 2 and vice versa, hence the two loops are *independent*. Contrarily, loop No.3 in the same figure is not independent of the other two and will be automatically satisfied if loops 1 and 2 are closed. Therefore, its closure is not accounted for and the full set of loop closure conditions to be considered for the system of Figure 6.3 equals $N^{cut}$.

The loop closure conditions that have to be imposed in order to have a spanning tree, equivalent to the initial closed-loop structure, lead to ; independent scleronomic and holonomic *algebraic constraints* on the generalized coordinates $q$ of the form:

$$h(q) = 0 \tag{6.3}$$

with $h \in \Re^m$, that are generally nonlinear and cannot be solved analytically for a general case. They must be satisfied at any time and at velocity and acceleration level as well, that is:

$$\begin{aligned} h(q) &= 0 \\ \dot{h}(\dot{q}, q) &= J_c(q)\dot{q} = 0 \\ \ddot{h}(\ddot{q}, \dot{q}, q) &= J_c(q)\ddot{q} + \dot{J}_c(\dot{q}, q)\dot{q} = 0 \end{aligned} \tag{6.4}$$

We shall see in Chapter 7 how the system 6.4 can be solved using the coordinate partitioning method and how the dynamic model can be produced in reduced form by means of the Lagrange multiplier technique.

For each independent loop, the kinematic equations characterizing the relative position and/or orientation of the two sides of the cut will serve to provide the corresponding constraint equations. As in [3] we will concentrate on three principal cutting procedures, because of their simplicity and generality when covering different possible cases of kinematic loops, intrinsic to all parallel manipulators that we deal with in our work.

### 6.2.1 Closed loop cutting procedures

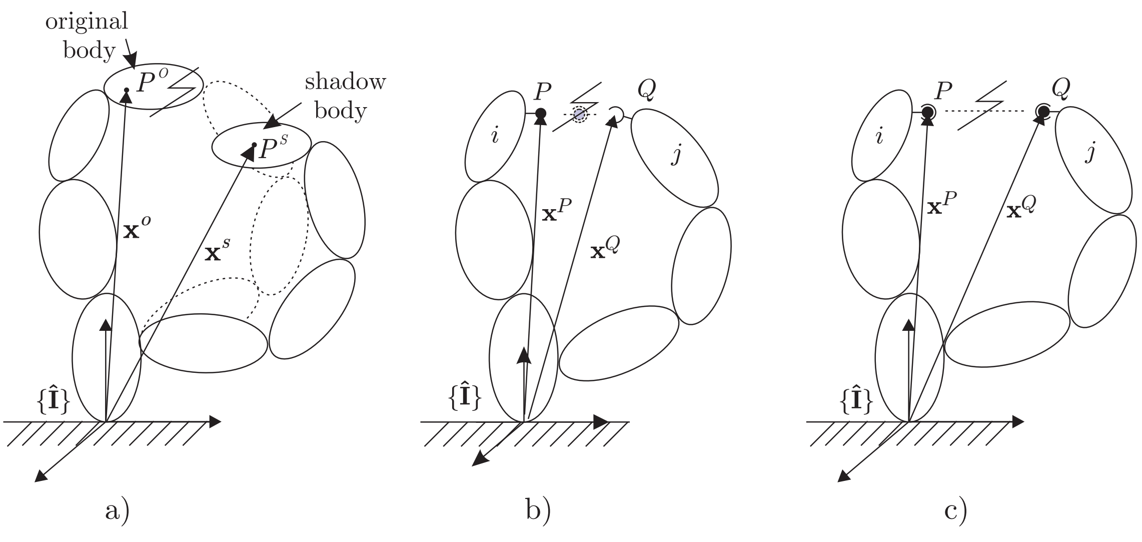



Figure 6.4: Loop cut procedures. a) Cut of a body; b) Cut of a ball joint; c) Cutting by disregarding a connecting rod.

#### Cut of a body

This is the most general cutting procedure, suitable for any kind of kinematic loop. It consists in cutting a body that belongs to the loop into two parts (see Figure 6.4a). The two parts are denoted as *original* body and its *shadow* body, respectively, the latter being massless and added to the original structure.

Closing a loop cut in such a way implies the satisfaction of six constraint equations:

- Three translational constraints, which state that the positions of the two bodies must coincide in some reference point $P^O \equiv P^S$. Defining the position vectors of $P^O$ and $P^S$ by $\mathbf{x}^O = [\hat{\mathbf{I}}]^T x^O$ and $\mathbf{x}^S = [\hat{\mathbf{I}}]^T x^S$, respectively, these constraints read:

$$h_t(q) \stackrel{\triangle}{=} x^S(q) - x^O(q) = 0 \tag{6.5}$$

- Three rotational constraints, resorting from the fact that the two body-fixed frames $\{\hat{\mathbf{X}}^O\}$ and $\{\hat{\mathbf{X}}^S\}$ must remain aligned and thus:

$$h_r \stackrel{\triangle}{=} R(q) - E = 0 \tag{6.6}$$

  where $R$ is such that $[\hat{\mathbf{X}}^S] = R(q)[\hat{\mathbf{X}}^O]$ and $E$ is the identity matrix.

  Formally, this matrix equation corresponds to nine scalar constraint equations, but since $R(q)$ is a rotation matrix that can be expressed in terms of three independent rotation variables (for instance, the angles of Tait-Bryan, mentioned in Section 3.2), only three of these scalar equations have to be considered in the total number of independent loop closure constraints (see [3] for more details).

Performing the first and second time derivative of the translational constraints (6.5) equals computing the velocities and accelerations of the two points $P^O$ and $P^S$:

$$\begin{array}{rcl} \dot{h}(\dot{q},q) \stackrel{\triangle}{=} \dot{x}^S(\dot{q},q) - \dot{x}^O(\dot{q},q) & = & J_t(q)\dot{q} = 0 \\ \ddot{h}(\ddot{q},\dot{q},q) \stackrel{\triangle}{=} \ddot{x}^S(\ddot{q},\dot{q},q) - \ddot{x}^O(\ddot{q},\dot{q},q) & = & J_t(q)\ddot{q} + \dot{J}_t(q)\dot{q} = 0 \end{array} \tag{6.7}$$

where $J_t(q)$ is the translational constraint Jacobian.

As regards the rotational constraints, it is demonstrated [3] how to express their velocity level via the relative angular velocity between frames $\{\hat{\mathbf{X}}^O\}$ and $\{\hat{\mathbf{X}}^S\}$:

$$\boldsymbol{\omega}^S - \boldsymbol{\omega}^O = [\hat{\mathbf{X}}^S]^T J_r(q)\dot{q} = 0 \tag{6.8}$$

The angular accelerations will give equivalent second derivatives of the original rotation constraint (6.6).

### Cut in a ball joint

The second cutting procedure, consists in cutting a ball (spherical for 3-D or revolute for 2-D systems) joint and is applicable if, in the loop to be cut, a ball joint exists that can be considered *ideal*, i.e. with no backlash and unable to transmit any torque. Such cases occur often in mechanisms, e.g. vehicle suspensions, planar parallel manipulators, etc. as long as the friction in the ball joints can be reasonably neglected.

The procedure of cutting a ball joint is illustrated in Figure 6.4b: the loop is cut by disregarding the joint, then a translational loop-closure constraint is introduced in order to ensure that the two points $P$ and $Q$, located at the center of the ball joint and attached to bodies $i$ and $j$, respectively, coincide. The three closure constraint equations read:

$$h(q) \stackrel{\triangle}{=} x^Q(q) - x^P(q) = 0 \tag{6.9}$$

They and their first and second time derivatives are strictly identical to the translational loop-closure conditions (6.5) and (6.7).

Compared to the first, more general, body cutting procedure, the loop opening by cutting a ball joint has two advantages:

- The three variables, representing the relative rotations occurring in the joint, are *not* included in the set of generalized coordinates $q$, characterizing the MBS, because the ball joint does not belong to the open structure, hence the dimension of system of equations, describing the MBS motion, *decreases*.
- The loop closure conditions involve only three constraint equations, thus allowing to *avoid* unnecessary computations. What is more, the *numerical convergence* – an important factor when seeking computational effectiveness – is generally better when no rotational constraints are needed.

### Cut of a connecting rod

When a loop in a given MBS contains a connecting rod (Figure 6.4c), attached to two bodies by means of *ideal* ball joints[4] (see above), and the mass and inertia of which can be neglected without loss of precision, the procedure of disregarding the connecting rod is particularly efficient.

In such cases, the rods purpose is to provide a kinematic effect on the structure, maintaining the distance between the anchor points $P$ and $Q$, equal to its length $l$, constant. This results in a single loop closing condition, giving the following closure constraint:

$$h(q) \stackrel{\triangle}{=} \left\|\mathbf{x}^Q(q) - \mathbf{x}^P(q)\right\| = l \text{ or } \|\mathbf{x}(q)\| - l = 0 \tag{6.10}$$

where $\mathbf{x}(q) \stackrel{\triangle}{=} \mathbf{x}^Q(q) - \mathbf{x}^P(q)$. From a computational efficiency point of view it is better to use the constraint in the following form:

$$h(q) \stackrel{\triangle}{=} \frac{1}{2}\|\mathbf{x}(q)\|^2 - \frac{l^2}{2} = \frac{1}{2}\,\mathbf{x}(q).\,\mathbf{x}(q) - \frac{l^2}{2} = 0 \tag{6.11}$$

which avoids root square evaluations.

The advantages of this type of cutting procedure are:

- Reduced number of dynamic equations, since the relative rotations between the rod and the connected neighbor bodies are not considered, thus decreasing the number of generalized coordinates used.

[4] or a universal joint on one side and a ball joint on the other.

- The loop closure involves a single constraint equation.

Beside the loop constraints described above, other algebraic constraints, denoted *user* constraints $h_u(q) = 0$, could be considered for different reasons: for example (see [3]), in order to impose parallelism between two bodies ($q^i = q^j$) of the MBS or to model the motion in a screw joint, involving a translation $q^i$ and a rotation $q^j$: $q^i = \alpha\ q^j$, where $\alpha[\frac{m}{rad}]$ is the thread pitch.

Let us conclude this section with the example of loop treatment for the 6-R planar parallel manipulator of Figure 6.5, used in this work.

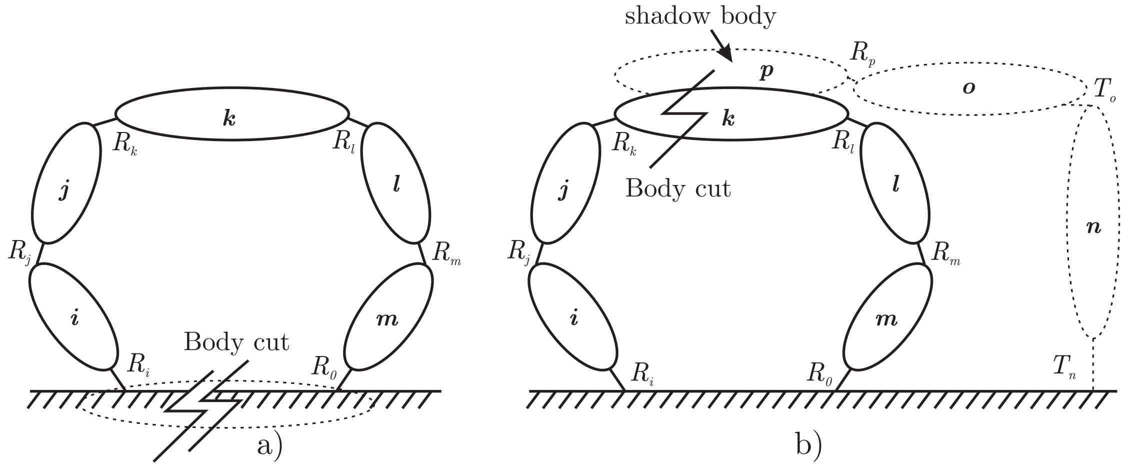



Figure 6.5: MBS topology of the 6-R planar parallel manipulator. a) – real structure loop cut (cutting the inertial body); b) – virtual loop cut (cutting the mobile platform)

In the figure $R$ stands for *revolute* joint and $T$ - for *translational* (prismatic) joint. A model particularity is the existence of *virtual* serial manipulator that induces a virtual second loop, formed by three *fictitious* massless bodies ($n$, $o$ and $p$) and three virtual joints ($T_n$, $T_o$ and $R_p$) corresponding to the manipulator task (operational) coordinates $X = [x\ y\ \theta]^T$.

The virtual serial manipulator[5] is used to solve the inverse kinematic task by imposing an operational motion $\{X, \dot{X}, \ddot{X}\}$ to the manipulator platform $k$.

In order to restore the manipulator topology to a spanning tree, two *body* cuts are performed, shown in Figure 6.5a and 6.5b, each of which induces three constraints (two translational and one rotational), since the manipulator is planar.

## 6.2.2 Coordinate partitioning method

In Section 3.3 we already referred to the coordinate partitioning method [4], when defining singularities in parallel manipulators via constrained ellipsoids. It is a useful tool for

[5]This specific approach is chosen by the author for computational efficiency and stability purposes as well (see Section 6.2.3).

situations, in which the generalized coordinates of a MBS are dependent on each other, due to some existing constraints on the system. The method provides means of finding a minimal subset of the MBS generalized coordinates that delivers a complete description of the MBS configuration.

The existence of functional dependencies between the coordinates is due to the presence of loop closure and user constraints, the equations of which can be written gathered in the generic form (6.4) at position, velocity and acceleration level.

### Independent/Dependent coordinate partitioning

Let a given multibody system be constituted of $n$ rigid bodies and subject to $m$ independent constraints $h(q) = 0$. If we apply the partitioning into *independent/dependent* generalized coordinates that we already used in Section 3.2 to the vectors $q$, $\dot{q}$, $\ddot{q}$ and the constraint Jacobian $J_c$, we can partition the system (6.4) accordingly:

$$\begin{array}{rcll} q_v & = & f_q(q_u) & \text{a)} \\ \dot{q_v} & = & B_{vu}\dot{q_u} & \text{b)} \\ \ddot{q_v} & = & B_{vu}\ddot{q_u} + d & \text{c)} \end{array} \tag{6.12}$$

with $B_{vu} = -J_{c_v}^{-1} J_{c_u}$ and $d = -J_{c_v}^{-1} \dot{J}_c \dot{q}$. For $m$ independent constraints, $q_u$ denotes the subset of $(n - m)$ *independent* coordinates, the number of which corresponds to the number of MBS degrees of freedom, and $q_v$ – the subset of $m$ *dependent* coordinates, expressed as functions of $q_u$.

In view of (6.12b) and (6.12c), the partitioning $\{u, v\}$ must be chosen in such a way that the resulting square $[m \times m]$ constraint Jacobian sub-matrix $J_{c_v}$ be well conditioned. This is a key point in our work, since large motion of parallel manipulators will be considered.

Thus, finding better numerical conditioning of $J_{c_v}$ is of crucial importance for the kinematic and dynamic solution techniques for closed-loop MBS, developed in the present study. It ensures not only good convergence of the non-linear numerical solver[6], explained hereunder, that we use for the solution of (6.12), but – as we shall see later – a *robust* numerical integration of the MBS direct dynamics and subsequent *robust* real-time control simulation as well.

In this work, LU-factorization of the *full* constraint Jacobian matrix $J_c(q)$ with column permutation on the basis of the largest pivot is used in order to obtain well-conditioned sub-Jacobian $J_{c_v}$. At every time step of the solution for (6.12a), the Jacobian conditioning is compared to a given limit, satisfactory in terms of matrix inversion and Newton-Raphson convergence, and a new LU-factorization is performed if necessary, the resulting left $[m \times m]$ square block of the factorized $J_c(q)$ is then chosen as the best "candidate" for $J_{c_v}$, whose column permutation indexes will correspond to a subset $v$ that is locally the best from a numerical point of view.

[6] *Newton-Raphson* algorithm is used in this work.

The solution of the system (6.12) can be performed robustly, as soon as the partition $\{u, v\}$ makes $J_{c_v}$ well conditioned. In particular, the Newton-Raphson iterative step:

$$q_v^{k+1} = q_v^k - J_{c_v}^{-1}(q_v^k, q_u)h(q_v^k, q_u) \tag{6.13}$$

in which the right-hand side is evaluated for $q_v = q_v^k$ and the given (known beforehand, e.g. by trajectory generation) values of $q_u$, is used to solve (6.12a). Since $J_{c_v}$ has been chosen with the largest pivots in its LU-factorization, the convergence of (6.13) is optimal. We have successfully experimented this on numerous 3-D multibody systems, like those depicted in Figure 6.2.

### Active/Passive (Actuated/Non-actuated) coordinate partitioning

It is important to point out the fact that a mismatch can often be found in the robotic literature between the coordinate partitioning into "independent/dependent" and "active/passive" coordinates. Let us clarify the difference, as it is crucial for understanding the numerical solution algorithms and redundant actuation strategy, proposed in the present work:

- The partitioning of the vector of generalized coordinates $q$ into independent $q_u$ and dependent $q_v$ ones is required for *model computation* purposes, and preferred for numerical stability reasons. The solution for the parallel manipulator kinematics (3.2), as well as the reduction of the system of differential-algebraic equations, describing the constrained dynamics of parallel manipulators, into a pure ODE system (see Section 7.4.1) require an "optimal" partitioning of the variables. The latter allows to compute properly $q_v, \dot{q}_v, \ddot{q}_v$, and eliminate the dependent accelerations $\ddot{q}_v$ and the Lagrange multipliers $\lambda$ in the equations of closed-loop MBS dynamics. Thus, among the possible combinations $\{q_{u_1}, q_{v_1}\}$, $\{q_{u_2}, q_{v_2}\}$, ..., the one that leads to the best numerical conditioning of the sub-Jacobian matrix $J_{c_v}$ will be preferred. As mentioned above, this is done by using LU-factorization with column pivoting of the constraint Jacobian $J_c(q)$. In particular, a repartitioning during robot motion will be envisaged on the basis of the $J_{c_v}$ conditioning and Newton-Raphson algorithm convergence, in order to ensure robust model for direct or inverse dynamics computation purposes (see Section 7.7.1 for details).

- The partitioning into "active" and "passive" (or $\{a, p\}$ with $a$ standing for *active* and $p$ – for *passive*) coordinates strictly corresponds to the choice of actuated and non-actuated manipulator joints. This is an important difference, because an active coordinate could without any restriction be dependent and a passive one - independent. Moreover, if we replace the $\{u, v\}$ partitioning of the generalized coordinates by an active/passive one, we obtain singularities that are *actuator-dependent*. In other words, the singularities arise because of the specific actuator locations, i.e. the partition $\{a, p\}$ makes $J_m$ or $J_{c_p}$ ($J_{c_v}$ of (3.9) with $\{a, p\}$ applied), or both of them, become singular. If, on the contrary, appropriate actuator locations are chosen so as to have the two Jacobian matrices well-conditioned, the manipulator motions will remain controllable and its structure will, for instance, be able to bear important external loads. It will be explained in the next part of this work (Chapter 8) how a special actuation strategy, developed by the author and used for elimination of local force singularity problems, can curiously lead to a *sequence* of $\{a, p\}$ partitions, without actually meaning that the re-partitioning induces physical changes in the actuator locations during motion.

### 6.2.3 Closed-loop inverse kinematics

The inverse kinematic problem for parallel manipulators was already discussed in Section 2.1, the discussion concentrated on its position level, given by the system of equations (2.2). In this section we will complete it with its development on velocity and acceleration level.

Let $s$ be the column vector that assembles the vectors $X$ and $q_u$ of absolute (end-effector) and independent joint generalized coordinates, i.e. $s = [X \ q_u]^T$. Referring to (2.1) and deriving it twice with respect to time gives:

$$\begin{array}{lcll} F(s) & = & 0 & \text{a)} \\ J_F\dot{s} & = & 0 & \text{b)} \\ J_F\ddot{s} + \dot{J}_F\dot{s} & = & 0 & \text{c)} \end{array} \tag{6.14}$$

We will call $J_F$ the *augmented* parallel manipulator Jacobian, spanning the null space of $\dot{s}$.

Recalling (3.12), we compare it with (6.14b) and obtain:

$$J_F\dot{s} = 0 \Leftrightarrow J_x\dot{X} + J_{q_u}\dot{q_u} = 0 \tag{6.15}$$

with the Jacobian partitioning $J_F = [J_x \ J_{q_u}]$ and $\dot{s} = [\dot{X} \ \dot{q_u}]^T$. Following the same reasoning, the system (6.14c) can be reformulated as:

$$J_F\ddot{s} + \dot{J}_F\dot{s} = 0 \Leftrightarrow J_x\ddot{X} + J_{q_u}\ddot{q_u} + \dot{J}_x\dot{X} + \dot{J}_{q_u}\dot{q_u} = 0 \tag{6.16}$$

Hence, using (2.2) and expressing. similarly to (6.12) the generalized joint velocities and accelerations from (6.15) and (6.16) gives the complete set of equations, describing the parallel manipulator inverse kinematics:

$$\begin{array}{lcl} q_u & = & g(X) \\ \dot{q_u} & = & -J_{q_u}^{-1}J_x\dot{X} = B_{qx}\dot{X} \\ \ddot{q_u} & = & -J_{q_u}^{-1}(J_x\ddot{X} + \dot{J}_F\dot{s}) = B_{qx}\ddot{X} + d_{qx} \end{array} \tag{6.17}$$

where $d_{qx} = -J_{q_u}^{-1}\dot{J}_F\dot{s}$ and assuming that $J_{q_u}$ is of full rank for the considered configurations.

The loop constraint solution becomes problematic, when $J_{q_u}$ is ill-conditioned, i.e. the parallel manipulator is in a vicinity of a force (parallel-type) singularity (see Section 3.2

for details). In such cases the augmented vector $s$ can be repartitioned in order to give better-conditioned Jacobian.

In fact, following the considerations of Section 6.2.2, $F(s) = 0$ can be viewed as a new set of constraint equations, describing the dependencies between the generalized coordinates $X$ and $q_u$, or we can consider the set $s = \{X \ q_u\}$ as assembling the generalized independent joint coordinates $q_u$ of the parallel manipulator and the absolute coordinates $X$ of an additional, "virtual" serial manipulator, the end-effector of which coincides with that of the parallel one. This treatment of the inverse kinematics of parallel manipulators comes to adding a new, virtual loop (depicted in Figure 6.5 b) to the manipulator topology and thus, the coordinate partitioning technique can be applied in order to solve the loop constraints for a new independent/dependent partition.

Indeed, partitioning $s$ into $s = [s_u \ s_v]^T$ in such a way that the corresponding sub-Jacobian $J_{s_v}$ be well-conditioned, we obtain:

$$\begin{aligned} s_v &= f_s(s_u) \\ \dot{s_v} &= B_{s_{vu}} \dot{s_u} \\ \ddot{s_v} &= B_{s_{vu}} \ddot{s_u} + d_s \end{aligned} \tag{6.18}$$

with $B_{s_{vu}} = -J_{s_v}^{-1} J_{s_u}$ and $d_s = -J_{s_v}^{-1} \dot{J}_s \dot{s}$, provided that $J_{s_v}$ is well-conditioned.

But, as by definition for the inverse kinematics (6.17) must hold, in cases that $X$ can not be kept as independent coordinates, the formulation (6.18) is generally not interesting, because some of the joint generalized coordinates have to be taken as independent ones, but their values are not assumed to be known beforehand, but calculated via (6.17).

# Chapter 7

# Multibody Dynamics

The dynamic modeling of parallel manipulators, as well as constructing algorithms for robust[1] solution of their dynamics, occupies an important place in the present research work. The direct and inverse dynamics solutions for reconfigurable parallel MBS using the coordinate partitioning technique is a solid basis for real-time simulation, actuation solutions with actuator redundancy and subsequent real-time control applications.

In this chapter, formalisms describing the dynamics of tree-like multibody systems are recalled and then their extensions to closed-loop systems are presented. In order to solve the latter, corresponding application of the coordinate partitioning method and treatment of the Lagrange multipliers representing the components of constraint (internal) forces are applied. Finally, algorithms for computation of the direct and inverse dynamics of closed-loop MBS are proposed and corresponding computer implementations are commented.

## 7.1 Direct dynamics of tree-like multibody systems

The *direct dynamic* problem of a multibody system concerns the computation of the generalized joint accelerations $\ddot{q}$ for a given configuration $\{q, \dot{q}\}$ in the presence of external/internal forces/torques acting on the system. The direct dynamic equations are extensively used for simulation purposes, i.e. to predict the motion of the system, starting from an initial configuration (at $t = 0$) $\{q_0, \dot{q}_0\}$, by time-integrating the accelerations $\ddot{q}(t)$.

Various approaches are used to compute the joint accelerations $\ddot{q}$, e.g. based on the virtual power principle, the Lagrange equations or the standard Newton/Euler laws formulated recursively [3]. In the following section a detailed description of the latter – used in this study – will be given.

Depending on the formalism, the equations describing the system dynamics can be

[1]for any trajectory, being it singularity-free or not

generated in two forms:

- Explicit form:

$$\ddot{q} = f(q, \dot{q}, F_{ext}, L_{ext}, g, Q) \tag{7.1}$$

- Semi-explicit form:

$$M(q)\ddot{q} + c(q, \dot{q}, F_{ext}, L_{ext}, g) = Q \tag{7.2}$$

  where:

  - $M$ is the symmetric positive definite generalized mass matrix of the system,
  - $c$ is a vector that contains the Coriolis, centrifugal and gravity terms as well as the contribution of external resultant forces $F_{ext}$ and torques $L_{ext}$ acting on the system,
  - $g$ is the gravity vector,
  - $Q$ represents the vector of generalized joint forces/torques.

$Q^i$, the $i^{th}$ element of $Q$, denotes the component of the force/torque vector $\mathbf{Q}^i = Q^i\hat{\mathbf{e}}^i$, produced in the prismatic/revolute joint $i$ by its parent body $h$ along the unit joint vector $\hat{\mathbf{e}}^i$. It can be a passive element (spring, friction, damping) or a force/torque due to an actuator.

In this work the generalized joint forces $Q_i$ will be assimilated to the actuator efforts (torques) only. All other types of generalized forces/torques (friction, damping, ...) will be transfered from the vector $Q$ to the $c$ vector (mainly because of formulation considerations), when searching for actuator torque solutions.

From (7.2), the accelerations $\ddot{q}$ can be determined by linear algebra techniques, the mass matrix being factorized on the basis of the Cholesky decomposition technique.

## 7.2 Inverse dynamics of tree-like multibody systems

The *inverse dynamics* concerns the computation of the generalized forces/torques $Q$ that have to be applied to the joints of a MBS for a given motion $\{q, \dot{q}, \ddot{q}\}$, in the presence of external forces and torques acting on the system:

$$Q = \Phi(q, \dot{q}, \ddot{q}, F_{ext}, L_{ext}, g) \tag{7.3}$$

This system is *implicit* with respect to the generalized joint accelerations $\ddot{q}$ (order-$N$ or $\mathcal{O}(N)$ formulation) and is therefore preferred to the *semi-explicit* form (7.2), which requires $\mathcal{O}(N^2)$ operations and is thus not optimal for real-time process applications (e.g. computed-torque control algorithm).

# 7.3 Newton-Euler recursive formalism

## 7.3.1 Introduction

The Newton-Euler recursive formalism (NER) [81] is among the most efficient techniques for generating the equations of motion of multibody systems in relative coordinates.

The NER formalism was initially developed for control purposes in the field of robotics [81], allowing to compute with a minimum number of arithmetic operations the *inverse* dynamic model (7.3) of *serial* manipulators. As we already mentioned, this form is *implicit* with respect to the generalized joint accelerations $\ddot{q}$, and therefore not suitable for computer time integration purposes. Hence, a modified NER scheme is used (see [3]) in order to obtain *recursively* the *semi-explicit* form (7.2) of the dynamic equations of tree-like MBS.

Being implicit or semi-explicit, the two main steps of NER formalism consist in:

- a forward kinematics computation of the velocity and acceleration vectors, conducted from the root (base inertial body 0) of the tree-like MBS to the leaf bodies;

- a backward dynamics computation of the joint forces and torques of the MBS, conducted from the leaf bodies to the root.

## 7.3.2 Forward recursive kinematics

Consider a rigid body $i$, carried by a parent body $h$ via joint $i$ (Figure 7.1).

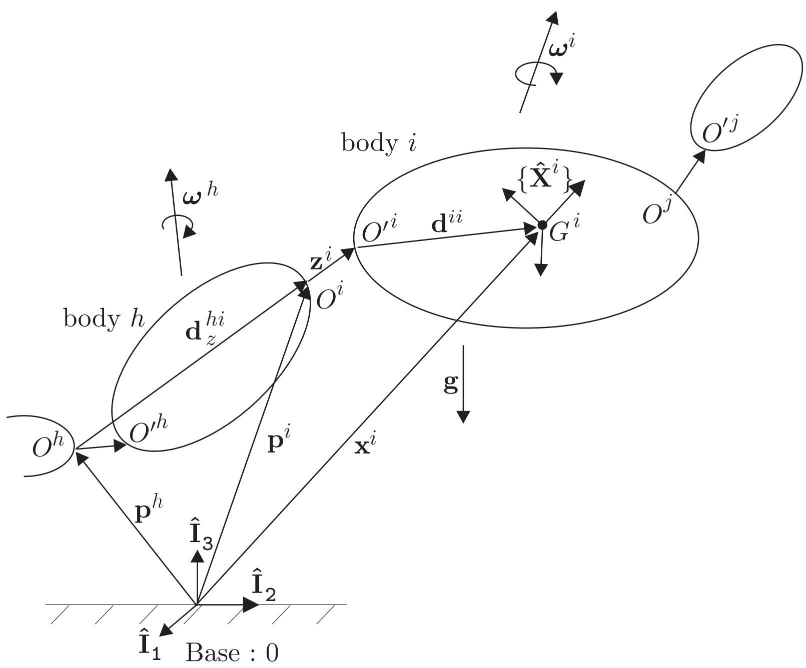



Figure 7.1: NER forward kinematics for body $i$

For body $i$, we can define the absolute position vector $\mathbf{p}^i = \mathbf{p}^h + \mathbf{d}_z^{hi}$, the absolute angular velocity vector $\boldsymbol{\omega}^i = \boldsymbol{\omega}^h + \boldsymbol{\Omega}^i = \boldsymbol{\omega}^h + \boldsymbol{\varphi}^i \dot{q}^i$ and the absolute angular and linear acceleration vectors:

$$\dot{\boldsymbol{\omega}}^i = \dot{\boldsymbol{\omega}}^h + \tilde{\boldsymbol{\omega}}^i \boldsymbol{\varphi}^i \dot{q}^i + \boldsymbol{\varphi}^i \ddot{q}^i \tag{7.4}$$

$$\ddot{\mathbf{p}}^i = \ddot{\mathbf{p}}^h + (\tilde{\dot{\boldsymbol{\omega}}}^h + \tilde{\boldsymbol{\omega}}^h \tilde{\boldsymbol{\omega}}^h)\mathbf{d}_z^{hi} + 2\tilde{\boldsymbol{\omega}}^h \boldsymbol{\psi}^h \dot{q}^h + \boldsymbol{\psi}^h \ddot{q}^h \tag{7.5}$$

Then, the following quantity can be *recursively* formulated:

$$\boldsymbol{\alpha}^i = \boldsymbol{\alpha}^h + \boldsymbol{\beta}^h \mathbf{d}_z^{hi} + 2\tilde{\boldsymbol{\omega}}^i \boldsymbol{\psi}^i \dot{q}^i + \boldsymbol{\psi}^i \ddot{q}^i \tag{7.6}$$

where $\boldsymbol{\alpha}^i \triangleq \ddot{\mathbf{p}}^i + 2\tilde{\boldsymbol{\omega}}^i \boldsymbol{\psi}^i \dot{q}^i + \boldsymbol{\psi}^i \ddot{q}^i - \mathbf{g}$ and $\boldsymbol{\beta}^i \triangleq \tilde{\dot{\boldsymbol{\omega}}}^i + \tilde{\boldsymbol{\omega}}^i \tilde{\boldsymbol{\omega}}^i$.

The generalized mass matrix of the semi-explicit form (7.2) can be expressed by isolating the generalized accelerations in the recursive equations (7.4) and (7.6), knowing that the accelerations appear linearly in these equations:

$$\dot{\boldsymbol{\omega}}^i = \sum_{k:k\leq i} \mathbf{O}_M^{ik} \ddot{q}^k + \dot{\boldsymbol{\omega}}_c^i \quad , \quad \boldsymbol{\alpha}^i = \sum_{k:k\leq i} \mathbf{A}_M^{ik} \ddot{q}^k + \boldsymbol{\alpha}_c^i \quad , \quad \boldsymbol{\beta}^i = \sum_{k:k\leq i} \mathbf{B}_M^{ik} \ddot{q}^k + \boldsymbol{\beta}_c^i \tag{7.7}$$

where the subscripts $M$ and $c$ stand for the generalized mass matrix and the vector $c$ of the semi-explicit form (7.2) of dynamic MBS equations, respectively.

The recursive computation of the last three equations can then be performed as in the original Newton-Euler recursive scheme for the inverse dynamics of the system. In vector form the algorithm reads:

*Initialization*:

$$\alpha_c^0 = -\mathbf{g} \; ; \; \boldsymbol{\omega}^0 = 0 \; ; \; \dot{\boldsymbol{\omega}}_c^0 = 0 \; ; \; \mathbf{O}_M^{ik} = 0 \; ; \; \mathbf{A}_M^{ik} = 0$$
$$(\forall i = 0 : N^{body}, \forall k = 0 : i)$$

*Recursion*:
For $i = 0 : N^{body}$

$$\begin{aligned} h &= \text{inbody(i) (index of the parent body)} \\ \boldsymbol{\omega}^i &= \boldsymbol{\omega}^h + \dot{q}^i \boldsymbol{\varphi}^i \\ \dot{\boldsymbol{\omega}}_c^i &= \dot{\boldsymbol{\omega}}_c^h + \tilde{\boldsymbol{\omega}}^i \boldsymbol{\varphi}^i \dot{q}^i \\ \boldsymbol{\alpha}_c^i &= \boldsymbol{\alpha}_c^h + \boldsymbol{\beta}_c^h \mathbf{d}_z^{hi} + 2\tilde{\boldsymbol{\omega}}^i \boldsymbol{\psi}^i \dot{q}^i \\ \boldsymbol{\beta}_c^i &= \tilde{\dot{\boldsymbol{\omega}}}_c^i + \tilde{\boldsymbol{\omega}}^i \tilde{\boldsymbol{\omega}}^i \end{aligned} \tag{7.8}$$

For $k = 1 : i$

$$\begin{aligned} \mathbf{O}_M^{ik} &= \mathbf{O}_M^{hk} + \delta^{ki} \boldsymbol{\varphi}^i \\ (\mathbf{B}_M^{ik} &= \tilde{\mathbf{O}}_M^{ik}) \\ \mathbf{A}_M^{ik} &= \mathbf{A}_M^{hk} + \tilde{\mathbf{O}}_M^{hk} \mathbf{d}_z^{hi} + \delta^{ki} \boldsymbol{\psi}^i \\ \text{with } \delta^{ki} &= 1 \text{ if } k = i \text{ and } 0 \text{ otherwise} \end{aligned} \tag{7.9}$$

end
end.

For practical (computer) implementations, all the vector computations of this algorithm have to be transformed into matrix forms by expressing all vectors and tensors in their appropriate frames, e.g. $\boldsymbol{\psi}^i = [\hat{\mathbf{X}}^i]^T \psi^i$, $\mathbf{z}^h = [\hat{\mathbf{X}}^i]^T z^h$, etc. The matrix form of the recursive scheme can be found in [3].

### 7.3.3 Backward recursive dynamics

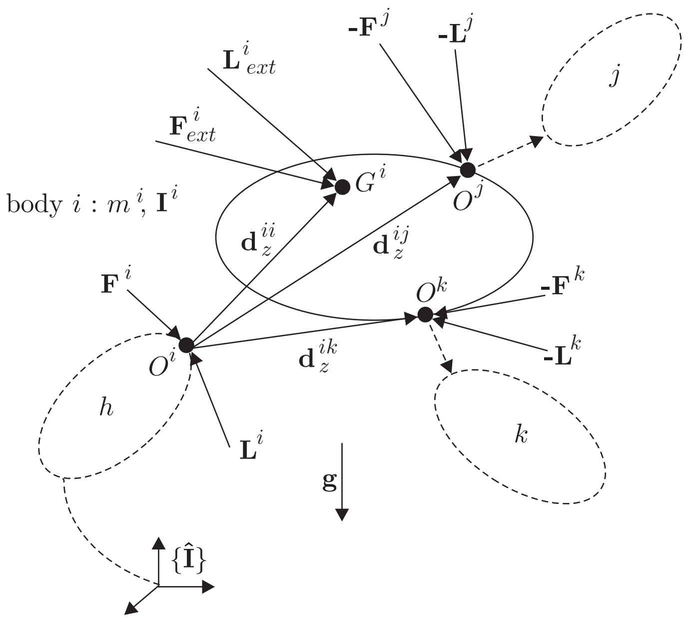



Figure 7.2: NER backward dynamics for body $i$

In order to develop the recursive formalism for the MBS dynamics, the vector equations of motion of body $i$ can be obtained easily on the basis of Newton-Euler equations. According to the first of them, the translational vector equation of motion of body $i$ reads:

$$\mathbf{F}^i - \sum_{j \in \bar{i}} \mathbf{F}^j + \mathbf{F}^i_{ext} + m^i \mathbf{g} = m^i \ddot{\mathbf{x}}^i \tag{7.10}$$

where $\mathbf{F}^i$, defined in Section 5.4, represents the resultant force acting on body $i$ through the joint $i$, evaluated at the connection point $O^i$ (see Figure 7.2), $\mathbf{F}^i_{ext}$ is the external resultant force (except the gravity force component), applied to the mass center $G^i$ of body $i$, and $\sum_{j \in \bar{i}}$ denotes the summation over all the child bodies of body $i$ ($j$ and $k$ in Figure 7.2).

Using equation (7.5) and the definitions from (7.6), we can rewrite equation (7.10) as:

$$\mathbf{F}^i = \sum_{j \in \bar{i}} \mathbf{F}^j + \mathbf{W}^i \tag{7.11}$$

with $\mathbf{W}^i = m^i(\boldsymbol{\alpha}^i + \boldsymbol{\beta}^i \mathbf{d}^{ii}_z) - \mathbf{F}^i_{ext}$.

Similarly, the Euler rotational equation of motion of body $i$ with respect to its center of mass $G^i$ can be written as:

$$\mathbf{L}^i = \sum_{j \in \bar{i}} (\mathbf{L}^j + \tilde{\mathbf{d}}_z^{ij} \mathbf{F}^j) + \tilde{\mathbf{d}}_z^{ii} \mathbf{W}^i - \mathbf{L}_{ext}^i + \mathbf{I}^i \dot{\boldsymbol{\omega}}^i + \tilde{\boldsymbol{\omega}}^i \mathbf{I}^i \boldsymbol{\omega}^i \tag{7.12}$$

where $\mathbf{I}^i$ is the inertia tensor of body $i$ with respect to its center of mass $G^i$, $\mathbf{L}^i$ represents the resultant joint torque acting on body $i$ via joint $i$ and $\mathbf{L}_{ext}^i$ is the external resultant torque (including the moments of all the external loads with respect to $G^i$) applied on body $i$.

Equations (7.11) and (7.12) can be recursively computed starting from the leaf bodies and going back to the base body 0 of the multibody system. This leads to the "classical" backward recursion procedure of the Newton-Euler scheme that provides (after projection onto the joint axes) the *inverse dynamic model* of the system in its implicit form (7.3). In order to get the generalized mass matrix $M$ and the vector $c$, the quantities $\mathbf{F}^i$ and $\mathbf{L}^i$ need to be split up and the contribution of each generalized acceleration $\ddot{\mathbf{q}}^i$ isolated:

$$\mathbf{F}^i = \sum_k \mathbf{F}_M^{ik} \ddot{q}^k + \mathbf{F}_c^i \quad , \quad \mathbf{W}^i = \sum_k \mathbf{W}_M^{ik} \ddot{q}^k + \mathbf{W}_c^i \quad , \quad \mathbf{L}^i = \sum_k \mathbf{L}_M^{ik} \ddot{q}^k + \mathbf{L}_c^i \tag{7.13}$$

These new quantities can also be computed in a recursive manner, introducing the relations (7.7) into the dynamic equations (7.11) and (7.12), finally obtaining the following computational recursive scheme:

For $i = N^{body} : 1$

$$\begin{aligned} \mathbf{W}_c^i &= m^i(\boldsymbol{\alpha}_c^i + \boldsymbol{\beta}_c^i \mathbf{d}_z^{ii}) - \mathbf{F}_{ext}^i \\ \mathbf{F}_c^i &= \sum_{j \in \bar{i}} \mathbf{F}_c^j + \mathbf{W}_c^i \\ \mathbf{L}_c^i &= \sum_{j \in \bar{i}} (\mathbf{L}_c^j + \tilde{\mathbf{d}}_z^{ij} \mathbf{F}_c^j) + \tilde{\mathbf{d}}_z^{ii} \mathbf{W}_c^i - \mathbf{L}_{ext}^i + \mathbf{I}^i \dot{\boldsymbol{\omega}}_c^i + \tilde{\boldsymbol{\omega}}^i \mathbf{I}^i \boldsymbol{\omega}^i \end{aligned} \tag{7.14}$$

For $k = 1 : i$

$$\begin{aligned} \mathbf{W}_M^{ik} &= m^i(\mathbf{A}_M^{ik} + \tilde{\mathbf{O}}_M^{ik} \mathbf{d}_z^{ii}) \\ \mathbf{F}_M^{ik} &= \sum_{j \in \bar{i}} \mathbf{F}_M^{jk} + \mathbf{W}_M^{ik} \\ \mathbf{L}_M^{ik} &= \sum_{j \in \bar{i}} (\mathbf{L}_M^{jk} + \tilde{\mathbf{d}}_z^{ij} \mathbf{F}_M^{jk}) + \tilde{\mathbf{d}}_z^{ii} \mathbf{W}_M^{ik} + \mathbf{I}^i \mathbf{O}_M^{ik} \end{aligned} \tag{7.15}$$

end

end.

We must emphasize at this point that the forward kinematic and backward dynamic algorithms allow for computing by means of (7.13a) and (7.13c) the vector force $\mathbf{F}^i$ and torque $\mathbf{L}^i$ applied to body $i$ through joint $i$.

Just as it was mentioned for the forward recursive kinematics, the backward dynamic algorithm has to be transformed into matrix form, in order to be appropriate for computer implementation. This form can be found in [3] as well.

The $i^{th}$ joint equation can be obtained by projecting the first and the last of the vector equations of motion (7.13), according to the vector equation:

$$Q^i = \mathbf{F}^i.\boldsymbol{\psi}^i + \mathbf{L}^i.\boldsymbol{\varphi}^i \tag{7.16}$$

giving the contributions $Q^i$ of each joint force $\mathbf{F}^i$ and torque $\mathbf{L}^i$ to the generalized force, associated with the generalized coordinate $q^i$.

The NER formalism was implemented in the ROBOTRAN symbolic generation software [5], systematically used in the computer implementations of the kinematic and dynamic solution algorithms for parallel manipulators, developed in the present work. We will give explanations on this matter in the following chapters.

## 7.4 Direct dynamics of closed-loop multibody systems

When the joint coordinates of a multibody system are not independent, it is usually called *constrained* MBS, because of the existing dependencies between the coordinates, known as *constraints* and expressed by (constraint) mathematical equations.

For $m$ independent loop-closure, user or joint constraints, applied on the system, the unknown generalized constraint forces can be computed from the $m$ unknown Lagrange multipliers $\lambda$ and the motion of the constrained multibody system can then be rewritten in its semi-explicit form:

$$M(q)\ddot{q} + c(q, \dot{q}, F_{ext}, L_{ext}, g) = Q + J_c(q)^T\lambda \tag{7.17}$$

in which the matrix $J_c(q)$ is the *constraint* Jacobian we already defined in Section 6.2, and the coefficients $\lambda \in \Re^m$ represent the *Lagrange multipliers*, associated with the *explicit* constraints.

The system 7.17, together with the system of constraint equations gives a set of $n+m$ differential/algebraic equations (DAE) in $n+m$ unknowns $q \in \Re^n$ and $\lambda \in \Re^m$:

$$\begin{aligned} M(q)\ddot{q} + c(q, \dot{q}, F_{ext}, L_{ext}, g) &= Q + J_c(q)^T\lambda \\ h(q) &= 0 \\ \dot{h}(\dot{q}, q) &= J_c(q)\dot{q} = 0 \\ \ddot{h}(\ddot{q}, \dot{q}, q) &= J_c(q)\ddot{q} + \dot{J}_c(\dot{q}, q)\dot{q} = 0 \end{aligned} \tag{7.18}$$

When the constraint equations on the generalized coordinates are nonlinear, usually it is difficult to express analytically some of the coordinates in terms of the others. However, if the $m$ constraints are *independent*, the reduction of system (7.18) to a purely differential one can be obtained applying to it the coordinate partitioning method described in Section 6.2.2.

Indeed, assuming that since the constraints are independent, the constraint Jacobian is of full rank $m$ and, according to the implicit function theorem, $m$ generalized coordinates contained in the vector $q$ can be expressed locally as functions of the $(n-m)$ others.

### 7.4.1 Plant dynamics reduction using the coordinate partitioning method

Let us first partition the generalized mass matrix $M$, the vectors $c$ and $Q$, and the constraint Jacobian $J_c(q)$ in the first set of equations of the system (7.18) according to the coordinate partitioning (3.2):

$$\begin{pmatrix} M_{uu} & M_{uv} \\ M_{vu} & M_{vv} \end{pmatrix} \begin{pmatrix} \ddot{q}_u \\ \ddot{q}_v \end{pmatrix} + \begin{pmatrix} c_u \\ c_v \end{pmatrix} = \begin{pmatrix} Q_u \\ Q_v \end{pmatrix} + \begin{pmatrix} J_{c_u}^T \\ J_{c_v}^T \end{pmatrix} \lambda \tag{7.19}$$

Since $J_{c_v}(q)$ is of full rank, eliminating the unknown multipliers $\lambda$ using the lower part of system (7.19) produces:

$$\begin{pmatrix} M_{uu} & M_{uv} \end{pmatrix} \begin{pmatrix} \ddot{q}_u \\ \ddot{q}_v \end{pmatrix} + B_{vu}^T \begin{pmatrix} M_{vu} & M_{vv} \end{pmatrix} \begin{pmatrix} \ddot{q}_u \\ \ddot{q}_v \end{pmatrix} + c_u + B_{vu}^T c_v = Q_u + B_{vu}^T Q_v \tag{7.20}$$

The generalized positions $q_v$ from (6.12a) are solved using the Newton-Raphson method (see eq.(6.13)), the generalized velocities $\dot{q}_v$ and accelerations $\ddot{q}_v$ are given by (6.12b) and (6.12c), respectively, and can also be eliminated from the differential equations (7.20). This leads to the final *reduced* system of equations for the closed-loop MBS direct dynamics:

$$\begin{aligned} &\left(M_{uu} + M_{uv}B_{vu} + B_{vu}^T M_{vu} + B_{vu}^T M_{vv} B_{vu}\right) \ddot{q}_u \\ &\qquad + \left(M_{uv} + B_{vu}^T M_{vv}\right) d + (c_u + B_{vu}^T c_v) = Q_u + B_{vu}^T Q_v \end{aligned}$$

which can be concisely written as:

$$\mathcal{M}_r(q_u)\ddot{q}_u + c_r(\dot{q}_u, q_u) = \mathcal{Q}_r \tag{7.21}$$

with:

- $\mathcal{M}_r(q_u) = M_{uu} + M_{uv}B_{vu} + B_{vu}^T M_{vu} + B_{vu}^T M_{vv} B_{vu}$,
- $c_r(\dot{q}_u, q_u) = \left(M_{uv} + B_{vu}^T M_{vv}\right) d + (c_u + B_{vu}^T c_v)$,
- $\mathcal{Q}_r = Q_u + B_{vu}^T Q_v$,

Thanks to this final elimination, the set of purely differential equations (7.21) is referred to as the *reduced* equations of motion of the constrained system described in terms of the $n - m$ independent generalized coordinates $q_u$. The sequence of computations, producing system (7.21), when dealing with constrained multibody system in the presence of internal (joint) friction forces, will be illustrated later on the flowchart in Figure 7.3. This sequence will be commented in Section 7.7.1.

Once the MBS motion computed, it can be interesting to find the values of some (or all) of the constraint forces during the motion. In order to do this, the Lagrange multipliers $\lambda$ have to be computed. This can be done by isolating them in the second raw of the system (7.19):

$$\lambda = (J_{c_v}^T)^{-1} \left\{ (M_{vu}\ M_{vv}) \begin{pmatrix} \ddot{q}_u \\ \ddot{q}_v \end{pmatrix} + c_v - Q_v \right\} \tag{7.22}$$

and using the relations (6.12c) gives:

$$\lambda = (J_{c_v}^T)^{-1} \{(M_{vu} + M_{vv}B_{vu})\ddot{q}_u + c_v - Q_v + M_{vv}d\} \tag{7.23}$$

which allows us to compute afterward the generalized constraint forces by means of the obtained Lagrange multipliers.

The Lagrange multipliers have to be computed, for example, when joint friction forces, depending on them, are accounted for in the dynamic models, as it will be shown in Section 7.6.

## 7.5 Inverse dynamics of closed-loop multibody systems

If we refer to the form of eq. (7.3) for closed-loop MBS and apply again the coordinate partitioning method with independent/dependent ($\{u, v\}$) partitioning, we obtain the following implicit partitioned form of the closed-loop system dynamics:

$$\begin{pmatrix} \Phi_u \\ \Phi_v \end{pmatrix} = \begin{pmatrix} Q_u \\ Q_v \end{pmatrix} + \begin{pmatrix} J_{c_u}^T \\ J_{c_v}^T \end{pmatrix} \lambda \tag{7.24}$$

from the dependent-coordinate sub-system of which we express once again and substitute the Lagrange multipliers in the sub-system of equations for the independent coordinates in order to eliminate them, obtaining:

$$\Phi_u = Q_u + B_{vu}^T (Q_v - \Phi_v) \tag{7.25}$$

or, expressed with respect to the generalized (joint) forces/torques:

$$A \begin{bmatrix} Q_u \\ Q_v \end{bmatrix} = \left(\Phi_u + B_{vu}^T \Phi_v\right) \tag{7.26}$$

where $A = \begin{bmatrix} E & B_{vu}^T \end{bmatrix}$.

From the actuation point of view, the coordinate partitioning into *active* and *passive* generalized coordinates, explained in Section 6.2.2, will be preferred, because the consistent part of the vector $Q$ correspond to loads applied on the actuated (active) joints. The *independent/dependent* partitioning, however, will be preserved to provide numerically stable computation of $q_v$, $\dot{q}_v$, $\ddot{q}_v$ and $B_{vu}$, defined by 6.12 and giving the complete set of generalized coordinates, velocities and accelerations for every time step of a given trajectory, from which the corresponding generalized joint (actuator) torques can be computed.

Applying a $\{a, p\}$ partitioning to the system of equations (7.3) for closed-loop MBS and eliminating the unknown Lagrange multipliers gives:

$$A_p \begin{bmatrix} Q_a \\ Q_p \end{bmatrix} = \left(\Phi_a + B_{pa}^T \Phi_p\right) \tag{7.27}$$

with $B_{pa} = -J_{c_p}^{-1} J_{c_a}$ and $A_p = \begin{bmatrix} E & B_{pa}^T \end{bmatrix}$.

Recalling that $Q_p = 0$, this leads to:

$$Q_a = \Phi_a + B_{pa}^T \Phi_p \tag{7.28}$$

for the case of non-redundant actuation. i.e. number of actuators equal to $d_{mbs}$. This system can be combined with (6.12) to give the *complete* set of equations, used to compute the inverse dynamics of a parallel manipulator for a given trajectory, prescribed in terms of independent joint generalized coordinates:

$$\begin{array}{rcl} q_v & = & f_q(q_u) \\ \dot{q_v} & = & B_{vu}\dot{q_u} \\ \ddot{q_v} & = & B_{vu}\ddot{q_u} + d \\ Q_a & = & \Phi_a + B_{pa}^T \Phi_p \end{array} \tag{7.29}$$

## 7.6 Dynamic model extension: joint friction forces

In practical situations, the manipulator joins are not dynamically *ideal*, i.e. non negligible internal friction forces appear in them during motion, giving birth to resistive efforts that dissipate energy and cause wear. The friction forces depend, in general, on the material and the state (roughness) of the surfaces in contact, on the presence of lubricant (dry or viscous friction forces), on their relative speed, etc.

Targeting better accuracy of our dynamic models that will allow, for instance, more precise motion simulations and control of the parallel manipulators considered, and better experimental prototype results, we shall introduce components of two principal types of internal friction in the manipulator MBS models:

- Friction forces, depending *linearly* on the normal components of the internal reaction (constraint) forces (Lagrange multipliers):

  $$\Gamma_c = \Gamma_c(\mu_0, \mu, \lambda) \tag{7.30}$$

  with $\mu_0$ being the joint static friction coefficient and $\mu$ – the joint dry sliding friction coefficient, required by the constitutive equations of $\Gamma_c$ (Coulomb friction laws),

- Friction forces, depending *linearly* on the generalized joint velocities $\dot{q}$ (*viscous* friction forces):

  $$\Gamma_v = \Gamma_v(\mu_v, \dot{q}) \tag{7.31}$$

  with $\mu_v$ being the kinematic viscosity coefficient of the lubricating fluid.

Hence, the systems (7.17) and (7.3) can be rewritten as:

$$M(q)\ddot{q} + c^*(q, \dot{q}, F_{ext}, L_{ext}, \Gamma_c, \Gamma_v, g) = Q + J_c^T \lambda \tag{7.32}$$

and

$$\Phi^*(\ddot{q}, \dot{q}, q, F_{ext}, L_{ext}, \Gamma_c, \Gamma_v, g) = Q + J_c^T(q)\lambda \tag{7.33}$$

where:

- $c^*(\dot{q}, q, F_{ext}, L_{ext}, \Gamma_c, \Gamma_v, g) \triangleq c(q, \dot{q}, F_{ext}, L_{ext}, g) - \Gamma_c(\mu, \lambda) - \Gamma_v(\dot{q})$,
- $\Phi^*(\ddot{q}, \dot{q}, q, F_{ext}, L_{ext}, \Gamma_c, \Gamma_v, g) \triangleq M(q)\ddot{q} + c^*(\dot{q}, q, F_{ext}, L_{ext}, \Gamma_c, \Gamma_v, g)$,
- $Q$ are the generalized torques due to the actuators only.

The two systems (7.32) and (7.33) can then be partitioned according to (3.2) and (7.19), leading to the following forms of constrained direct and inverse dynamics of the multibody system including joint friction:

$$\mathcal{M}_r(q_u)\ddot{q}_u + c_r^* = \mathcal{Q}_r \tag{7.34}$$

and

$$A\ Q = \Phi_u^* + B_{vu}^T \Phi_v^* \tag{7.35}$$

The form (7.34), for instance, can be used to perform plant dynamics time-integration and control simulations on the basis of actuator loads computed via (7.35). A computer implementation of this procedure is described in Section 7.7.1.

## 7.7 Closed-loop multibody system dynamics computation

The equations of motion (7.18) of closed-loop multibody systems are not only differential, but differential-algebraic, abbreviated as DAE [82]. Their time integration in the field of multibody dynamics can be treated using three principal types of methods: constraint stabilization [83], coordinate partitioning [4] and direct methods [84].

- The constraint stabilization [83], probably one of the first approaches used to solve DAE systems, consists in transforming the DAE into a system of ordinary differential equations(ODE), by differentiating the constraints and introducing *stabilization* terms. The constraints are thus not satisfied exactly but oscillate with a given stabilization period and damping constants around their exact solutions.
- The generalized coordinate partitioning method [4] also transforms the original DAE system into an ODE, but, as already shown in Section 7.4.1, does it in a quite different way, by a system *reduction* that includes an accurate algebraic solution of the constraints at position, velocity and acceleration level.
- The direct methods [84] demonstrate an impressive efficiency in terms of computational speed [85]. The basic idea behind them is to solve the complete DAE set directly by applying, for instance, a backward differentiation formula to the variables and by solving the resulting system at each step using Newton-Raphson type methods (e.g. [86]). The major drawback of this type of methods is the presence of certain lack of versatile and robust DAE integration schemes, despite of the development of high level mathematical theories targeting increase in their stability

and accuracy. It is also worth mentioning that DAE integrators can not be found implemented in many of the most widespread computational software packages, as MATLAB, for example.

All these considerations justify the choice of the second type of time integration methods for the models of parallel manipulators, developed in the present work, the constraint solution precision being among the main characteristics they have to hold. Other reasons, important to us, lie behind this choice. Firstly, in the framework of the present study the coordinate partitioning method is extensively used, both for reduction of the system kinematics and dynamics, and for finding suitable manipulator actuation schemes in the presence of force singularities. Secondly, the models developed for the purposes of real-time plant dynamics simulation and control in the SIMULINK environment must hold an ODE-form, as the time integration methods that SIMULINK employs are not dedicated to treatment of DAE systems. And, finally, the developments of the present research "stay in the good tradition" of the robust computational methods, elaborated at the Université catholique de Louvain, that rely on the coordinate partitioning method [4].

However, the time integration of the systems (7.21) or (7.34) will become problematic when $J_{c_v}$ approaches singularity. In order to eliminate this problem, i.e. to make the integration procedure robust with respect to numeric constraint closure problems, a special time-integration process is developed, based as well on the coordinate partitioning method and consisting in a *piecewise*[2] time integration of the parallel manipulator direct dynamic model.

### 7.7.1 Direct dynamics numerically robust time integration

The need of numerically robust and computationally efficient time integration procedure for parallel manipulator direct dynamic models is of essential importance in this study, because real-time plant dynamics control simulations are targeted as an approach validation and controller-tunning tool. In the lines hereunder, we reveal the development of an algorithm for such a procedure, based on system reduction via coordinate partitioning, and its subsequent computer implementation.

Let us first explain the sequence of computations, necessary to obtain the independent generalized accelerations $\ddot{q}_u$ from the reduced direct dynamics (7.34) at a given integration time $t$. This sequence is illustrated by the flowchart of Figure 7.3.

For a given independent/dependent coordinate partition ensuring well-conditioned $J_{c_v}$, an initial configuration $q_{u0}, \dot{q}_{u0}$ is entered as input data. At every integration time step, after computing $q_v$, $B_{vu}$, $\dot{q}_v$, the components of $M$ and $c$ are evaluated using the Newton-Euler recursive scheme and the reduced direct dynamics (7.34) is constructed. Then, the independent accelerations are calculated from it by means of linear algebra techniques, and time-integration is performed.

[2]independent from the piecewise actuation process that will be explained in the next part!

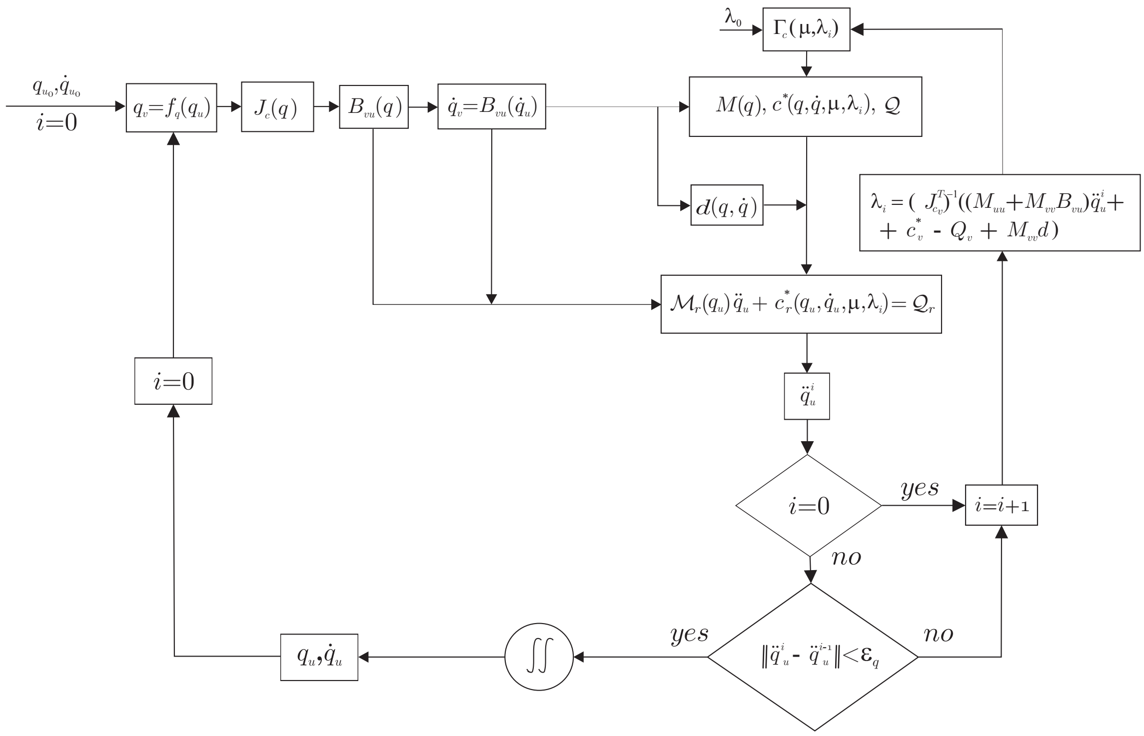


Figure 7.3: Direct dynamics coordinate partitioning and time-integration scheme with joint friction.

It is important to mention that the presence of internal friction forces of the form (7.30), which can even depend nonlinearly on the Lagrange multipliers, makes impossible the multiplier elimination within a unique system reduction. Therefore, the algorithm includes an iterative computation procedure, based on the convergence in terms of $\ddot{q}_u$: at a time $t$ of the integration, an implicit iterative procedure (see Figure 7.3) performs iterations $i$ over $\lambda$ (starting with $\lambda_0$ equal to the last iteration values for $\lambda$ at time $t - \Delta t$, except for $t = 0$, for which $\lambda_0 = 0$), until good convergence of $\ddot{q}_u^i$ to $\ddot{q}_u^{i-1}$ is achieved[3].

Let us now describe the *piecewise* nature of the *complete* direct dynamics computation algorithm (Figure 7.4). As numeric loop closure problems have to be avoided in order to integrate the plant dynamics for the whole trajectory time $[t_0 \; t_{final}]$, at every complete time integration step, the conditioning of $J_{c_v}$ is compared to a given ill-conditioning indication value and the Newton-Raphson method convergence – to a convergence indicator[4]. In case of bad Newton-Raphson convergence or Jacobian conditioning, the integration is interrupted in order to perform a local LU-refactorization with column pivoting of the constraint Jacobian $J_c$, from which a new, locally best-conditioned $J_{c_v}$ is extracted. The computational procedure of Figure 7.3 is then relaunched for the new $\{q_u^j, q_v^j\}$ partition and a set of new initial conditions $(q_{u0}^j, \dot{q}_{u0}^j)$ equal to the corresponding values of $q$, stored at the interruption of the integrator.

[3]This convergence is generally not problematic, but is not guaranteed either: more sophisticated numerical methods like Newton-Raphson could be envisaged if needed.

[4]Maximal admissible number of iteration steps is used by the author for this purpose.

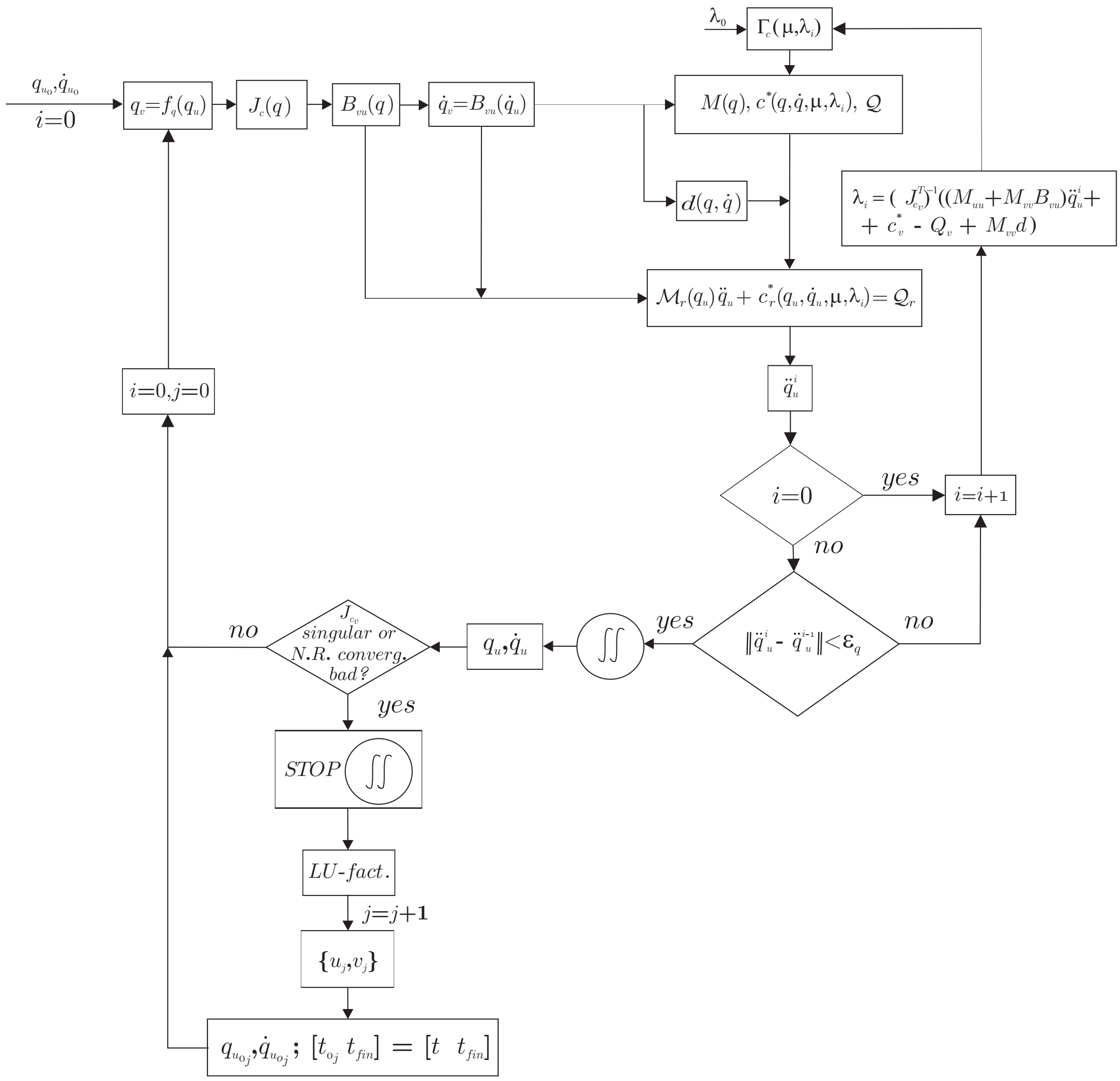



Figure 7.4: Direct dynamics piecewise time-integration scheme.

This re-partitioning process is repeated as many times $(j)$ during the whole time integration, as there are ill-conditionings of $J_{c_v}$ detected. The integration time interval segmentation that is due to the deliberate integrator interruptions (and corresponds in fact to a trajectory segmentation) justifies the utilization of the term "piecewise" for the direct dynamics solution.

We would like to emphasize the fact that the piecewise time integration procedure relies on time integration methods, e.g. the Runge-Kutta or Dormand-Price method[5]. These methods were found stable for all our simulations with respect to the deliberate restarting of the integration process, provided the initial conditions are correct for every start.

Independently of the proposed approach, other means of circumventing the constraint Jacobian ill-conditioning problems exist, like for instance, the use of DAE integration methods. As we mentioned above, they solve numerically the whole set of equations

[5]Found in the MATLAB routines *ode45.m* and *ode113.m*, used in this work.

(including the algebraic constraints), but cannot provide the reduced state-space form of the plant dynamics. Thus they are neither suitable, nor versatile with respect to control simulation and real-time robot control applications, targeted in the present work.

### Algorithmic implementation example

Let us give an example validating the proposed algorithm for time integration. The validation is based on the direct dynamic model of a 3-d.o.f. planar parallel manipulator, extensively used by the author of this work as a test closed-loop MBS[6], as well as a conceptual model for the development of a prototype. The generation of the kinematics and the plant dynamics of the manipulator, as well as the algorithmic implementation of the piecewise time-integration procedure, have been performed using MATLAB/SIMULINK language and the multibody symbolic generation software ROBOTRAN [5].

The manipulator topology is schematically represented in Figure 7.5 (left): it consists in five bars interconnected by revolute joints (denoted $q_1$ ... $q_6$). The same figure shows the trajectory to be followed. It is characterized by a sinusoidal motion along the X-axis and a constant velocity motion along Y, and can be compared (because of the manipulator reconfiguration) to a "turning out of a sock", for instance. The same type of trajectory is used for the actuation strategy, detailed in the next part.

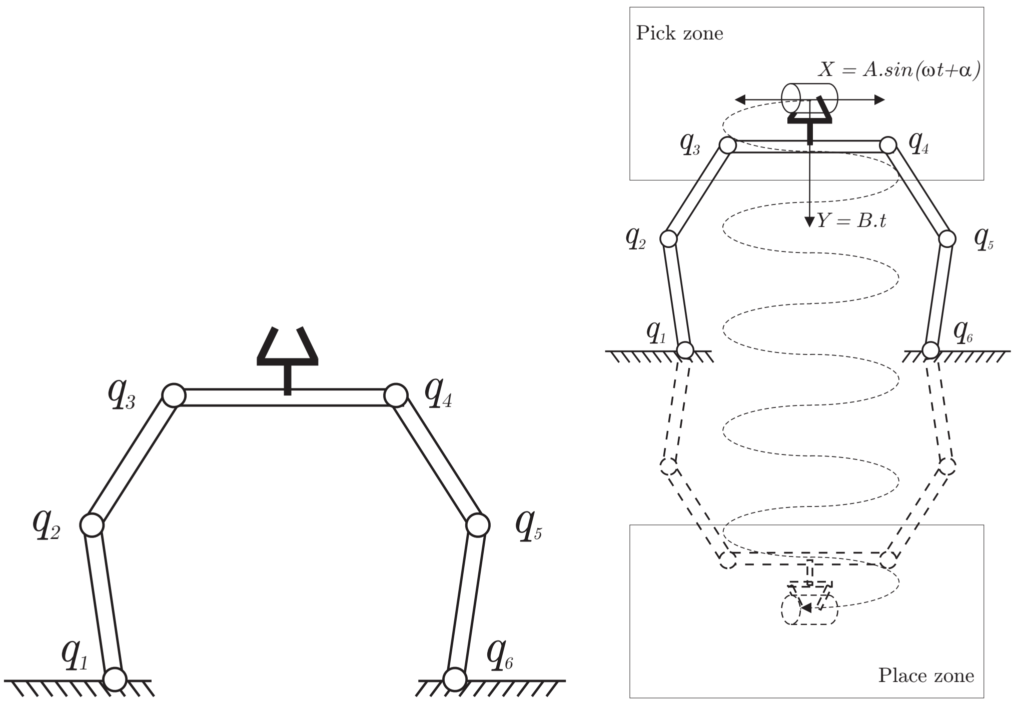



Figure 7.5: Topology representation of the planar parallel manipulator (left) and the example trajectory it follows (right).

In order to validate the piecewise time integration procedure, for the given trajectory

[6]along with a four-bar mechanism

two time integrations on the basis of an explicit Runge-Kutta (4,5) method[7] (Dormand-Prince pair) were performed applying a non-redundant actuation ($R_1$, $R_2$ and $R_3$ actuated) on the plant dynamics model, in the first of which the piecewise partitioning algorithm was deliberately deactivated. As shown in Figure 7.6 (left), the integration stopped at time $t = 1.17$ sec due to Newton-Raphson convergence problems, because for the chosen *constant* partition $\{u, v\}$ the corresponding sub-Jacobian $J_{c_v}$ becomes highly singular (ill-conditioned), indicated by the rise in its conditioning number and corresponding to the configuration shown on the right in the figure. When the piecewise coordinate partitioning, incorporated in the time integration algorithm, is not deactivated (normal algorithm operation), the integration is performed *completely* (6 second-long trajectory) and without convergence problems.

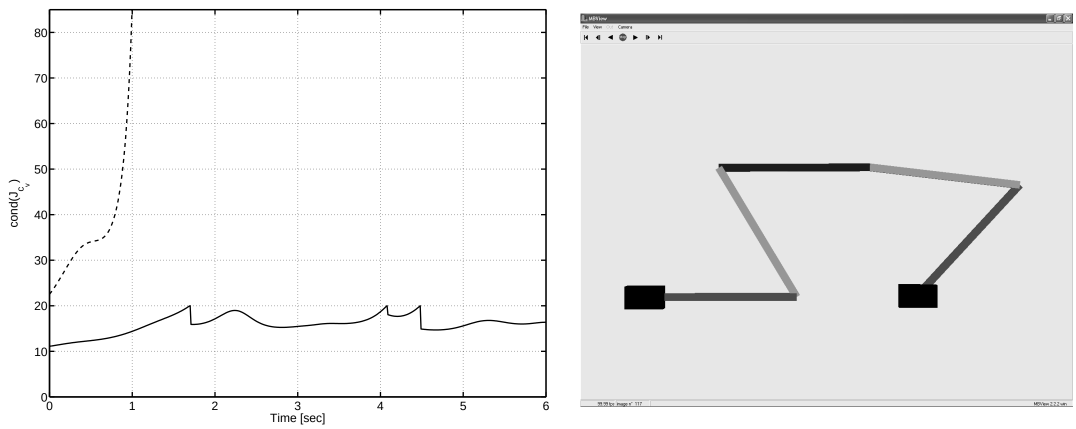


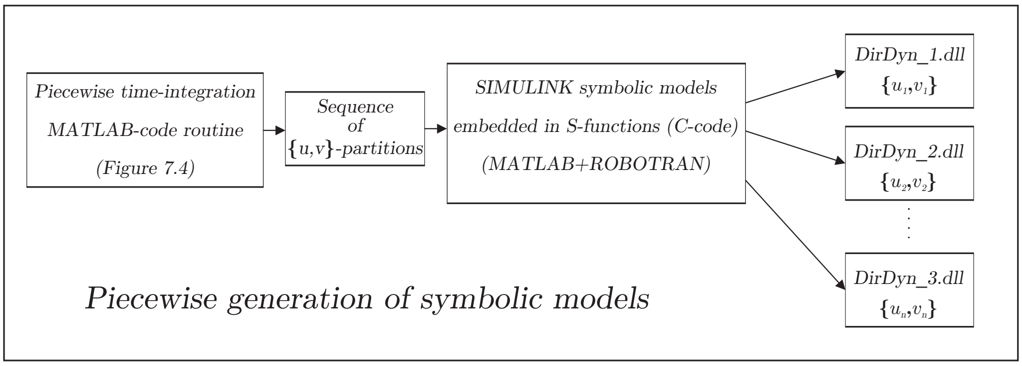

Figure 7.6: Validation results of the piecewise integration algorithm. Left – condition number of $J_{c_v}$ (dotted line for deactivated piecewise partitioning), right - the singularity corresponding to the integration interruption due to $J_{c_v}$ ill-conditioning.

On the basis of the time integration algorithm of Figure 7.4, a special MATLAB program was developed for the purposes of real-time control simulation performed in the SIMULINK environment. The process comprises the following steps (see Figure 7.7):

- For the given robot trajectory, a sequence of partitions $\{u, v\}$ was stored according to the procedure of Figure 7.4. For each of these partitions, a symbolic reduced plant dynamic model is generated by ROBOTRAN in the form of SIMULINK S-function (C-compiled routines).
- The main MATLAB program successively calls the S-functions until the complete integration performed. The initial conditions of every activated model (S-function) correspond to the manipulator configurations, for which a new $\{u, v\}$ re-partitioning has been performed.

This procedure is illustrated in Figure 7.7. Its main advantages are the following:

[7]The dedicated MATLAB routine ODE45.m was used



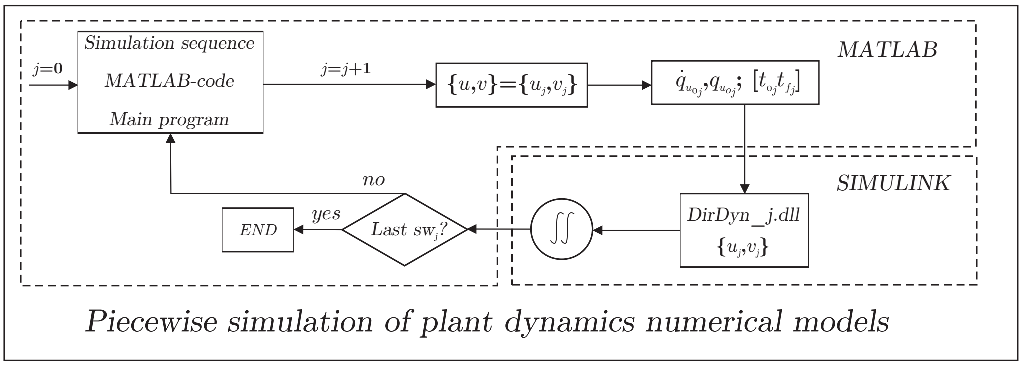




Figure 7.7: Plant dynamics piecewise generation and piecewise simulation scheme.

- Thanks to the fully symbolic generation of the reduced plant dynamics and the C-compiled form, far more than a real-time simulation is achieved. This allows us to make use of optimization procedures and design of experiments within reasonable computation time in order, for instance, to tune some relevant control parameters. This will be detailed in the next part.

- The availability of the successive plant dynamic models in SIMULINK is very beneficial. Indeed, being self-contained, they do not depend on specific multibody software and can easily be exploited, so as to tune any control algorithm for non-redundantly or redundantly actuated closed-loop MBS.

# Part III

# Redundant Actuation and Singularities

# Chapter 8

# Piecewise and redundant actuation

## 8.1 Methodology

In Chapter 4 we already gave definitions and a classification of the force redundancy, and commented on the principal fields of its application and methods for solution, found in the literature. In this part of our work we shall concentrate on the main scientific contributions it offers, namely, the development of parallel manipulator actuation strategy for elimination of force singularities and subsequent control applications of the solutions obtained.

As previously stated, a key point of the proposed strategy (already introduced in [36]) is that – in contrast to many approaches – it does not try to *avoid* singular configurations. On the contrary, taking profit from the fact that parallel singularities are *actuator-dependent*, the proposed actuation solutions are valid for trajectories that can include any possible configuration of the manipulator, be it singular or not. What is more, singularity avoidance strategies generally lead to a reduced robot workspace or additional task constraints, for instance.

Further, in Chapter 4 we specified that for the parallel manipulator applications presented, the force redundancy is achieved by redundantly actuating only the accessible (*passive*) joints (ex. Figure 4.6), and not changing the manipulator architecture by adding extra legs (parallel branches) with actuators (ex. Figure 4.7, left). Indeed, this situation has the disadvantage of reducing the manipulator workspace, due to additional kinematic loops introduced. This problem has already been commented in [38] by Dasgupta and Mruthyundjaya, who demonstrated as well the actuation redundancy effectiveness in terms of eliminating force singularities of parallel manipulators. In Chapter 4, we cited the recent contribution of Firmani and Podhorodeski [53], who showed how the force singularity manifold is reduced with each added actuator in different redundant actuator configurations, and the one of Krut, Company and Pierrot [54], in which they analyzed the velocity isotropy of parallel mechanisms with actuation redundancy, emphasizing the convenience of the latter for an improvement of the manipulator velocity performances.

Encompassing kinematic and dynamic considerations[1], the actuation approach we propose in this work delivers redundant actuation solutions, *effective* with respect to force singularity problems. Furthermore, these solutions will be advantageously exploited (see Chapter 9) by implementing them in control algorithms for redundantly actuated closed-loop MBS, tested via simulations and prototype experiments.

The proposed actuation approach is tested on a type of trajectory, corresponding to the one already depicted in Figure 7.5 (right). The trajectory is chosen such that force (parallel) singularities exist within it for certain actuator configurations. The interesting part of the trajectory is characterized by a motion, for which good performances in terms of overcoming resistance from internal and external loads at a given velocity are targeted. In this trajectory region, the manipulator must displace specific tools or objects in a smooth, continuous manner, eliminating the effects of force singularities that would appear for certain actuator configurations. Moreover, the prescribed end-effector trajectory must be followed with acceptable path tracking error and satisfying some optimality criteria (e.g. actuator torque limit or energy considerations).

Depending on the manipulator topology, the trajectory and the actuated joint choice criteria, the proposed actuation strategy produces as a result a non-redundant or redundant actuator configuration. For the second case of actuation, a convenient solution must be found to be compatible with the chosen optimality criteria.

Two of the solution criteria that we reviewed in Section 4.2.2 are used in the present study - a solution, mathematically equivalent to a two-norm torque minimization (using pseudoinverse matrix) and an infinity-norm torque minimization. We are aware that other criteria could be chosen, for which our methodology could be used straightforwardly, but to our opinion these two are appropriate for the considered actuation approach and its goals.

The two solutions will be detailed in Section 8.3, where the mathematical equivalence between the first of them and a pseudoinverse matrix solution will be proved as well.

## 8.2 Piecewise actuation approach

The force singularities are revealed by a sudden increase of some of the actuated joint generalized forces. This is principally caused by a local ill-conditioning of $J_{c_p}$: indeed, if we recall equation (7.28), the term to the right will go to infinity (because of $B_{pa}$) when $J_{c_p}$ becomes singular. In order to eliminate this problem when the parallel manipulator follows the desired trajectory, a coordinate partitioning into *active* and *passive* coordinates is performed, regardless of the independent/dependent coordinate partitioning scheme (see Section 6.2.2). The active and the passive coordinates correspond to the actuated and non-actuated joints, respectively, and are used in the process of reducing the inverse dynamics to the form (7.28) in order to find a solution for the actuator torques.

[1]that can easily be simplified to quasi-static cases

The active/passive partitioning is performed at a certain stage of the actuation strategy algorithm that still does not require redundant actuators. It is activated for configurations (time instants) of the followed trajectory, where changes in the actuator locations are necessary, so as to obtain an inverse dynamics solution, which is (locally) free of force singularity problems. The re-partitioning can be done on the basis of LU-factorization with column pivoting of the constraint Jacobian $J_c$ again, in order to obtain better conditioned $J_{c_p}$. In our approach, we add use a criterion for partitioning into active and passive coordinates that satisfies actuator torque performance limits and is oriented toward a minimal degree of redundant actuation, which seems reasonable since full degree is generally not required by the design specifications. The explanations follow.

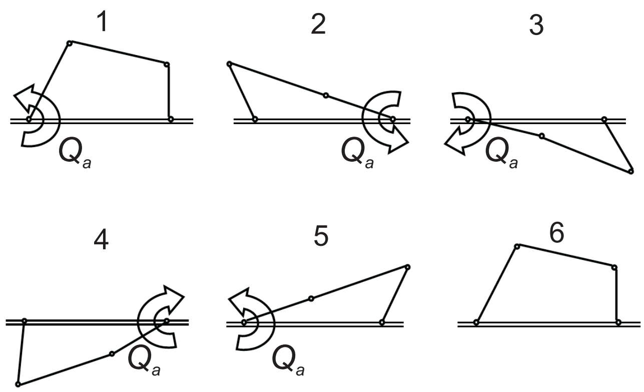



Figure 8.1: Example of piecewise actuation of a four-bar mechanism.

The proposed actuation approach is illustrated by means of the two test parallel manipulators, used throughout this work - a simple, four-bar mechanism (the example motion of which is schemed in Figure 8.1) and a 3-d.o.f. planar parallel manipulator (Figure 8.2).

In the actuation strategy, the $\{a, p\}$ re-partitioning that may lead to new actuator locations, is done on the basis of a particular procedure, represented in Figure 8.2. The inverse dynamics computation starts with a feasible actuation scheme (the filled in red actuated joints of Situation 1, to the left in the figure). The actuator locations are chosen to be compatible with the manipulator particularities in terms joints physically available for actuation (circled in blue) and the actuator performance limits. At every computation step that follows, a check on the actuator torque values is performed. If, due to a local ill-conditioning of $J_{c_p}$, at least one of the components of $Q_a$ reaches a predefined maximum (corresponding to the nominal torque limit of the actuators used), the inverse dynamics computation is temporarily hold (Situation 2 in the figure). An exhaustive sequence of subsets of joints is constructed from the set of generalized coordinates available for actuation, every subset representing a possible vector $q_a$. Among all the $\{a, p\}$ partitions resulting from the obtained subsets (all the possible vectors $q_a$), the one that satisfies the actuator torque limits and matches in the same time most closely the former $\{a, p\}$ used, is picked up (Situation 3 in the figure). As a result, a local robustness with respect to

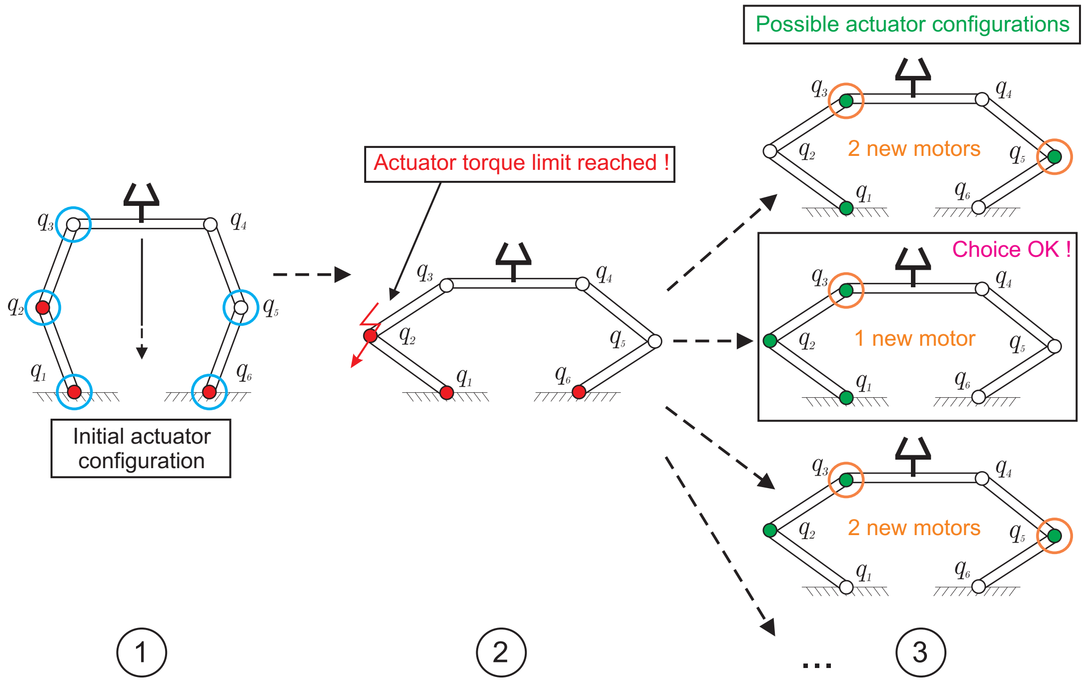



Figure 8.2: Illustration of the piecewise actuation strategy of minimal redundancy degree. At the detection of a torque limit trespassing, the new actuator locations (joints in green) are chosen among the available for actuation (circled in blue).

force singularity problems along with "maximal" preservation of the motor locations, are achieved.

The example of the four-bar mechanism motion in Figure 8.1 illustrates a similar actuation strategy result - the required trajectory is followed by activating four times the re-partitioning process for two of the four revolute joints: $q_1$ and $q_4$, taken as the available for actuation. Thus, the *piecewise* actuation scheme produces five consequtive vectors $q_a$, each of which corresponds in size to the single degree of freedom of the mechanism.

It is worth mentioning that in the examples above, cases of mass/inertia characteristics of the two multibody systems and internal joint friction torques were considered, for which the two parallel manipulators could eventually not follow the required trajectories without changes in the actuated joints, provoked by force singularity problems. In other words, for such cases the actuation strategy will inevitably *add* new actuators to the initial non-redundant actuator configuration. This will practically lead to redundant actuation, as explained hereunder. We shall see later, that depending on the trajectory and the manipulator specificities, non-redundant actuation schemes can be a possible strategy result as well. But if the trajectories to be followed reveal parallel (force) singularities, such schemes do not eliminate their negative effects.

We explained in Chapter 6 that the partitioning into active/passive joints is completely independent from the $\{u, v\}$ one applied for numeric stability purposes: some of the locally *independent* variables could be *passive*, for example. What is more, in case of redundant actuation the dimensions of the vectors $q_u$ and $q_a$ will be different, as the former always corresponds to the number of MBS degrees of freedom.

The above sequences of $\{a, p\}$ partitions, obtained from the detection of force singularities "along the way", corresponds to *virtual* changes of the actuator locations: the actuators can obviously not be "detached" from and "reattached" to the manipulator joints during motion! At this stage the solution for the inverse dynamics (7.28) is *non-redundant*, i.e. the actuators virtually pass from one joint to another, their number being equal to the number of manipulator degrees of freedom. Taking into account all the above considerations, as in [36] we shall refer to the successive $\{a, p\}$-reformulations of (7.28) during the motion as *piecewise* inverse dynamics of the parallel manipulator.

The actuator location virtual "switching" can be performed with respect to different criteria, according to the chosen model (static, quasi-static, dynamic). If different partitions $\{a_i, p_i\}$ occur during the piecewise motion over the prescribed trajectory, it is clear that actuators have to be *permanently* placed and operative on *all* of the joints that have participated in the successive vectors $q_{a_i}$ ensuring the force singularity elimination. This is a necessary condition not only because it is impossible to change actuator places during motion, but also because smoothness and continuity are desired for a normal manipulator operation and successful control: the actuators can not be instantly switched on and off. These conditions eventually lead to applying a *redundant* actuation to the manipulator.

## 8.3 Redundant actuation of parallel manipulators

The actuation strategy result depends on numerous factors: the manipulator topology, its mass and inertia characteristics, the presence of internal friction. It is based as well on the possible actuator locations and authorized number, entered as an input data in the form of joint index vector, when defining the manipulator and the task specifications. With respect to all this, the piecewise actuation strategy that we propose will produce different output:

- if the desired actuated joint index vector is of size, corresponding to the number of manipulator degrees of freedom, either a solution respecting the actuator torque limits will be computed, or the impossibility to compute such will be signaled, the latter meaning that other locations or/and higher torque performance motors have to be supplied to the algorithm;

- if the size of the actuated joint index vector is greater than the size of $q_a$, the strategy will deliver (depending on the user's option) a non-redundant solution (if such exists) or produce a minimal number of sequential partitions $\{a_i, p_i\}$, trying to keep as long as possible the actuators on their places. When actuator torque minimization is required, the piecewise actuation process can produce a redundant actuation scheme, because most probably different $\{a_i, p_i\}$ partitions will occur, meaning a resulting number of actuators, greater than the number of multibody degrees of freedom $d_{mbs}$. In other words, the lower the actuator torque limit values, the higher the chances to end up with a redundant actuation for the given trajectory;

If the piecewise actuation process produces a resulting number of actuators $s$, greater than $d_{mbs}$, i. e. a redundant actuation, then the system (7.28) becomes under-determined, as more unknowns (generalized joint torques) exist than equations of motion. This amounts to an $\{s - d_{mbs}\}$-infinity of solutions for the actuator torques, therefore a convenient technique to find a single solution to the problem must be applied.

Different approaches exist that can be referred to for this purpose, most of them based on optimization techniques. In this work we shall concentrate[2] on two optimal solutions for the under-determined inverse dynamics, depending on the optimization criterion:

1. When an overall minimization of the actuator torques is desired, a solution method, equivalent to the *pseudoinverse* solution technique, will be applied to complete the under-determined system and then solve it ordinarily. As we shall see in Chapter 9, this method will be successfully exploited in the control of the two benchmark manipulators – a four-bar mechanism and a planar parallel manipulator, when projections of the vector of non-redundant torque control outputs onto the redundant joint torque input space is necessary;

2. When compliance with the actuator performance limits is sought, a minimal infinity-norm torque solution will be computed, using a linear-programming technique.

### 8.3.1 Pseudoinverse-equivalent force redundancy solution

Let us consider a system of linear algebraic equations of the following form:

$$Px = y \tag{8.1}$$

in which the coefficient matrix $P$, the members of which are real numbers, is of dimension $(m \times n)$, the vector of unknowns $x$ and the right-hand side vector $y$ - of dimension $n$.

When $P$ is rectangular, i.e. $m \neq n$, we resort to under-determined $(m < n)$ or over-determined[3] $(m > n)$ systems of equations that are not solvable using classical linear algebra techniques. A commonly used solution in such cases is written as:

$$x = P^{+}y \tag{8.2}$$

where $P^{+} = P^{T}(PP^{T})^{-1}$ is called *pseudoinverse* matrix of $P$, provided that $rank(P) = m$. The pseudoinverse matrix, defined by Penrose [87], minimizes the Euclidian norm of $x$ ($\|x\|^2$, also known as "two-norm"), providing a solution of minimal error in a least squares

[2]Other solution methods can be envisaged, of course, but the two cited here are suitable and sufficient for the purposes of the present methodology.

[3]having less or more equations than unknowns, respectively.

sense. The pseudoinverse of any matrix is a unique matrix that must satisfy the following *Moore-Penrose* conditions:

$$PP^+P = P, \quad P^+PP^+ = P^+, \quad (PP^+)^T = PP^+, \quad (P^+P)^T = P^+P$$

The pseudoinverse method was logically the first to use for the cases of redundant actuation of this work, in which the under-determined actuator torque problem had to be solved. However, a solution that is strictly equivalent to the pseudoinverse one (as it was already demonstrated in [55]) was preferred afterward, because of its physical interpretation and its advantages in terms of computational efficiency and real-time implementation.

This solution consists in adding to the under-determined system of equations (7.28) a specific set of additional equations that will be extracted from the following system:

$$J_{c_a} Q_a = 0 \tag{8.3}$$

This system is derived from the partitioned form of $J_c Q = 0$:

$$[J_{c_a} \quad J_{c_p}][Q_a^T \quad Q_p^T]^T = 0 \quad \Rightarrow \quad J_{c_a} Q_a = -J_{c_p} Q_p \tag{8.4}$$

recalling that $Q_p = 0$ for cases of non-redundant ("normal") actuation.

The set (8.4) represents a mathematical "translation" of applying more torque to those joints that are locally better in terms of canceling force singularity problems. In other words, the joints that can locally be driven "easier" with respect to torques applied, are preferred as candidates for actuation, the "preference" being numerically evaluated via the ratio $B_{pa}$.

From the system (8.4), as many equations will be extracted and used to complete (7.28), as there are redundant actuators. Because of the fact that some (or all) of the formerly passive generalized coordinates will become active (actuated), this time the vector of active joint coordinates $q_a$ will consist of two new sub-vectors:

- $q_{a_d}$, the size of which equals that of the former $q_a$, used for non-redundant actuation cases, and corresponds to the number of degrees of freedom $d_{mbs}$ of the parallel manipulator;

- $q_{a_r}$, containing the redundant active coordinates (redundantly actuated joints). The size $r$ of this vector can be lesser than or equal (giving *partial* or *full* actuation redundancy, respectively) to the size of the former vector of passive generalized coordinates $q_p$. In case of partial actuation redundancy, there will be as well a new vector $q_p$ of size $N^{joint} - s$, in case of full redundancy a vector of passive joint coordinates will not exist.

Thus, in case of redundant actuation the dimension $s$ of the *new* vector $q_a$ is superior to the number of degrees of freedom $d_{mbs}$: $s = d_{mbs} + r$, and hence: $d_{AR} = r$ (see (4.1).

Applying this new partitioning $\{a_d, p_d\}$ (that corresponds to $a_d$) to the vector of actuator torques $Q$ and expressing $Q_{a_r}$ by analogy with (8.4) gives:

$$Q_{a_r} = B_{rd} Q_{a_d} \tag{8.5}$$

with $B_{pa} = -J_{c_p}^{-1} J_{c_a}$ and $B_{rd} \triangleq -\left\{J_{cp_d}^{-1}\right\}_r J_{ca_d}$, where the operation "$\{...\}_r$" signifies the construction of a sub-matrix by selecting the lines that correspond to the $r$ redundantly actuated joints. Apparently, in cases of fully redundant actuation (all passive joints actuated) we will have $B_{rd} \equiv B_{pa}$.

The set (8.5), added to (7.28), gives a complete, *determined* system of equations describing a solution of the redundant inverse dynamics of constrained MBS:

$$\begin{bmatrix} E^{(d_f \times d_f)} & B_{rd}^T \\ B_{rd} & -E^{(r \times r)} \end{bmatrix} \begin{bmatrix} Q_{a_d}^{(d_f \times 1)} \\ Q_{a_r}^{(r \times 1)} \end{bmatrix} = \begin{bmatrix} b_d \\ 0^{(r \times 1)} \end{bmatrix} \tag{8.6}$$

with $B_{pa_d} = -J_{cp_d}^{-1} J_{ca_d}$ and $b_d = \Phi_{a_d} + B_{pa_d}^T \Phi_{p_d}$ , $E$ and 0 denoting an identity matrix and a zero column vector of appropriate dimensions.

The solution for the actuator torques of the redundantly actuated closed-loop MBS is then straightforward:

$$\begin{bmatrix} Q_{a_d} \\ Q_{a_r} \end{bmatrix} = \begin{bmatrix} E^{(d_f \times d_f)} & B_{rd}^T \\ B_{rd} & -E^{(r \times r)} \end{bmatrix}^{-1} \begin{bmatrix} b_d \\ 0 \end{bmatrix} \tag{8.7}$$

offering a compact and computation time efficient (in terms of matrix operations, compared to the pseudoinverse solution) formulation of the redundant actuation problem. Let us now prove that this solution is mathematically strictly equivalent to a solution via pseudoinverse matrix.

We recall first that the constrained minimization problem over a vector $x$ obeying the $m$ independent constraint equations $Px = y$:

$$\min_{Px=y} \left(\frac{1}{2} x^T x\right) \tag{8.8}$$

where the matrix $P$ is rectangular, admits a solution, if the necessary condition for extrema:

$$x - P^T \eta = 0 \tag{8.9}$$

is fulfilled, provided the existence of a vector $\eta$, such that:

$$\frac{\partial}{\partial x}\left(\frac{1}{2} x^T x + \eta^T (y - Px)\right) = 0 \tag{8.10}$$

If we left-multiply the equation (8.9) by the matrix $P$, we can express $\eta$:

$$Px = PP^T \eta = y \Longrightarrow \eta = \left(PP^T\right)^{-1} y \tag{8.11}$$

and hence, substituting this expression into (8.9), we find that the solution to the minimization problem under constraints is:

$$x = P^+ y \tag{8.12}$$

where is the pseudoinverse matrix of $P$, provided that $rank(P) = m$.

As a next step, in order to prove the equivalence of the two solutions, we construct the pseudoinverse matrix for our redundancy problem, using as minimization constraint equations the upper half of the matrix to the left in system (8.6) and denoting $x = \left[Q_{a_d}^T \;\; Q_{a_r}^T\right]^T$, $P = \left[E^{(d_f \times d_f)} \;\; B_{rd}^T\right]$ and $y = b_d$:

$$P^+ = \underbrace{\begin{bmatrix} E^{(d_f \times d_f)} \\ B_{rd} \end{bmatrix}}_{P^T} \underbrace{\left[E^{(d_f \times d_f)} + B_{rd}^T B_{rd}\right]^{-1}}_{(PP^T)^{-1}} \Longrightarrow P^+ = \begin{bmatrix} \left[E^{(d_f \times d_f)} + B_{rd}^T B_{rd}\right]^{-1} \\ B_{rd}\left[E^{(d_f \times d_f)} + B_{rd}^T B_{rd}\right]^{-1} \end{bmatrix} \tag{8.13}$$

Now, if we use the underdetermined system (8.6) and express $Q_{a_d}$ as follows:

$$Q_{a_d} + B_{rd}^T B_{rd} Q_{a_d} = b_d \Longrightarrow Q_{a_d} = \left[E^{(d_f \times d_f)} + B_{rd}^T B_{rd}\right]^{-1} b_d \tag{8.14}$$

we can substitute this result back in (8.5), which gives:

$$Q_{a_r} = B_{rd}\left[E^{(d_f \times d_f)} + B_{rd}^T B_{rd}\right]^{-1} b_d \tag{8.15}$$

Finally, regrouping the actuator torques results in:

$$\begin{bmatrix} Q_{a_d} \\ Q_{a_r} \end{bmatrix} = \underbrace{\begin{bmatrix} \left[E^{(d_f \times d_f)} + B_{rd}^T B_{rd}\right]^{-1} \\ B_{rd}\left[E^{(d_f \times d_f)} + B_{rd}^T B_{rd}\right]^{-1} \end{bmatrix}}_{P'} b_d \tag{8.16}$$

proving that the two solutions $x = P^+ y$ and $x = P' y$ are equivalent.

### 8.3.2 Minimal infinity-norm force redundancy solution

The infinite-norm torque minimization solution, free of matrix inverse computations and used when better consistency with the actuator physical limits is targeted, is set up in this work according to an approach, pursued in [79] and extended here to cases of redundantly actuated parallel manipulators.

Compared to the two-norm torque minimization (pseudoinverse solution), the infinity-norm solution minimizes the largest component of a vector. In the field of robotics this type of minimization is also known as *minimum-effort* solution [88, 89, 90].

For redundantly actuated parallel manipulators it can be formulated as follows. Let us introduce a scalar variable $w \geq 0$, representing the value of $\|Q_a\|_\infty$. Then, according to [79], the infinity-norm $\|Q_a\|_\infty$ minimization problem can be rewritten as:

*minimize* $w$, *subject to:*

$$\begin{bmatrix} E & -e \\ -E & -e \end{bmatrix} \begin{bmatrix} Q_a \\ w \end{bmatrix} \leq 0 \tag{8.17}$$

*and*

$$Q_a = \Phi_a + B_{pa}^T \Phi_p \tag{8.18}$$

where $E \in \Re^{(s\times s)}$ denotes an identity matrix and $e \in \Re^{s}$ denotes an appropriately dimensioned column-vector of ones.

The latter optimization problem can be transformed into classical *linear programming* problem of the form [79]:

*minimize* $p^T x$, *subject to:*

$$\begin{array}{rcl} A_{eq}\, x & = & b_{eq} \\ A_{ineq}\, x & \leq & b_{ineq} \\ x_{min} \leq & x & \leq x_{max} \end{array} \tag{8.19}$$

where:

- $p = [0^{(1\times s)}\ 1]^T$, $x = [Q_a^T\ w]^T$;
- $A_{eq} = \begin{bmatrix} E^{(d_f\times d_f)} & B_{rd}^T \\ & 0^{(1\times s)} \end{bmatrix}$, $A_{ineq} = \begin{bmatrix} E^{(s\times s)} & -e^{(s\times 1)} \\ -E^{(s\times s)} & -e^{(s\times 1)} \end{bmatrix}$;
- $b_{eq} = [b_d^T\ \ 0]^T$, $b_{ineq} = 0^{(2s\times 1)}$; $x_{min} = [Q_{a_{min}}^T\ \ 0]^T$ and $x_{max} = [Q_{a_{max}}^T\ \ \infty]^T$.

Here $Q_{a_{min}}$ and $Q_{a_{max}}$ represent the lower and the upper boundary of the actuator torque, usually corresponding to the nominal torque values given in the motor technical specifications. For instance, for the DC-motors used in the parallel manipulator prototype discussed in this paper, $Q_{a_{min}} = -4.5\ Nm$ and $Q_{a_{max}} = 4.5\ Nm$.

The above optimization task can be found programmed in a dedicated MATLAB routine, called *linprog* and used in the inverse dynamics solution algorithms of this work.

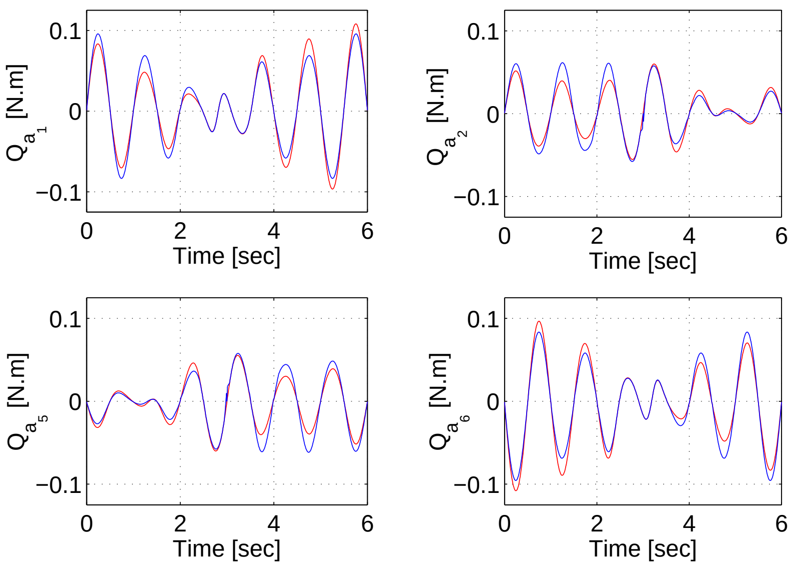



Figure 8.3: Actuator torques for a simple force redundancy ($d_{FR} = 1$) of the 3-d.o.f. planar parallel manipulator of Figure 7.5. Pseudoinverse solution in red, minimal infinite-norm solution in blue.

Two redundant actuator torque solutions, computed using the methods described above, are compared in Figure 8.3. They relate to one of the two closed-loop MBS, used to test the developments of this work: the 3-d.o.f. planar parallel manipulator of Figure 7.5. A slight decrease of the torque overall maxima (obtained for $Q_{a_1}$ around 5.75 sec and for $Q_{a_6}$ around 0.26 sec) and a consequent torque redistribution over the four actuators throughout the trajectory can be noticed for the infinite-norm torque minimization solution. As we shall see in Chapter 9, even for the case of simple redundant actuation ($d_{FR} = 1$) a significant decrease of the maximal torque values is observed when using one of these optimal torque solutions[4], compared to a non-redundant actuation.

## 8.4 Algorithmic computer implementation

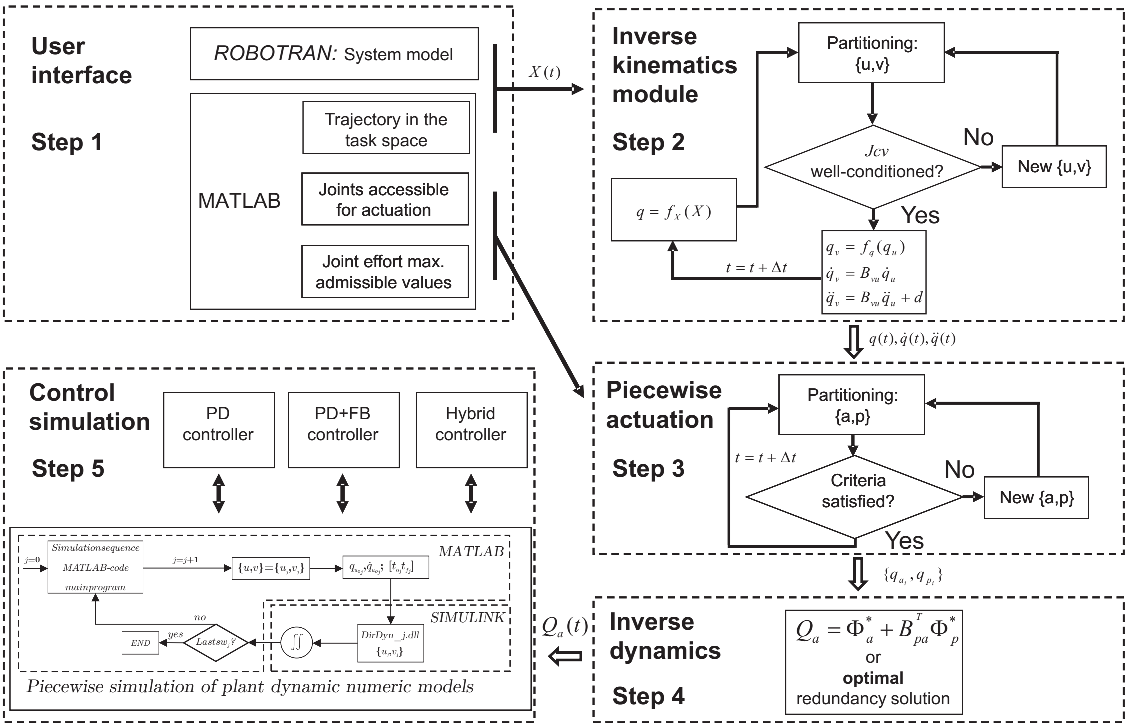



Figure 8.4: Algorithmic implementation of the proposed actuation strategy and actuator torque solution.

The different steps in the algorithmic implementation of the actuation strategy with subsequent actuator effort solution are illustrated in Figure 8.4:

- Step 1: User data acquisition – concerns the data (topology, characteristics, possible actuation configurations) on the parallel manipulator to be treated, and on the task (trajectory);

[4]the pseudoinverse one in our examples, giving a decrease by a factor of about 2 !

- Step 2: Inverse kinematics solution, based on the coordinate re-partitioning $\{u, v\}$ into *independent/dependent* generalized coordinates (see Section 6.2.2); Output: $q$, $\dot{q}$, $\ddot{q}$;

- Step 3: *Piecewise* actuation; Output: one or a series of sequential partitions into active/passive coordinates (actuated/non-actuated joints), defining the actuator locations for every successive *piece* (segment) of the trajectory (see Sections 6.2.2 and 7.5–7.6 for details);

- Step 4: Solution for the MBS inverse dynamics (pseudoinverse or infinity-norm torque minimization in cases of force redundancy as described in Sections 8.3.1–8.3.2); Output: actuator (joint) torques ($Q_a$);

- Step 5: Simulation of the controlled plant dynamics. A *master* MATLAB routine sequentially activates as many "slave" SIMULINK C-code plant dynamic models (generated beforehand using ROBOTRAN/MATLAB) as there are $\{u, v\}$ partitions (Section 7.7.1, Figure 7.7). The results ($Q_a$) from the previous step are used as a reference input in case of a feed-forward term in the tested virtual controllers.

Chapter 9 presents some basic control theory notions in order to explain the control algorithms used in our work, as well as results from simulations of the controlled plant dynamics. In Chapter 10 results from experimental validation of the tested control algorithms on a prototype will be given.

# Chapter 9

# Control of redundant parallel robots

## 9.1 Introduction: robot control

The control of redundantly actuated parallel manipulators is a vast domain of research and development in robotics. In the framework of this study, we shall not dedicate a numerous pages to this matter not only because it is not possible to encompass all the existing control algorithms and approaches in a single research thesis, but due as well to the fact that our principal goal is the development of strategies for redundant actuation, and not of new control algorithms. In other words, the matter of the present work is treated – in general terms – using control engineering tools, and not the control theory apparatus exclusively.

In robotics, the control tasks generally concern the computation of control input to apply to the robot actuators in order to have specific system behavior, e.g. a reaction to a variable external load applied to the end-effector, or certain tasks accomplished (trajectory plus end-effector output forces). The principal difficulty of the parallel manipulator control arises from the fact that it concerns multibody closed-loop, non-linear systems, in which all or many of the principal physical phenomena (dynamic terms, friction, joint and gear backlash) usually have to be considered. Moreover, nowadays the task specificities and their huge variety require very high end-effector velocities and precision along with system parameter robustness, very good dynamic performances and reliability.

According to the classics of the control theory that developed during the sixties and the 70ies, a control algorithm has to rely by default on position and velocity feedback, used to produce control outputs by comparing it to the actual system state, in order the corresponding real controller to be robust with respect to external perturbations and modeling errors. On the basis of this feedback, *proportional* (P), *proportional-derivative* (PD) and *proportional-integral-derivative* (PID) control schemes, commented in Section 9.1.2, are usually developed and can be found nowadays in the core of many modern control algorithms. Along with them, novel control approaches in terms of adaptive, model predictive and fuzzy logic control appeared in the last two decades, but we shall not comment on, nor make use of some of them, as this would go beyond the limits of this research. Detailed description and analysis of different control algorithms and classes, supported by numerous application examples can be found in [91, 92] and other literary issues.

As the final part of our work concentrates on the control of redundantly actuated parallel manipulators, let us briefly recall some of the developments in this domain of the robotic control, still relatively new.

### 9.1.1 Control of redundantly actuated parallel manipulators: research achievements

Increased appearance of control solutions for redundant parallel manipulators is observed over the past decade, as the principal interest in redundant parallel robots augmented during that period and more application fields are viewed nowadays.

One of the earliest works, treating the problems of robot control in the presence of singularities, was that of Nakamura and Hanafusa [93]. They proposed for the purposes of control stability a substitute to the pseudoinverse to solve the inverse kinematics in a singularity neighborhood, naming this substitute *SR-inverse* (from "singularity-robust"), which amounts to a least square solution. However, this solution inevitably leads to modifications of the end-effector trajectory in terms of velocity, because of the preference given to its feasibility over its exactness. In [94], Hanon compared methods for control of redundantly actuated parallel manipulators, drawing an important conclusion on their control depending on the number of actuators used – the more the actuators, the better the control. Moreover, he demonstrated the effectiveness of using optimization techniques to solve for the force redundancy. Another interesting control application we already mentioned in Chapter 4 is that of Lee et al. [75], in which they successfully reduce, by applying redundant actuation, the effects of shocks during parallel robot motion, comparing through simulation and experiments three torque distribution solutions: a minimum torque norm, a minimum torque rate and torque limit solution. In Chapter 4, we also discussed the works of Kock and Schumacher [42], and Yi and Freeman [41] and their contributions to the control of RAPM. In two other interesting and more recent research works [95, 96], Liu et al. and Cheng et al. make use of redundant actuation in order to eliminate undesired singularity effects in parallel manipulators, based on the developments of [69]. They experimented as well on kinematic and dynamic control methods for redundantly actuated parallel manipulators, thus verifying the efficiency of the proposed algorithms and proving the effectiveness of the Nakamura-based equivalent torque solution method, used for the purposes of control. Important and probably very perspective research results were achieved by Muller in [64, 97]. In these recent works, he took profit from a general solution for the inverse dynamics of redundantly actuated parallel manipulators to develop a computationally-efficient open-loop preload control scheme (depending on a single preload parameter) the simplicity of which makes it applicable in real-time control applications with backlash avoidance, especially for the cases of simply-redundant (i.e. one redundant actuator) parallel robots.

We end here this short state-of-the-art that does not cover by far all the developments in the domain of redundantly actuated parallel robot control, but many of them – like for instance the hybrid adaptive control scheme proposed in [98] – go beyond the scope of the present research.

## 9.1.2 Description of widespread robot controllers

### Classic PID-controller

The PID-controllers, named according to the *Proportional*, *Integral* and *Derivative* control actions they perform - are used in the vast majority of automatic process control applications in industry today. PID controllers are responsible for regulating flow, temperature, pressure, level, and other industrial process variables. The classical PID-control scheme of Figure 9.1 that can be found programmed in the controllers of many industrial robots, considers the manipulator dynamics to be linear and applies independent control inputs with constant gains to the joints that have to be controlled by the actuators. This type of control is simple to implement and computationally time-efficient, but usually suffers from bad precision at high velocity rates.

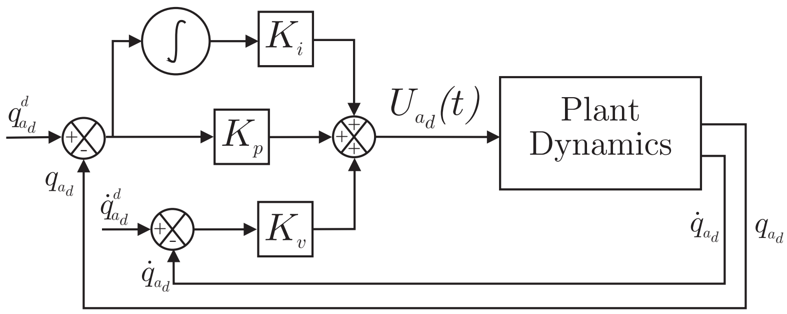



Figure 9.1: Classical PID-control scheme

At a time instant $t$, the control law $C(t)$ produces an output control signal $U_{a_d}(t)$ (in $[N.m]$) that acts on the actuated joints $q_{a_d}$, the number of which corresponds to the degrees of freedom of the multibody system[1]. The control signal is a sum of three terms that can be instantaneously considered as acting independently:

$$C(t) = U_{a_d}(t) = K_p\ e(t) + K_v\ \dot{e}(t) + K_i \int_{t_1}^{t_2} e(\tau)d\tau \tag{9.1}$$

where:

[1]As we shall see later in this Chapter, redundant control schemes can be considered successfully as well.

- $q_{a_d}$ and $\dot{q}_{a_d}$ are the current active generalized positions and velocities of the manipulator, the feedback on which is obtained by means of appropriate sensors;

- $q^d_{a_d}$ and $\dot{q}^d_{a_d}$ are the desired (reference) values for $q_{a_d}$ and $\dot{q}_{a_d}$, obtained by trajectory generation;

- $e(t)$ is the column-vector of tracking errors for $q_{a_d}$: $e(t) = q^d_{a_d} - q_{a_d}$;

- $\dot{e}(t)$ is the column-vector of time derivatives of the tracking errors: $\dot{e}(t) = \dot{q}^d_{a_d} - \dot{q}_{a_d}$;

- $K_p$ is the diagonal matrix of *error-proportional* gains $k_p$ of units $[N.m]$;

- $K_v$ is the diagonal matrix of *derivative-proportional* gains $k_v$ of units $[N.m.s]$. As generalized joint *velocities* $\dot{q}$ will be considered in this work in order to construct $\dot{e}(t)$ without time-deriving $q$, $K_v$ instead of the usual $K_d$ for "derivative" will be used;

- $K_i$ is the diagonal matrix of *integral-proportional* gains $k_i$ of units $[N.m/s]$.

Some of the three terms can be left out if they are not needed in the specific control design. Thus, it is possible to have a PI-, PD- or just a P-control (an ID-control scheme is generally not used).

If we consider proportional control alone ($K_v = K_i = 0$), the controller will react proportionally to *present* tracking errors – the larger their values, the larger the controller output. But the P-control will not be able to completely eliminate them, because for small $e(t)$ there will be small or insignificant control signal $U(t)$ generated. This major drawback of the P-controllers is often called static error, set-point (desired value) *drift* or *offset* (offset error). Very often in robotics the drift is caused by the forces of gravity.

In order to eliminate the offset, a term, proportional to the integral of the error must be considered. The integral control mode of a controller produces a long-term corrective change in controller output, driving the offset to zero. Very often, the following relation between the proportional and the integral gains is used:

$$K_i = \frac{K_p}{T_i} \tag{9.2}$$

where the constant $T_i$ is called *integral time.* The I-control term can be viewed as a reaction of the controller to past error values, accumulated through the time interval $[t_1\ t_2]$. The integral term tends to slow down the system responses to changes in its states. In order to speed up the system response, a derivative term must be used.

The D-control mode acts based on the rate of change of the error, hence its output is sometimes called *rate.* This control mode is very sensitive to measurement noises and makes tuning difficult if trial-and-error methods are applied. What is more, the bigger the derivative gains, the more important the risks of system high-frequency structural mode oscillations. Nevertheless, it can make a control loop respond faster and with less overshoot. In some sense, since the error "tendency" is monitored through its rate of change, the derivative control action can be viewed as an action to future error values.

Even if these predictive or anticipative capabilities are technically not true, the PID-control appears to provide more control action, and sooner than possible with P or PI-control, reducing the time it takes to return to the system set points.

### Computed-torque controller

When the applications demand rapidity and precision in the controlled behavior of the system (most often the case in robotics), more sophisticated control algorithms that take into account the system dynamics have to be addressed. The use of the so called *non-linear decoupling* control algorithms is very common and generally suits well this purpose [90]. This type of control is also known as *dynamic* control, due to the fact that it considers the system dynamics.

The dynamic control algorithms transform by means of state feedback the problem of non-linear system control into linear system control. In general, this is a complex problem, not always solvable. However, in robotics, finding control laws that decouple and linearize the system is simplified by the fact that the robot inverse dynamics model is usually available.

According to [90], if at time $t$ estimates of the reduced generalized mass matrix $\hat{\mathcal{M}}_r(q_{a_d})$ and the reduced vector $\hat{c}_r(q_{a_d}, \dot{q}_{a_d})$ can be computed and the joint generalized positions and velocities measured, then a control law of the following form can be chosen:

$$C(t) = \hat{\mathcal{M}}_r(q_{a_d})U_{a_d}(t) + \hat{c}_r \tag{9.3}$$

In case of a quasi-perfect dynamic model ($\hat{\mathcal{M}}_r(q_{a_d}) \approx \mathcal{M}_r(q_{a_d})$, $\hat{c}_r(q_{a_d}, \dot{q}_{a_d}) \approx c_r(q_{a_d}, \dot{q}_{a_d})$), using (7.21) we obtain the following set of equations for the closed-loop MBS direct dynamics:

$$\mathcal{M}_r(q_{a_d})\ddot{q}_{a_d} + c_r(\dot{q}_{a_d}, q_{a_d}) = C(t) = \hat{\mathcal{M}}_r(q_{a_d})U_{a_d}(t) + \hat{c}_r(\dot{q}_{a_d}, q_{a_d}) \Longrightarrow \ddot{q}_{a_d} = U_{a_d}(t) \tag{9.4}$$

which is in fact a set of $n$ (for an MBS constituted of $n$ rigid bodies) linear, independent second order differential equations.

Thus, the control signal vector $U_{a_d}(t)$ allows for obtaining by means of (9.3) the torques governing the MBS dynamics. Different forms can be envisaged for $U_{a_d}(t)$, but in this research work we shall limit them to a PD-controller scheme that we estimate as sufficient for our research purposes:

$$U_{a_d}(t) = K_p(q^d_{a_d} - q_{a_d}) + K_v(\dot{q}^d_{a_d} - \dot{q}_{a_d}) \tag{9.5}$$

in which the corresponding error-proportional gains $k_p$ and error derivative-proportional gains $k_v$ are of units $[s^{-2}]$ and $[s^{-1}]$, respectively.

Moreover, when the trajectory to be followed by the manipulator is entirely known in advance (e.g. as a result of a trajectory generation algorithm), the following control law can be considered:

$$U_{a_d}(t) = \ddot{q}^d_{a_d} + K_p(q^d_{a_d} - q_{a_d}) + K_v(\dot{q}^d_{a_d} - \dot{q}_{a_d}) \tag{9.6}$$

and, using the result (9.4) will give:

$$\ddot{q}_{a_d} = U_{a_d}(t) = \ddot{q}^d_{a_d} + K_p(q^d_{a_d} - q_{a_d}) + K_v(\dot{q}^d_{a_d} - \dot{q}_{a_d}) \tag{9.7}$$

Hence, the system of equations governing the controlled closed-loop MBS dynamics will become:

$$\ddot{e} + K_v\ \dot{e}(t) + K_p\ e(t) = 0 \tag{9.8}$$

representing a set of $n$ second-order decoupled differential equations for the tracking errors, compensating the system dynamics in the control law. This control scheme, depicted in Figure 9.2, is also known in the literature as *computed-torque* control.

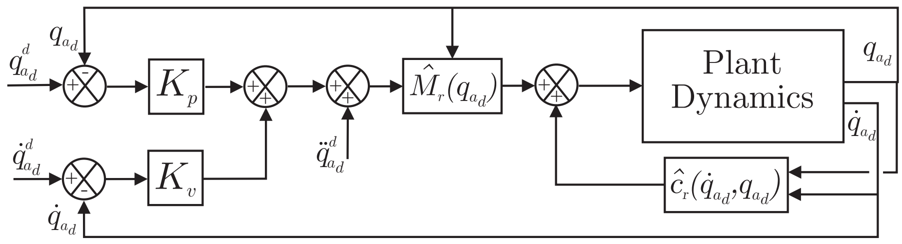



Figure 9.2: Computed-torque control scheme.

Compared to the PID control law (9.1), the computed-torque controller is characterized by position-independent gain values $k_{p_i}$ and $k_{v_i}$ ($\mathcal{M}_r(q_{a_d})$ and $c_r(\dot{q}_{a_d}, q_{a_d})$ are simplified in the equations 9.4). They can be specified separately for each controlled joint (due to the decoupling) in such a way that the damping of the system response is optimal and the error $e$ asymptotically tends to zero. In fact, the system transient response is described by the equations:

$$\ddot{q}_{a_d} + K_v\ \dot{q}_{a_d} + K_p\ q_{a_d} = 0 \tag{9.9}$$

obtained from 9.7, which can be viewed as a system of classical second-order damped oscillator equations of the form:

$$\ddot{x}_i + 2\zeta_i\omega_i\ \dot{x}_i + \omega_i^2\ x_i = 0 \tag{9.10}$$

Comparing (9.9) and (9.10), it is straightforward to write: $\omega_i = \sqrt{k_{p_i}}$, $\zeta_i = k_{v_i}/(2\sqrt{k_{p_i}})$. As it is well known from the mechanics theory, an optimal damping $\zeta_i$ is characterized by a value of $1/\sqrt{2}$. This value can be obtained when applying a computed-torque control, for instance choosing appropriately the gains $k_{p_i}$ (in order not to have resonances of $\omega_i$ with the natural frequencies of the manipulator) and computing $k_{v_i}$ from the equation $k_{v_i}/(2\sqrt{k_{p_i}}) = 1/\sqrt{2}$, i.e. $k_{v_i} = \sqrt{2k_{p_i}}$.

If the estimates $\hat{\mathcal{M}}_r$ and $\hat{c}_r$ are computed on the basis of the trajectory known beforehand, and not the measured values for $q_{a_d}$ and $\dot{q}_{a_d}$, the scheme transforms into a *feed-forward* (predictive) computed-torque control (Figure 9.3).

Since for a correct tracking $\hat{\mathcal{M}}_r(q^d_{a_d}) \equiv \mathcal{M}_r(q_{a_d})$ and $\hat{c}_r(q^d_{a_d}, \dot{q}^d_{a_d}) \equiv c_r(q_{a_d}, \dot{q}_{a_d})$, this new control algorithm will linearize and decouple the non-linear system dynamics as in

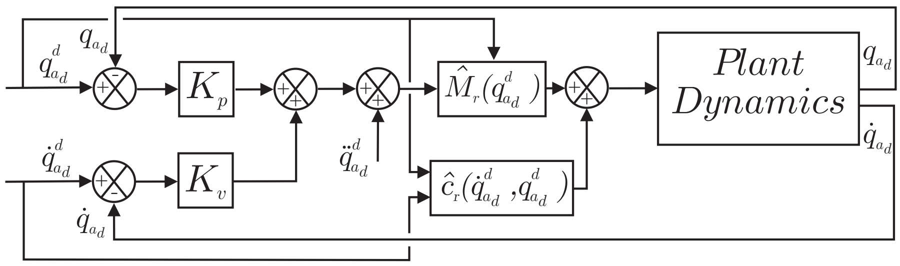



Figure 9.3: Feed-forward computed-torque control scheme.

the previous case, provided that the modeling errors are negligible.

The computation of the two presented computed-torque algorithms can be performed in real-time as a part of the system control process, on the basis of the Newton-Euler recursive (NER) algorithm, explained in Chapter 7 (Section 7.3). This procedure amounts in fact to computing the manipulator inverse dynamics, because the torques $C(t)$ that are applied as control inputs are calculated using the trajectory $\{q_{a_d}^d, \dot{q}_{a_d}^d, \ddot{q}_{a_d}^d\}$. Of course, if the dynamics is evaluated on-line, the numerical algorithm has to be efficient with respect to computational time (number of flops): this justifies the use of the NER formalism in relative coordinates.

A way of adding computational efficiency to the feed-forward control algorithm for prescribed trajectories is to compute the estimates in advance, store them in a database and directly inject their values during the control process – an approach that we actually use in the correspondent controller scheme for the applications of this study.

When the modeling errors can not be neglected, e.g. because the inertia parameters of the MBS are not completely known or joint friction forces are present, or a unknown external load is applied to the end-effector, the estimates for $\mathcal{M}_r(q_{a_d})$ and $c_r(q_{a_d}, \dot{q}_{a_d})$ do not match closely enough the real values. In such cases, according to (9.4) and (9.6), the closed-loop computed torque control schemes of Figures 9.2 and 9.3 are governed by tracking error equations of the following form [90]:

$$\mathcal{M}_r \ddot{q}_{a_d} + c_r = \hat{\mathcal{M}}_r(\ddot{q}_{a_d}^d + K_v\ \dot{e} + K_p\ e) + \hat{c}_r \tag{9.11}$$

then

$$\hat{\mathcal{M}}_r^{-1}\mathcal{M}_r \ddot{q}_{a_d} + \hat{\mathcal{M}}_r^{-1} c_r = \hat{\mathcal{M}}_r^{-1}\hat{\mathcal{M}}_r \ddot{q}_{a_d}^d + \hat{\mathcal{M}}_r^{-1}\hat{\mathcal{M}}_r(K_v\ \dot{e} + K_p\ e) + \hat{\mathcal{M}}_r^{-1}\hat{c}_r \tag{9.12}$$

and

$$\begin{aligned} -\hat{\mathcal{M}}_r^{-1}\hat{\mathcal{M}}_r \ddot{q}_{a_d}^d + \hat{\mathcal{M}}_r^{-1}\hat{\mathcal{M}}_r \ddot{q}_{a_d} &= \hat{\mathcal{M}}_r^{-1}\hat{\mathcal{M}}_r \ddot{q}_{a_d} - \hat{\mathcal{M}}_r^{-1}\mathcal{M}_r \ddot{q}_{a_d} + \\ &+ \hat{\mathcal{M}}_r^{-1}\hat{\mathcal{M}}_r(K_v\ \dot{e} + K_p\ e) + \hat{\mathcal{M}}_r^{-1}(\hat{c}_r - c_r) \end{aligned} \tag{9.13}$$

from which after simple transformations we obtain:

$$\ddot{e} + K_v\ \dot{e} + K_p\ e = \hat{\mathcal{M}}_r^{-1}\left[(\mathcal{M}_r - \hat{\mathcal{M}}_r)\ddot{q}_{a_d} + (c_r - \hat{c}_r)\right] \tag{9.14}$$

The latter indicates that the modeling errors constitute an excitation for the equations of the tracking errors. Therefore, the gain values must be increased with the increase of the model imprecision. As cited in [90], the robustness of the computed-torque controllers is sufficient and sometimes compensates for up to 80-90% of modeling errors.

If the manipulator motion is defined in the space of its end-effector coordinates $X$ (task space), control algorithms equivalent to the mentioned are applied following two principal approaches [90]:

- the task space trajectory is transformed into a joint space one, using the inverse kinematic relations of Section 6.2.3, and the control is performed in the joint space as described above, or
- the task space trajectory is directly used for a task space control of the manipulator on the basis of its task space dynamics.

### Hybrid position/force controller

The *hybrid* control algorithm is addressed in numerous literary issues, e.g. Raibert and Craig [99], Dombre and Khalil [90], Lipkin and Duffy [100]. It is extensively used for robot control applications, e.g. such in which force controlled compliant task frame motion must be performed [101, 102]. In our work, we test experimentally an analogue of such type of control algorithm, applying it to a redundantly actuated planar parallel manipulator, for which torque control is considered for the redundant actuator torques (treated as internal reaction torques), whereas joint position/velocity control is performed over subspace of joints, on which are applied the non-redundant torques.

The hybrid control can be viewed in some sense as an analogy to the dynamic control we already described, in which some part of the position and velocity tracking errors are replaced by force/torque errors. The actuators receive control inputs based on the simultaneous contributions of the two error types, contained in two reciprocal sub-spaces. For clarity purposes, we shall call the latter *motion sub-space* and *force sub-space*. The choice on which independent generalized coordinates should be considered for motion sub-space control, and which ones – for force sub-space control, generally depends on the manipulator specificities, the task requirements and the available sensor data.

A general hybrid control algorithm, in which force control is performed in the manipulator joint space[2], is shown in Figure 9.4. A translation can be performed if end-effector force control is targeted, using the manipulator relation $W = (J_m^T)^{-1} Q_{a_d}$, in which $W$ denotes an external *wrench* (resultant force plus resultant torque), supported by the manipulator end-effector, and the joint actuator torques $Q_{a_d}$ are computed using the manipulator inverse dynamics model.

The generalized coordinate distribution over the two sub-spaces is handled by means of a diagonal matrix $S_d$ of size $(d_{mbs} \times d_{mbs})$. Its elements $s_{d_i}$ are equal to 1 if the i-th

[2]This type of force sub-space control was initially assumed to be relevant to the applications considered in the present work.

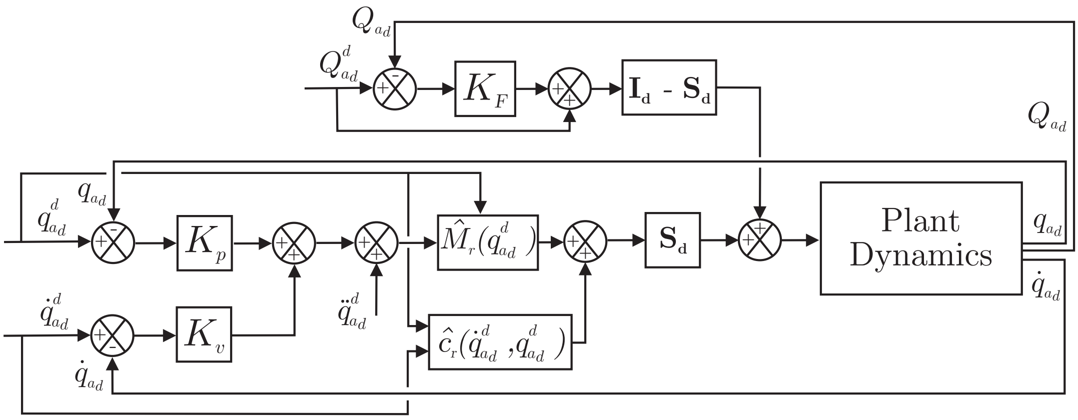



Figure 9.4: Hybrid control scheme (inspired from [90]).

actuated joint coordinate belongs to the motion sub-space, and to 0, if the corresponding to it joint force/torque is controlled.

In Figure 9.4, $I_d$ represents an identity matrix of size, equal to the size of $S_d$, $K_F$ - the matrix of force-proportional gains and $Q^d_{a_d}$ – the desired (reference) values of $Q_{a_d}$, computed using the manipulator inverse dynamics. The vector of control inputs for this control algorithm can be written as follows [90]:

$$C(t) = S_d\ [\hat{\mathcal{M}}_r(\ddot{q}^d_{a_d} + K_v\ \dot{e} + K_p\ e) + \hat{c}_r] + (I_d - S_d)\left[Q^d_{a_d} + K_F(Q^d_{a_d} - Q_{a_d})\right] \tag{9.15}$$

The motion sub-space control and the force sub-space control terms in the equations above are decoupled. This fact can easily be verified, as it is straightforward to write: $S_d = S_d^T = S_d^T S_d = S_d S_d^T$, $(I_d - S_d) = (I_d - S_d)^T = (I_d - S_d)^T(I_d - S_d) = (I_d - S_d)(I_d - S_d)^T$ and $S_d^T(I_d - S_d) = (I_d - S_d)^T S_d = 0^{(d_{mbs} \times d_{mbs})}$, where $0^{(d_{mbs} \times d_{mbs})}$ is a square matrix of zeros of dimension $0^{(d_{mbs} \times d_{mbs})}$. Thus, if for instance we multiply (9.15) to the left by the matrix $S_d^T$, we obtain the computed-torque control law (9.7) for the joints to be controlled in the motion sub-space, and if the left multiplication is performed using $(I_d - S_d)$, a force control[3] is "selected" to compensate the dynamics of the closed-loop MBS for the joints to be controlled in the force sub-space.

### Hybrid control in cases of redundant actuation of parallel manipulators

In this work, the hybrid control was initially supposed to be a possible alternative to the PD and PDFF-control schemes, when redundant actuation is applied to parallel manipulators. We were inspired from some research publications (e.g. [102, 104]) to use hybrid control in order to treat the additional, redundant control torques as "reaction forces" belonging to a *joint force* sub-space.

Following the reasoning of [104], for cases of hybrid control of redundantly actuated parallel manipulators a new control law, depicted in Figure 9.5, that acts on all the

[3] often called *active-stiffness* control in the literature [103] when performed in the end-effector coordinate space.

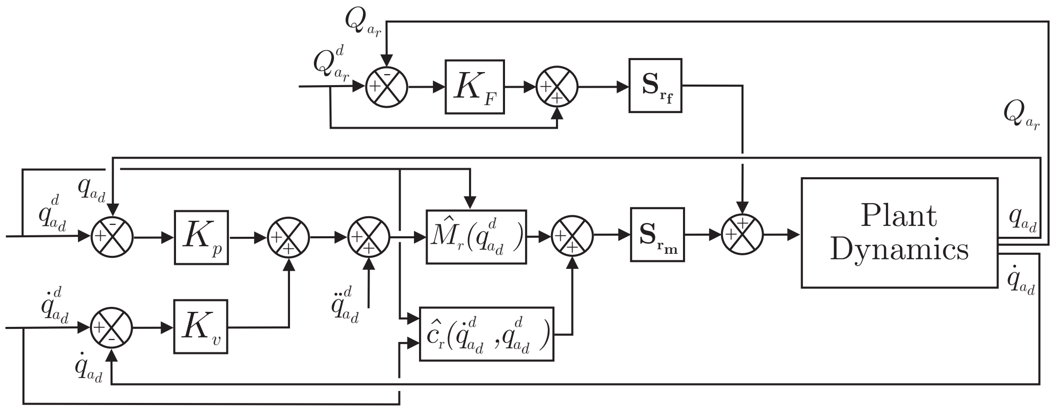



Figure 9.5: Hybrid control scheme for cases of redundant actuation.

actuated joints, can be written in the following form:

$$C_a(t) = S_{r_m}[\hat{\mathcal{M}}_r(\ddot{q}^d_{a_d} + K_v\ \dot{e} + K_p\ e) + \hat{c}_r] + S_{r_f}\left[Q^d_{a_r} + K_F(Q^d_{a_r} - Q_{a_r})\right] \quad (9.16)$$

where the rectangular matrices $S_{r_m}$ and $S_{r_f}$ are of size $(s \times d_{mbs})$ and $(s \times d_{FR})$, with $s$ being the total number of actuators (actuated joints), superior to the degrees of freedom $d_{mbs}$ (see Setion 8.3.1), and $d_{FR}$) the degree of force redundancy. Hence, the matrix $S_{r_m}$ selects the actuated joint coordinates to be controlled in the motion sub-space, and $S_{r_f}$ – the (redundant) forces/torques to be controlled in the joint force sub-space. The number of non-zero elements of $S_{r_m}$ (equal to 1), is strictly equivalent to $d_{mbs}$, whereas the number of those, equal to 1, in $S_{r_f}$ – to $(s - d_{mbs})$. In terms of example, let us consider a case of simple redundant actuation ($d_{FR} = 1$) of the 3-d.o.f. planar parallel manipulator of Figure 7.5, for which joints $q_1$ to $q_4$ are actuated. If for a given piece of the trajectory to be followed an active coordinate partition $q_{a_d} = [q_1\ q_2\ q_3]^T$ is picked up by the piecewise actuation strategy as the best locally (see the previous Chapter), then $Q_{a_r} = Q_{a_4}$ and the selection matrices for this piece of trajectory will be [104]:

$$S_{r_m} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad S_{r_f} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (9.17)$$

These matrices are such that $S^T_{r_m} = S^+_{r_m}$, $S^T_{r_f} = S^+_{r_f}$ and the two control terms (motion and force sub-space control) are decoupled again, i.e.: $S^T_{r_m} S_{r_f} = 0^{(d_{mbs} \times s)}$ and $S^T_{r_f} S_{r_m} = 0^{(s \times d_{mbs})}$. Indeed, if we recall that

$$\mathcal{M}_r(q_{a_d})\ddot{q}_{a_d} + c_r(\dot{q}_{a_d}, q_{a_d}) = \mathcal{Q}_r = C(t) = \hat{\mathcal{M}}_r(q^d_{a_d})(\ddot{q}^d_{a_d} + K_v\ \dot{e} + K_p\ e) + \hat{c}_r(\dot{q}_{a_d}, q_{a_d}) \quad (9.18)$$

and multiply for instance the control law (9.16) to the left by the matrix $S^T_{r_m}$, we obtain (as in the case of non-redundant hybrid control) the computed-torque control (9.7) to compensate for the closed-loop MBS dynamics in the motion sub-space:

$$S^T_{r_m} C_a(t) = C(t) = \hat{\mathcal{M}}_r(q^d_{a_d})(\ddot{q}^d_{a_d} + K_v\ \dot{e} + K_p\ e) + \hat{c}_r(\dot{q}_{a_d}, q_{a_d}) \quad (9.19)$$

and if the left multiplication is done using $S_{r_f}^T$, then the closed-loop dynamics for the redundant actuator efforts that are considered as reaction forces and belong to the force sub-space, is obtained from:

$$S_{r_f}^T S_{r_f} Q_{a_r} - S_{r_f}^T S_{r_m} [\hat{\mathcal{M}}_r (\ddot{q}_{a_d}^d + K_v \, \dot{e} + K_p \, e) + \hat{c}_r] = S_{r_f}^T C_a(t) = Q_{a_r}^d + K_F (Q_{a_r}^d - Q_{a_r}) \quad (9.20)$$

and hence:

$$Q_{a_r} = Q_{a_r}^d + K_F (Q_{a_r}^d - Q_{a_r}) \quad (9.21)$$

### 9.1.3 Controller tuning

For any practical control application, stability and non-oscillatory form of the plant response are sought, no matter what is the combination of functional conditions and desired set-points. These requirements are met by applying appropriate tuning techniques:

1. For controlling *linear* systems:
   - techniques based on classic control theory design methods, such as state-feedback, root-locus, Bode plot, Nyquist criterion, pole-placement;
   - techniques based on real or virtual (computer simulation) experiment-based methods, e.g. the well known Ziegler-Nichols method, Chein-Hrones-Reswick (modified Ziegler-Nichols) method, etc;
   - manual trial-and-error techniques.
2. For control of non-linear systems:
   - techniques of linear system control – when the non-linear system loop control is linearized by algorithms such as the system decoupling dynamic control described in the previous section,
   - techniques attempting to introduce auxiliary non-linear feedback in such a way that the system can be treated as linear: these are often called *feedback linearization*;
   - techniques treating the system as a linear one in a limited range of operation and using linear design techniques for each operational region, e.g. gain scheduling technique, adaptive control.

Let us give some examples of tuning of a PID-controller, applied on a linear system with one state variable[4].

[4]Also known as single-input single-output (SISO) system

### Manual tuning method

A wide-spread manual tuning method consists in the following. Firstly, the values of $K_i$ and $K_v$ are set to zero. Then, $K_p$ is increased until the output of the control loop starts oscillating – at that point the value of $K_p$ is considered as *critical* (often denoted $K_{p_c}$ or $K_c$). Then the "optimal" $K_p$ is set to be approximately half of $K_{pc}$ in order to obtain the so-called "quarter amplitude decay" type response[5]. $K_i$ is increased afterward until the offset error enters (in sufficient time) in the desired range for the process, paying attention to the fact that too high value of $K_i$ will cause system response instability. Finally, $K_v$ is increased until the response reaches its reference with an acceptable rapidity, after applying a load disturbance. However, too high $K_v$ will cause overshoot in the response. Nevertheless, a fast PID-controller usually overshoots slightly to reach the set-point more quickly. Some systems cannot accept overshoot, in which case a "critically damped" controller tune-up is required, with a $K_p$ significantly less than half of $K_{pc}$, so as to have a non-oscillatory system response.

As major drawbacks of the manual tuning methods could be cited the lack of precision and the presence of a great deal of subjective intuition. These methods rarely give satisfactory results in cases of complex controller structures and/or systems. Better tuning is obtained using methods like the ones described hereunder.

### Ziegler-Nichols method

The Ziegler-Nichols and Chein-Hrones-Reswick are tunning methods, largely used in practical situations, mostly when off-line (i.e. not acting in a control process) experiments can be performed on the controller. These methods use the information obtained from a system response to an external *step input* or *frequency response* tests.

The Ziegler-Nichols method is the earliest design (tuning) method for PID-controllers, developed in 1942 by John Ziegler and Nathaniel Nichols, and received some light modifications (Chein-Hrones-Reswick method) later. As in the method above, the $K_i$ and $K_v$ gains are first set to zero. The $K_p$ gain is increased until it reaches its critical value $K_c$. The latter and its corresponding oscillation period $P_c$ of the system response are used to set the gains as shown in the following table:

| Controller type | $K_p$ | $K_i$ | $K_v$ |
|---|---|---|---|
| P | $0.5K_c$ | – | – |
| PI | $0.45K_c$ | $1.2K_p/P_c$ | – |
| PID | $0.6K_c$ | $2K_p/P_c$ | $K_pP_c/8$ |

Table 9.1: Values for the PID-controller gains using Ziegler-Nichols tuning method

The gain values in Table 9.1 are obtained by experimental analysis of many different

[5]A system response, the amplitude of oscillation of which decreases by 1/4 over every half oscillation period, starting from the initial amplitude peak value.

systems, using models constructed by means of heuristic rules that are based upon step and frequency responses of the controlled plants. The design criteria ensure that the amplitude of the closed-loop oscillation decays at a rate of 1/4, which for many practical cases results in a insufficient oscillation damping.

Other types of tuning techniques, e.g. based on *optimization* methods (see [105] for instance), *genetic algorithms* [106], etc. exist. In this research work, we do not consider them, but a particular technique of tuning, based on an exhaustive control simulation procedure (see Section 9.3 for details). Therefore, we shall limit our discussion to the techniques commented above.

## 9.2 Envisaged control schemes

Three control algorithms were tested via computer simulations of the motion of a four-bar mechanism and the planar 3-d.o.f. parallel manipulator (Figure 7.5): a classic PD-control (eq. (9.1), $K_i = 0$), a feed-forward (predictive) computed-torque control (Figure 9.3), and a PD-control containing a torque feed-forward term. A schematic representation of the third control law is given in Figure 9.6. The torque feed-forward term consists in injecting the vector of desired (pre-computed) values for the actuator torques, multiplied by a selection matrix $S_t$. The latter is either a unitary matrix, or a zero square matrix of size, equal to the number of actuated joints. Its role is to transform the PD-control law into a PD-control containing feed-forward terms, and vice versa, thus making easier the computer implementation of the control algorithms.

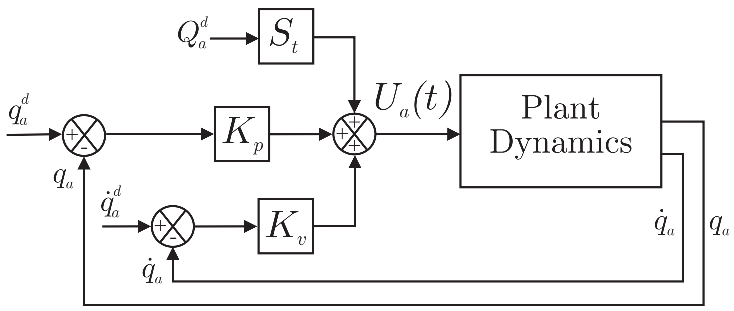



Figure 9.6: PD-controller containing direct torque feed-forward terms. $Q_a^d$ - precomputed reference (desired) values for the actuator forces/torques.

Let us precise that from now on and for the sake of simplicity, we shall use the following abbreviations: "PD-control", "PDFF-control" for PD-control including torque feed-forward term, and "FFCT-control" – to denote feed-forward computed-torque control algorithm.

For the simulations, as well as for the experimental validation that will be presented in Chapter 10, example trajectories containing singularities were generated for the two systems.

Simulations of the two controlled plant dynamics were performed both for non-redundant and redundant actuation cases, in order to not only test the control algorithm validity on trajectories containing force singularities, but to draw as well general conclusion on the redundant actuation advantages in terms of manipulator control.

To clarify the particularities in the structure of the tested control algorithms, important points have to be highlighted for each one of them, as follows.

### 9.2.1 PD-control

Classic PD-control algorithm of the form (9.5) was used to serve mostly as a means of comparison with the other control algorithm tested. For the cases of redundant actuation of the two manipulators, the control was performed over the space of all $s$ actuated joints $q_a$ ($s > d_{mbs}$), following the control design considerations of Ghorbel [107], Liu et al. [95] and Cheng et al. [96].

In [107], Ghorbel draws some important conclusions on the *direct applicability* of classic control algorithms, such as PD control and PD/gravity-compensation control – valid for serial manipulators, to closed-loop manipulator topologies, proving that they work as well for the latter and lead to asymptotic stability, if properly tuned.

The control design procedure of [95, 96], based on the Nakamura equivalent torque method (proposed in [93]), develops a controller for a given redundantly actuated parallel manipulator, considering equivalent control for the open-loop structure of its serial leg chains, when "opening" the closed-loop manipulator topology by cutting out the moving platform through the anchor joints. The equivalent controllers must act in parallel, in such a way that the anchor points follow the same prescribed trajectory as when they are part of the closed kinematic loops.

By analogy with the Nakamura's equivalent torque method, transcribing the PD-control law (9.5) for cases of redundant actuation of parallel manipulators will give:

$$U_a(t) = K_v \dot{e}_r + K_p e_r \tag{9.22}$$

where $e_r = q_a^d - q_a$, with $q_a = [q_{a_d}^T \quad q_{a_r}^T]^T$ and $U_a = [U_{a_d}^T \quad U_{a_r}^T]^T$ – corresponding to the whole set of active (non-redundant and redundant) generalized coordinates.

### 9.2.2 PDFF-control

The equivalent control law in case of PD/Feed-forward algorithm, acting on a redundant number of active coordinates, is written analogically as:

$$U_a(t) = K_v \dot{e}_r + K_p e_r + Q_a^d \tag{9.23}$$

with direct feed-forward term $Q_a^d$ computed in advance by (8.7) and corresponding to $Q_a = [Q_{a_d}^T \; Q_{a_r}^T]^T$.

### 9.2.3 FFCT-control

In order to test via simulation the predictive computed-torque control algorithm in cases of redundant actuation, a projection of the non-redundant control inputs onto the redundant-actuation space is performed in the following manner (similar to the indirect method of Nakamura used in [95, 96]):

$$U_a(t) = A^+ \left\{ \mathcal{M}_r(\ddot{q}_{a_d})(q^d_{a_d} + K_v\dot{e} + K_p e) + c^*_r \right\} \tag{9.24}$$

where the left-hand side corresponds to the left-hand side of eq. (7.34), $e = q^d_{a_d} - q_{a_d}$ the superscript "$d$" standing for *desired* (reference) values, and $A^+ = A_p^T(A_pA_p^T)^{-1}$ is the *pseudoinverse* of $A_p$ of (7.27).

### 9.2.4 Tuning of the considered controllers

The Ziegler-Nichols method was firstly used for the cases of PD and PDFF-control. The critical gain $K_c$ and period $P_c$, obtained by following the rules of this tuning method, were evaluated via simulations of the controlled plant response. Then the corresponding values for $K_p$ and $K_v$ were calculated using the formulae presented in Table 9.1.

The "tuning", tried for the feed-forward computed torque control, was performed with respect to finding the gains of the matrix $K_p$, using the manual tuning procedure already explained in Section 9.1.3. The values of the diagonal of the matrix $K_v$ were computed afterward, using the formula $k_{v_i} = \sqrt{2k_{p_i}}$ that we already mentioned in Section 9.1.2, in order to have optimal damping of the closed-loop system response.

As a second tuning method, an exhaustive control simulation procedure that performs multiple repetitive simulations for different combinations of values of the controller gains over one and the same trajectory, was developed by the author. At the end of each subsequent simulation run within this procedure, carried out for controller gain values situated in pre-defined ranges[6], an estimate for a specific integral controller performance criterion $I_{pc}$ is calculated and stored in a data table along with the gain value combination. The performance criterion is computed by means of the following formula:

$$I_{pc} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left[(W_a\epsilon_{a_i})^2 + (W_v\epsilon_{v_i})^2 + (W_p\epsilon_{p_i})^2 + (W_t\epsilon_{t_i})^2\right]} \tag{9.25}$$

where:

- $\epsilon_{a_i}$ is the global acceleration error for all the controlled joints at the i-th simulation time step - a sum of the absolute values of the elements of vector $\ddot{e}_i$ or $\ddot{e}_{r_i}$ (depending on the actuation case: non-redundant or redundant) in $[rad/s^2]$;
- $\epsilon_{v_i}$ is the global velocity error for all the controlled joints at the i-th simulation time step - a sum of the absolute values of the elements of vector $\dot{e}_i$ or $\dot{e}_{r_i}$ in $[rad/s]$;

[6]the corresponding limits of which are chosen so as to have stable closed-loop control response.

- $\epsilon_{p_i}$ is the global position error for all the controlled joints at the i-th simulation time step - a sum of the absolute values of the elements of vector $e_i$ or $e_{r_i}$ in $[rad]$;
- $\epsilon_{t_i}$ is the global joint (actuator) torque error for all the controlled joints at the i-th simulation time step - a sum of the absolute values of the elements of vector $e_t = Q^d_{a_d} - U_{a_d}$ or $e_{t_r} = Q^d_a - U_a$ in $[Nm]$;
- $W_a$ is the weight of the global acceleration error in the overall error $I_{pc}$, in $[s^2]$;
- $W_v$ is the weight of the global velocity error in the overall error $I_{pc}$, in $[s]$;
- $W_p$ is the weight of the global position error in the overall error $I_{pc}$, dimensionless;
- $W_t$ is the weight of the global joint torque error in the overall error $I_{pc}$, in $[1/Nm]$;
- $N$ is the total number of time-steps completed by the simulation process.

As a final result of the exhaustive control simulation procedure, the combination of gains that corresponds to the best controller performance score (the minimal $I_{pc}$), is reported.

This procedure was preferred by the author and extensively used for the control simulations that followed, because it is characterized by higher precision, compared to experimental tuning methods (manual, Ziegler-Nichols, etc.), and takes profit from the effectively generated and numerically stable manipulator dynamic models. Even if for the feed-forward computed-torque controllers no real tuning is necessary, because the closed-loop control of the MBS is decoupled and the gains – position-independent, this type of control was "tuned" via exhaustive simulations as well, in order to have a confirmation of its advantages, and verify if the "tuning" results produce gain coefficients that are coherent with the optimal damping condition $k_{v_i} = \sqrt{2k_{p_i}}$.

A detailed description of the control simulation tuning procedure, including its schematic representation, follows in the next section.

## 9.3 Computer implementation

The computer implementation of the discussed control algorithms, tested on for cases of non-redundant and redundant actuation of the two closed-loop MBS we mentioned before, is realized in the MATLAB/SIMULINK environment, on the basis of the symbolic modeling and robust time integration approach detailed in Section 7.7.1.

Thanks to the computation efficiency of the symbolic direct dynamic models, every simulation of the controlled plant is carried out faster than the trajectory (real) time. This fact allows for realizing the exhaustive simulation procedure, mentioned in the previous section and depicted in Figure 9.7, in reasonable amounts of time.

Every simulation within the exhaustive control simulation loop is performed with ranges of values[7] for $K_p$ and $K_v$, varying with increments $\Delta K_p$, $\Delta K_v$, which induce the embedded algorithmic loops (shown in Figure 9.7). In the core of every loop lies the control

[7] Gain values, for which the plant dynamics control is unstable, are rejected.

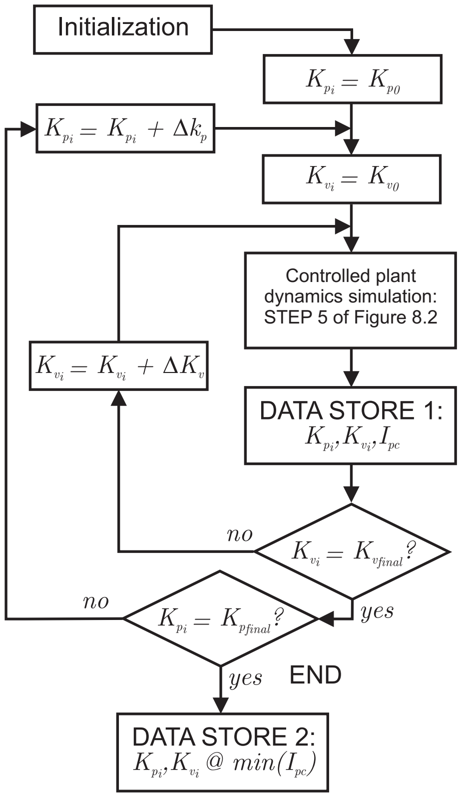



Figure 9.7: Block schematics of the exhaustive controller tuning simulation procedure for the case of PD controller with direct torque feed-forward (gain $K_t$).

simulation algorithm of Figure 7.7 that includes the sequential calls of as many pre-generated SIMULINK models as there are $\{u, v\}$-partitions. After each simulation, the values of the gain and the corresponding value of the integral performance index $I_{pc}$ are stored in a MATLAB data structure (DATA STORE 2 in Figure 9.7). All weights in the expression (9.25) are set to 1, so that neither of the different errors is privileged. Of course, other combinations of weight factor values could be considered in practice, e.g. $W_p = 1, W_v = 1, W_a = 0, W_t = 0$, as for many robotic applications of motion tracking only the position and the velocity error are of interest. Several hundreds of simulations were carried out during every run of the global simulation procedure. Two main results are obtained after each exhaustive simulation run:

- An error surface (corresponding to the data, recorded in the Data store 1 in Figure 9.7), each point of which represents a value of the controller performance index $I_{pc}$ for a given set of gain values. An example of such an error surface is given in Figure 9.8, representing $I_{pc}$ as a function of the gains $K_p$ and $K_v$, for the case of non-redundant PD-control of the four-bar mechanism.

- An "optimal tuning" set of gain values, for which the minimum for $I_{pc}$ is obtained.

Figure 9.8 illustrates well another fact: the use of optimization methods, which can be envisaged for such kind of tasks, is often complicated by the existence of numerous local

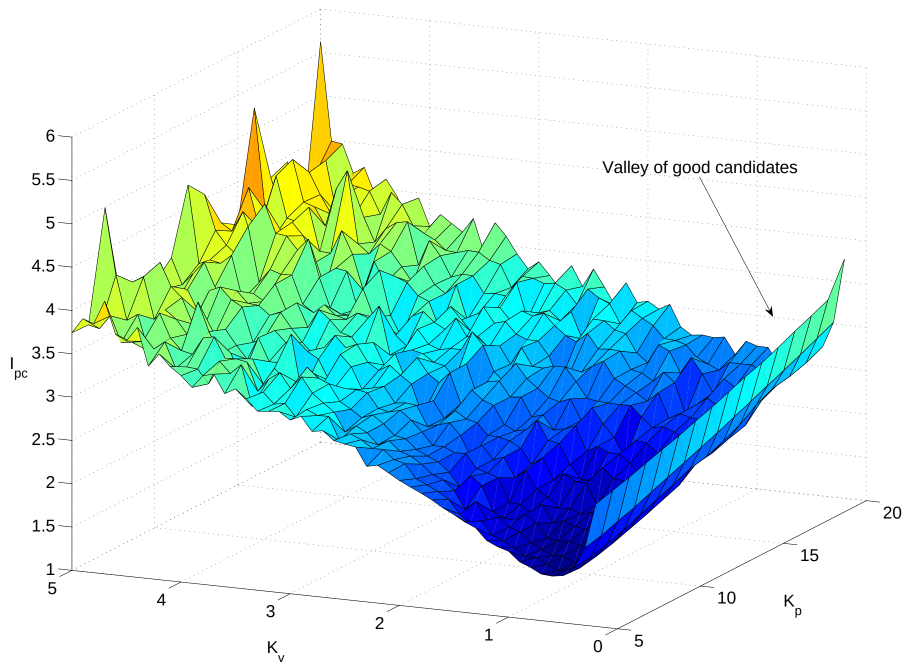



Figure 9.8: Plot of the $I_{pc}$ surface for the case of four-bar mechanism non-redundant PD-control.

minima in the explored cost-function surface. In our case this problem has (supposedly) a numerical nature: in the figure it can be noticed by the increasing "roughness" of the $I_{pc}$ surface for high $K_v$ values. We thus prefer to resort to an "exhaustive" search, which was made possible thanks to the size of the system and the limited number of parameters.

In the sections that follow, control simulation results for the two benchmark systems used in this work will be presented by: firstly, describing their topology models and test simulation conditions, secondly, demonstrating the advantages of the redundant actuation with respect to control problems when overcoming force singularities, and, finally, giving a brief comparison of successful control using the tested control schemes.

## 9.4 Four-bar mechanism: control simulations

### 9.4.1 Multibody model and test description

For the both MBS we treated, modeling in relative coordinates was performed based on the multibody formalisms we reviewed in Chapter 5. The kinematic loops were opened using the body-cut procedure, described in Section 6.2.1 and depicted in Figures 6.4a and 6.5a. This type of kinematic loop cut was chosen to allow for including in the models all constitutive bodies (with their mass/inertia parameters) and all joints of the mechanisms[8], thus making the latter available for partitioning and virtual actuation purposes.

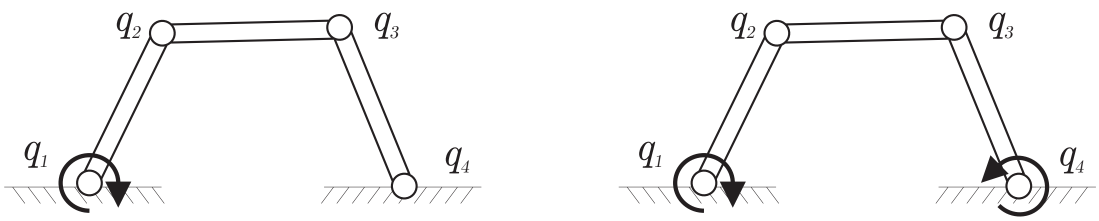



Figure 9.9: Topology representation of the four-bar mechanism. Left: case of non-redundant actuation (joint $q_1$ actuated), right: case of simple redundant actuation (joints $q_1$ and $q_4$ actuated).

The multibody topology of the four-bar mechanism is schemed in Figure 9.9. It consists in three bars of identical length and mass, and four revolute joints, interconnecting them. A loop cut through the base body was performed, in order to restore a tree-like topology. In case of non-redundant actuation control, only the joint $q_1$ is actuated, whereas in case of redundant actuation control, a simple force redundancy (the number of actuators surpasses the number of d.o.f. by one) is chosen, the actuated joints being $q_1$ and $q_4$.

The trajectory, used for the simulations of the four-bar, is similar to that depicted in Figure 8.1 and is shown here in Figure 9.10 by a "snap-shot" sequence of animation images, taken from the ROBOTRAN animation module.

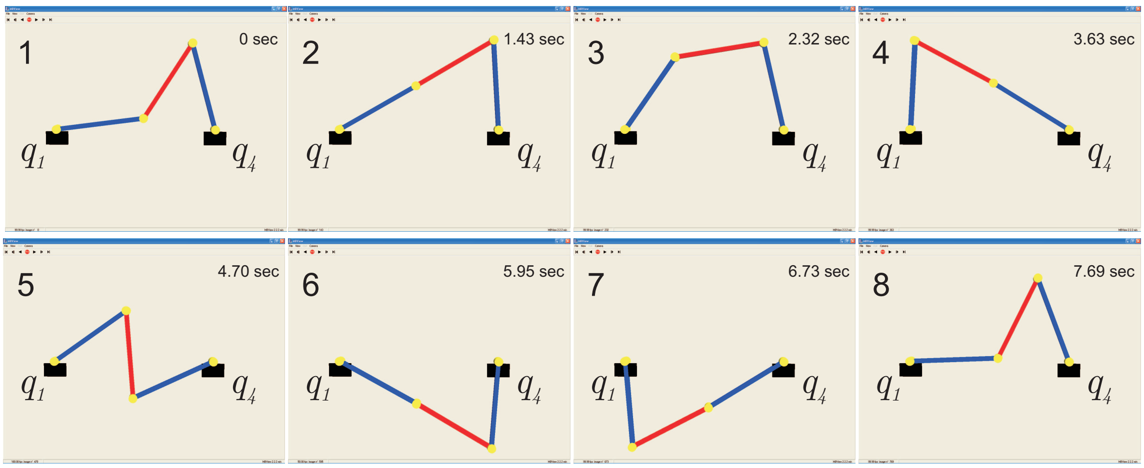



Figure 9.10: Trajectory of the studied four-bar mechanism.

The trajectory reveals four force singularities when actuating only *one* of the joints $q_1$ and $q_4$ for every subsequent "piece" of trajectory path: two singularities with respect to actuating $q_1$ (images 4 and 7 in the figure) and two "mirror" singular configurations (images 2 and 6) with respect to $q_4$.

[8]Considering a massless rod loop cut for the intermediate bar of the four-bar mechanism would make $q_2$ and $q_3$ disappear from the model, for instance.

According to the prototype, designed for experimental validation purposes (see Chapter 10), three identical bars of 0.2m length and 0.15kg mass were used in the model and dry friction with a approximate friction coefficient in the bearings (evaluated experimentally on the designed prototype) of 0.015 was considered for the four revolute joints. Viscous friction forces were not taken into account.

In order to compare the tested controllers, the exhaustive simulation tuning procedure was performed using two different problem cases: a non-perturbed trajectory case and a double perturbation case. The latter comprises an initial condition deviation of about 15%, plus a sudden (at $t = 4sec$) impulse perturbation acting on the angular displacements and velocities.

The actuator torque values for the feed-forward term, employed in the redundant PDFF-controller, are computed using the pseudoinverse-equivalent solution (see Section 8.3) – one of the two torque minimization solutions we described in Chapter 8.

The controller efficiency is evaluated both in terms of *settling time*[9] $T_{s_p}$ for the position response, and $I_{pc}$ value. The numerical results of the exhaustive control simulation tuning procedure are regrouped in Table 9.2, presented in Section 9.4.3, followed by a discussion.

Let us continue by commenting the results from simulations of the controlled four-bar mechanism, starting from a case, for which the advantage of applying redundant actuation is evident.

### 9.4.2 Four-bar mechanism: overcoming force singularities

Demonstrating the advantages of the redundant actuation of parallel manipulators is a key point of the present research work. Therefore, examples of a PDFF-control of the two closed-loop multibody systems, for which the corresponding reference trajectories are correctly followed only if redundant actuation is applied, are proposed here and in Section 9.5.2. The PDFF-controller, used in both cases, was tuned by means of the exhaustive procedure, described earlier. The gain values correspond to that of Tables 9.2 and 9.3, which will be commented in the next section.

Let us consider the simulations of a non-redundant ($q_1$ actuated) and a redundant ($q_1$ and $q_4$ actuated) PDFF-control of the four-bar mechanism for a specific combination of joint dry friction and damping forces. In our simulations, along with the dry friction coefficient of 0.015, additional joint damping terms with damping coefficient of 0.0039 Nm.s/rad are applied in order to recreate this practical situation of robots, some characteristics of which (e.g. joint friction components) change in time, and observe its effect on the non-redundant and redundant control.

Sequences of snapshots, similar to those, taken from the animation of the complete four-bar motion simulation, are given in Figures 9.11 and 9.12 as a means of visualizing

[9]Measured as the amount of time in which the system response enters for the last time in the +/- 5% margin of the reference.

the virtual system behavior for the two cases – non-redundant and redundant actuation control.

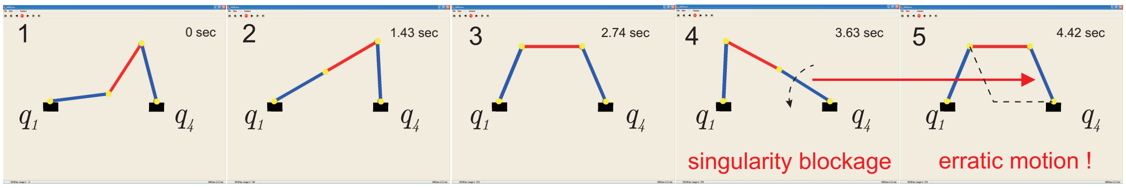



Figure 9.11: Snapshot sequence of the four-bar non-redundant PDFF-control in the presence of joint friction and damping torques: singularity blockage and subsequent bad tracking.

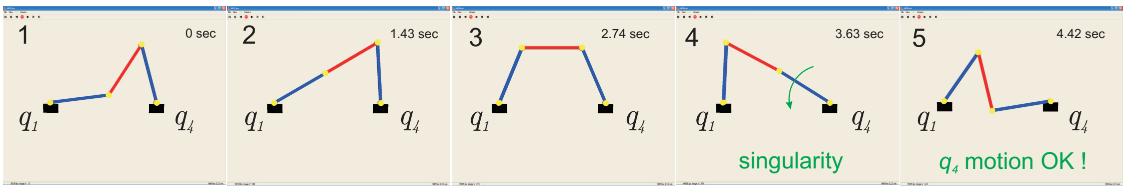



Figure 9.12: Snapshot sequence of the four-bar redundant PDFF-control in the presence of joint friction and damping torques: singularity effect elimination and smooth passage.

When only the joint $q_1$ is controlled by the PDFF algorithm, even with the contribution of the direct torque feed-forward term, the mechanism (as expected) does not succeed in following the prescribed trajectory. It is instantly blocked at the singular configuration of snapshot 4 (at 3.63 sec), then continues moving in a way, completely different from the desired one. The erroneous motion can be noticed as well by the graphs in Figure 9.13, representing the same (as in the snapshots) joint coordinate time evolution, and torque time evolution for joints $q_1$ and $q_4$. As seen in Figure 9.13, left, the force singularities cause important discontinuities in the non-redundant torque, applied to joint $q_1$.

In contrast to this situation, when both $q_1$ and $q_4$ are actuated via control torque inputs, the four-bar precisely follows the prescribed trajectory and the problems caused by the force singularities are completely eliminated. Moreover, with the chosen control scheme, the both input control torques $Q_1$ and $Q_4$ (see for instance $Q_1$ shown in Figure 9.13, right) are smooth and of lower amplitudes than the non-redundant one.

### 9.4.3 Comparison of the simulated control algorithms

The following table regroups the "optimal" gains of the tested control schemes, obtained by exhaustive control simulation loops for two trajectory cases (non-perturbed and with two perturbations – an initial one and a second one at $t = 4s$), as well as the controlled plant response settling times and integral errors, obtained by simulating the perturbed trajectory tracking. The results from two different tunings were tested on the same, perturbed trajectory in order to emphasize the reaction rapidity (settling time) and robustness of

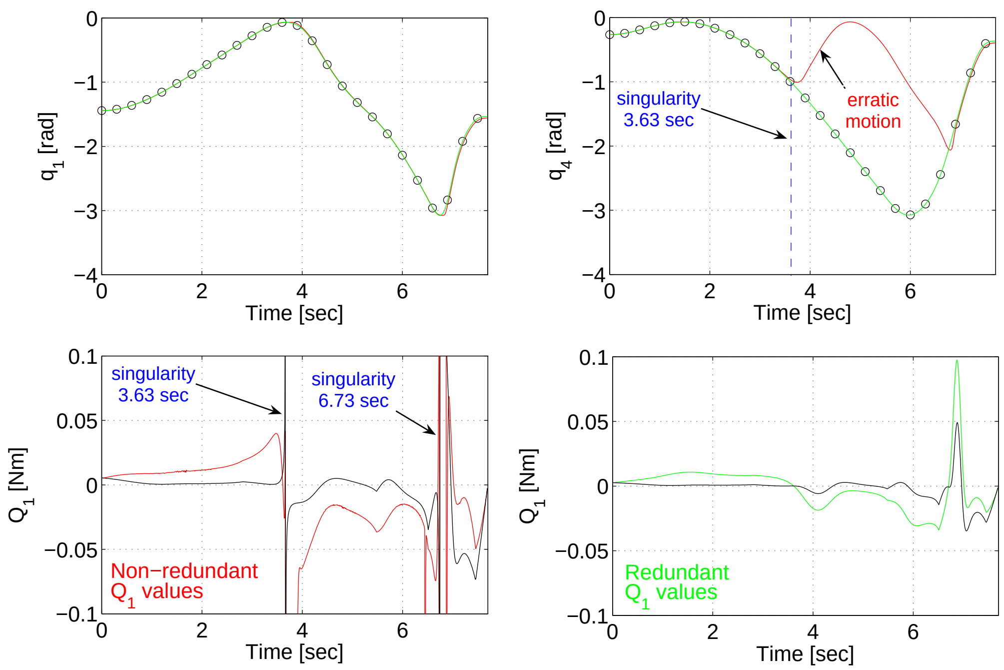



Figure 9.13: Four-bar PDFF-control in the presence of joint friction and damping torques: joint angle and torque responses. Non-redundant control ($q_1$ actuated) in red, redundant control ($q_1$ and $q_4$ actuated) in green, reference in black circles and lines.

| | TNPT | | RPT | | TPT | | RPT | |
|---|---|---|---|---|---|---|---|---|
| Controller | $K_p[Nm]$ | $K_v[Nms]$ | $T_{s_p}[s]$ | $I_{pc}[-]$ | $K_p[Nm]$ | $K_v[Nms]$ | $T_{s_p}[s]$ | $I_{pc}[-]$ |
| NRD PD | 7 | 0.7 | 0.05 | 4.46 | 9 | 2.5 | 0.08 | 2,86 |
| RD PD | 6 | 0.3 | 0.03 | 6.55 | 6 | 1.5 | 0.07 | 2,04 |
| NRD PDFF | 5 | 0.2 | 0.05 | 5.36 | 5 | 2.7 | 0.15 | 1.84 |
| RD PDFF | 5 | 0.2 | 0.04 | 5.67 | 5 | 1.1 | 0.07 | 1.60 |
| Controller | $K_p[s^{-2}]$ | $K_v[s^{-1}]$ | $T_{s_p}[s]$ | $I_{pc}[-]$ | $K_p[s^{-2}]$ | $K_v[s^{-1}]$ | $T_{s_p}[s]$ | $I_{pc}[-]$ |
| NRD FFCT | 6 | 3.3 | 0.8 | 1.07 | 6 | 3.2 | 0.8 | 1.07 |
| RD FFCT | 20 | 5 | 0.38 | 1.95 | 5 | 5 | 1 | 1.86 |

Table 9.2: Gain values, settling time and integral error for the four-bar control simulations. Legend: NRD - "non-redundant actuation", RD - "redundant actuation", TNPT - "Tuning for the non-perturbed trajectory", TPT - "Tuning for the perturbed trajectory", RPT - "Response for the perturbed trajectory".

the different controllers with respect to disturbances. It is important to notice with respect to the important conclusions of the previous section, that non-redundant control schemes were nevertheless tested for the purposes of this general control comparison. In order to avoid the singularity blockage problems, commented above, and have control over the whole trajectory, the non-redundant control simulations were carried out using dynamic models that take into account the internal joint friction, whilst damping was not considered neither in the model, nor in simulation.

The following observations can be made on the basis of the tabular data:

1. For a given tuning method, the results in terms of position settling time of the perturbation response of the PD-control and the PDFF-control are comparable, and better than those of the FFCT-control, with a slight advantage of the redundant over the non-redundant actuation case for all the three control laws;

2. For a given control scheme, better results in terms of settling time are achieved using a tuning over the non-perturbed trajectory, whereas better results in terms of integral performance index are obtained using the perturbed trajectory tuning;

3. Better $I_{pc}$ scores are obtained for the redundant PDFF-controller that uses a direct torque feed-forward, than for the non-redundant PDFF-control;

4. In terms of overall scoring, the best $I_{pc}$ results for the non-redundant controllers are obtained using the FFCT-controller (as expected, due to the decoupling and the dynamic compensation). The results for the non-redundant case reveal gain coefficients $k_{v_i}$ that are close to the ones that could be found via the optimal damping condition $k_{v_i} = \sqrt{2k_{p_i}}$;

5. As regards the redundant controllers, the best score is achieved using PDFF-control;

6. The controllers tuned over the non-perturbed trajectory demonstrate less robustness with respect to the disturbances in the velocities, giving worse integral performance, but are more effective in terms of position response oscillation damping, i.e. settling time $T_{s_p}$.

The above conclusions on the relative controllers' performances are illustrated via simulations of the four-bar non-redundant control over the perturbed trajectory in Figures 9.14 - 9.15.

As it can be seen from the plots, when applying the three control algorithms, tuned over the non-perturbed trajectory, the PD and PDFF-control give better results in terms of position settling time for the price of worse damping (bigger overshoot, lesser robustness) with respect to the disturbances at velocity and acceleration level, compared to the FFCT-control.

### Non-redundant/redundant control comparison for the complete four-bar trajectory

In this section, the non-redundant and redundant action of some of the controllers is compared. The simulation results, obtained using the gains from Table 9.2, are given in the following figures.

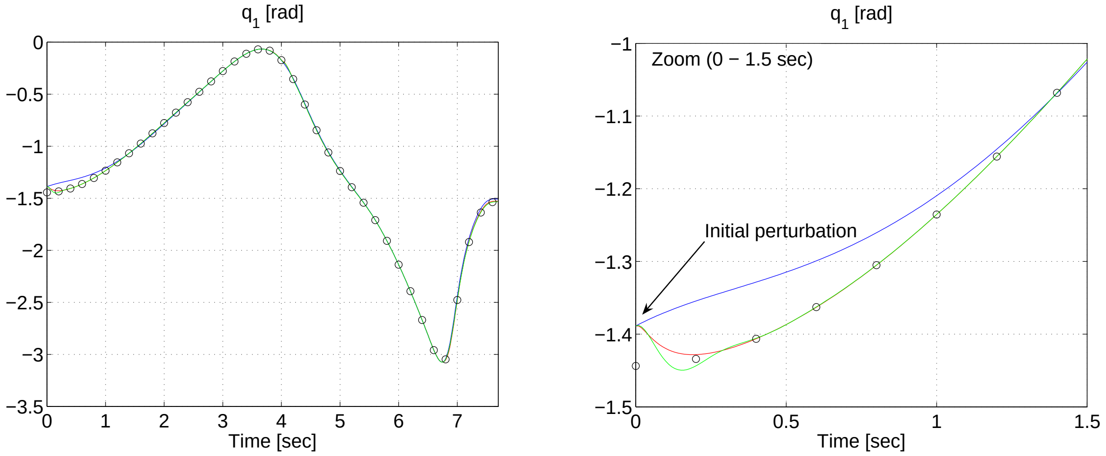



Figure 9.14: Four-bar mechanism non-redundant control simulation ($q_1$ actuated): joint $q_1$ angle response based on a non-perturbed trajectory tunning. Left: complete trajectory, right: excerpt (0 - 1.5 sec). PD-control in red, FFCT-control in blue, PDFF-control in green, reference data in black circles.

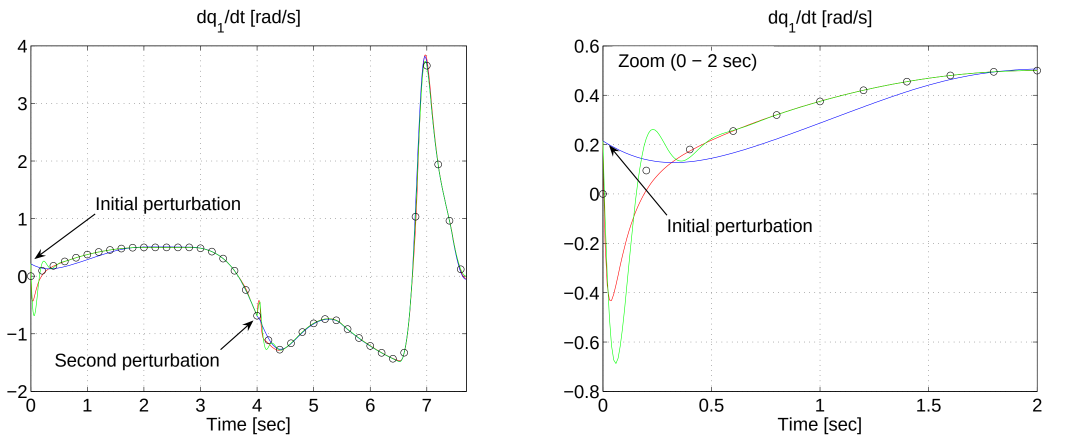



Figure 9.15: Four-bar mechanism non-redundant control simulation ($q_1$ actuated): joint $q_1$ velocity response based on a non-perturbed trajectory tunning. Left: complete trajectory, right: excerpt (0 - 2 sec). PD-control in red, FFCT-control in blue, PDFF-control in green, reference data in black circles.

As a first example, a comparison between non-redundant and redundant PDFF-control, based on a perturbed tuning, is shown in Figures 9.16-9.18.

Just as in the case of problematic non-redundant control when joint viscous friction (amounting to joint damping torques), not accounted for in the dynamic model, exists, the two force singularities (at 3.63 sec and 6.73 sec in Figure 9.10) can be detected visually. In Figure 9.18, left, they are represented by the discontinuities in the values both for the reference torque (computed in advance by the inverse dynamics robust computation algorithm) and the control input torque acting on joint $q_1$.

However, unlike the case of erroneous motion tracking (Figure 9.13), here the non-redundant PDFF-control law allows for tracking the complete trajectory, for the price of

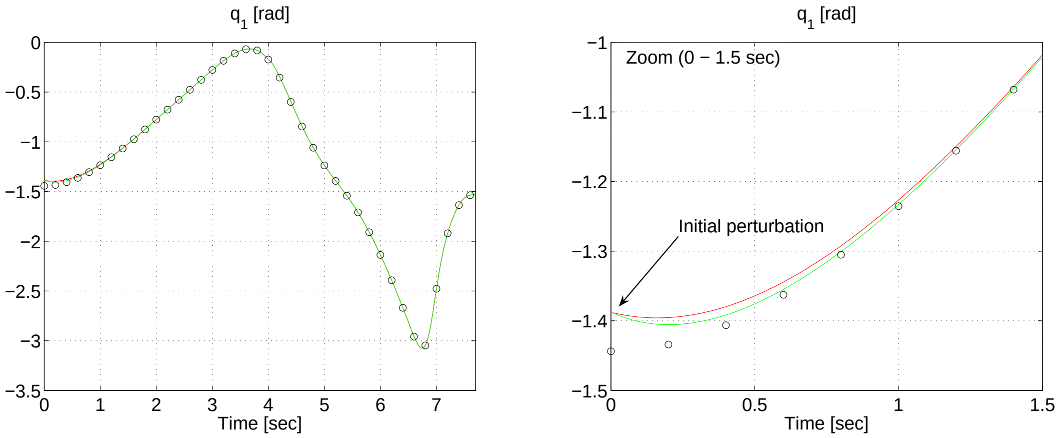



Figure 9.16: Four-bar mechanism non-redundant and redundant PDFF-control simulation: joint $q_1$ angle response, based on a perturbed trajectory tunning. Left: complete trajectory, right: excerpt (0 - 1.5 sec). Non-redundant control in red, redundant control in green, reference data in black circles.

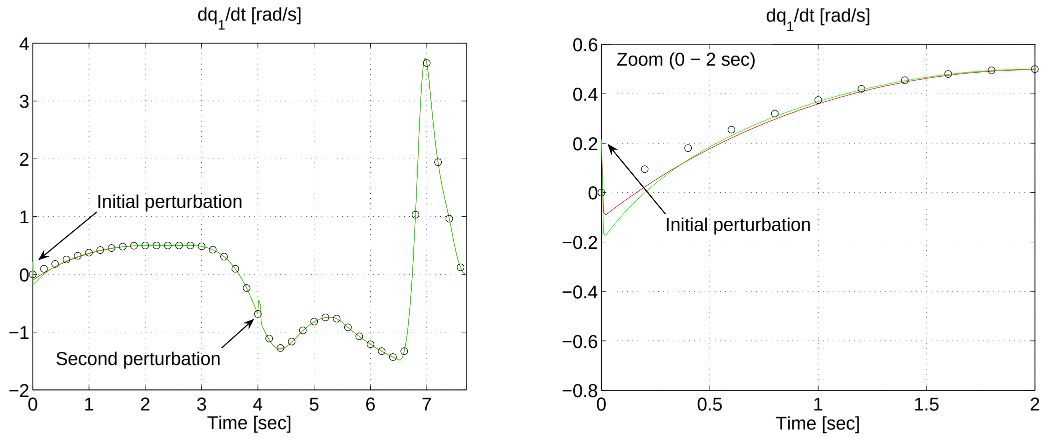



Figure 9.17: Four-bar mechanism non-redundant and redundant PDFF-control simulation: joint $q_1$ velocity response, based on a perturbed trajectory tunning. Left: complete trajectory, right: excerpt (0 - 2 sec). Non-redundant control in red, redundant control in green, reference data in black circles.

the torque problems of Figure 9.18. This is, from our point of view, a ”virtual reality” result, not ensuring robust manipulator operation and control in practical situations, in which, for instance, the joint friction and damping conditions usually vary in time (if the manipulator is not regularly inspected and looked after). In such situations, the unsuccessful non-redundant control, observed via simulation, would most probably occur, blocking and saturating the actuators at the corresponding singular configurations.

Let us note that in some of the figures in Section 9.4 and Section 9.5, peak values of some of the variables were cut out for the single purpose of having a better visibility for the graphical representations of the results. For example, in Figure 9.18, left, peak values both for the reference and the control torque were cut out by limiting the vertical axis to

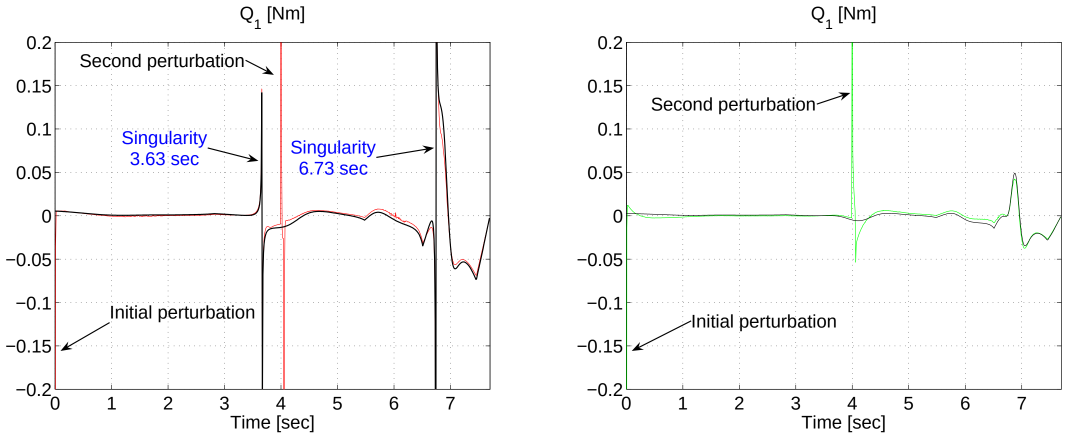



Figure 9.18: Four-bar mechanism non-redundant and redundant PDFF-control simulation: joint $q_1$ torque response, based on a perturbed trajectory tunning. Non-redundant control in red, redundant control in green, reference data in black.

an interval, equal to $[-0.2Nm \div +0.2Nm]$.

The results from simulations, comparing non-redundant and redundant FFCT-control of the four-bar mechanism, tuned over the perturbed trajectory, are presented in Figures 9.19 - 9.21. The FFCT-control demonstrates very good robustness with respect to external disturbances, both for its non-redundant and redundant variant.

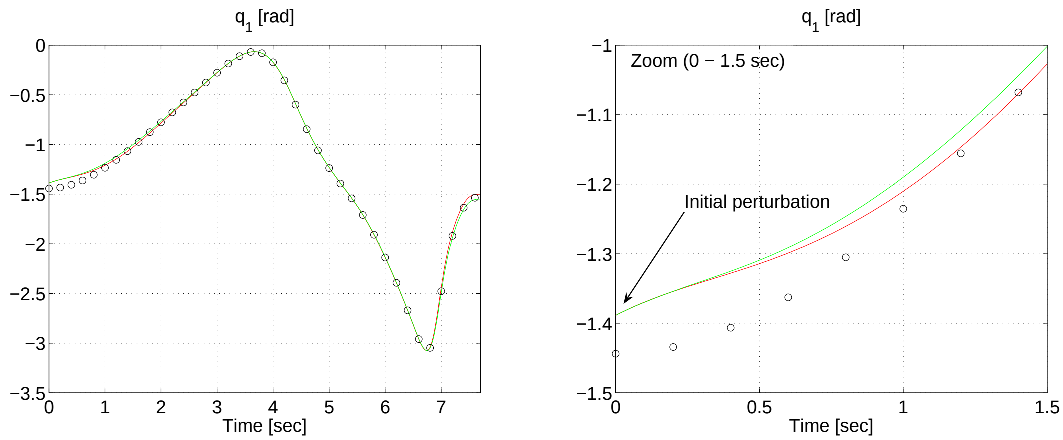



Figure 9.19: Four-bar mechanism non-redundant and redundant FFCT-control simulation: joint $q_1$ angle response, based on a perturbed trajectory tunning. Left: complete trajectory, right: excerpt (0 - 1.5 sec). Non-redundant control in red, redundant control in green, reference data in black circles.

However, a particular problem of the redundant FFCT-control exists that could make difficult its application if force singularities render its non-redundant form unusable. The problem consists in peaks of 0.02s duration, observed during simulation. These peaks actually constitute additional controller disturbances (see Figure 9.21, right, for a detail).

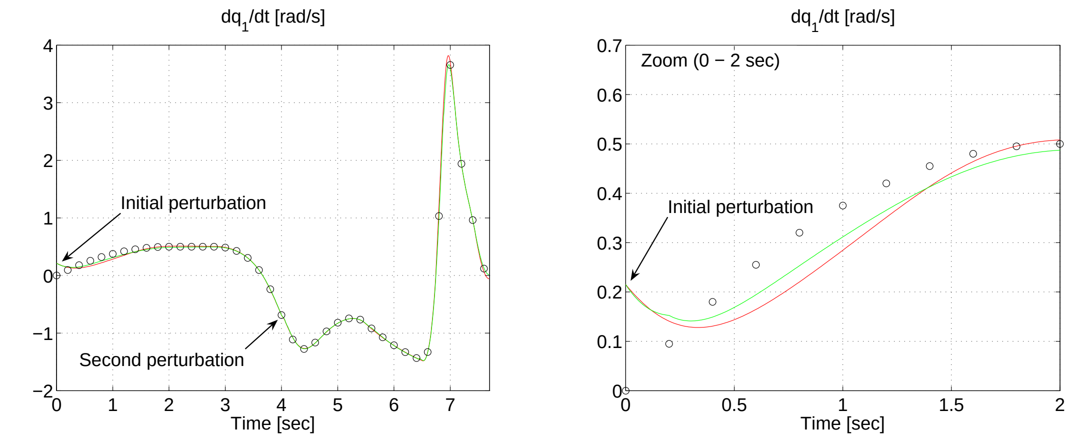



Figure 9.20: Four-bar mechanism non-redundant and redundant FFCT-control simulation: joint $q_1$ velocity response, based on a perturbed trajectory tunning. Left: complete trajectory, right: excerpt (0 - 2 sec). Non-redundant control in red, redundant control in green, reference data in black circles.

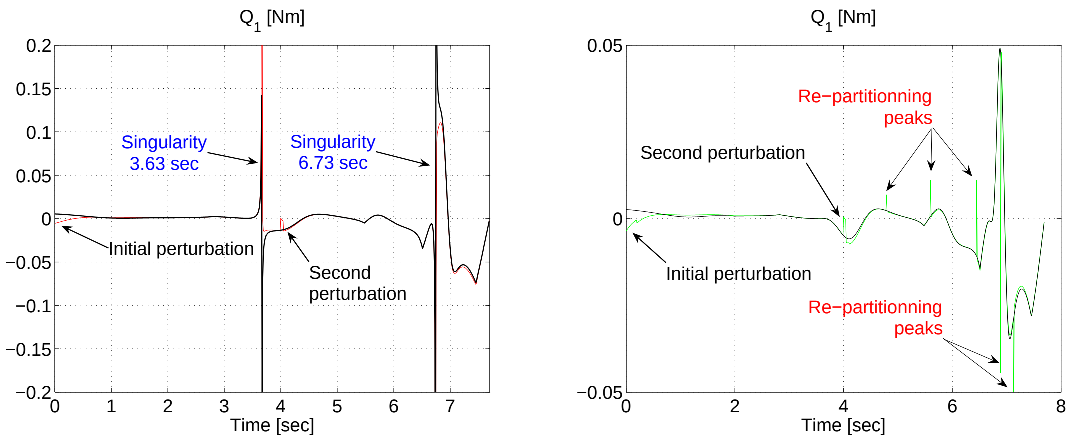



Figure 9.21: Four-bar mechanism non-redundant and redundant FFCT-control simulation: joint $q_1$ torque response, based on a perturbed trajectory tunning. Non-redundant control in red, redundant control in green, reference data in black.

These new artificial disturbances are due to the fact, that the computed-torque control schemes act *only* over the sub-space of actuated variables, the number of which corresponds to the number of MBS d.o.f. As already demonstrated in Chapter 8 (Section 8.2), the piecewise partitioning algorithm we developed will produce as many different $\{a, p\}$-partitions while computing the inverse dynamics, as it detects force singularities. The piecewise $\{a, p\}$-partition changes during control simulations cause local FFCT-controller "confusion", as it passes from one set of control (actuated) variables $q_{a_d}$ to another. The presence of these additional disturbances explains as well the worse results for $I_{pc}$ (bigger values) in this particular case of redundant control.

Moreover, the same "peak" problem would reappear in the case of hybrid control (Section 9.1.2, Figure 9.4), for instance, because the part of its control scheme, acting in the motion sub-space, contains the same feed-forward computed-torque controller, for which the $\{a, p\}$-partitions apply.

Some improvement should be undertaken, so as to smooth the successive control parameter "switches" due to the $\{a, p\}$-partition changes, and we envisage this as one of the directions of our future work.

## 9.5 Planar parallel manipulator: control simulations

The principal modeling and control simulation conditions we considered for the 3-d.o.f. planar parallel manipulator are identical to that for the case of four-bar mechanism. An important difference is the different number of mechanism degrees of freedom – for the planar parallel manipulator it is the biggest possible in the plane. Hence, a second major difference lies in the chosen test trajectories. The trajectory we used for the planar parallel manipulator is much more complex and reveals more force singularity configurations than that of the four-bar mechanism.

### 9.5.1 Multibody model and test description

Just like for the case of four-bar mechanism, modeling in relative coordinates and body-cut procedure are used for this closed-loop MBS.

The multibody topology of the planar parallel manipulator is shown in Figure 9.22. The

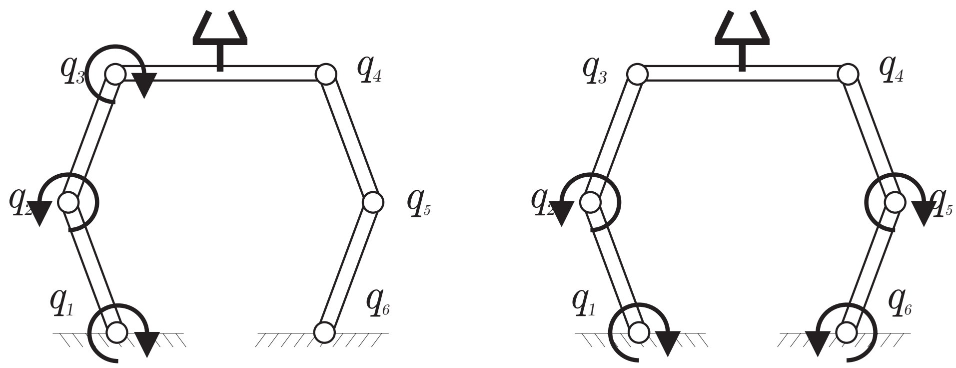



Figure 9.22: Topology representation of the 3-d.o.f. planar parallel manipulator. Left: case of non-redundant actuation (joint $q_1$, $q_2$ and $q_3$ actuated), right: case of simple redundant actuation (joints $q_1$, $q_2$, $q_5$ and $q_6$ actuated).

system model comprises five bars of identical length and mass, and six revolute joints. In case of non-redundant actuation control, joints $q_1$, $q_2$ and $q_3$ are actuated, whereas in case of redundant actuation control, a simple force redundancy (the number of actuators surpasses the number of d.o.f. by one) is chosen, with actuated joints $q_1$, $q_2$, $q_5$ and $q_6$.

The trajectory, used for the simulations of the four-bar, is of the type depicted in Figure 7.5, right, and is shown here in Figure 9.23 via animation"snapshots".

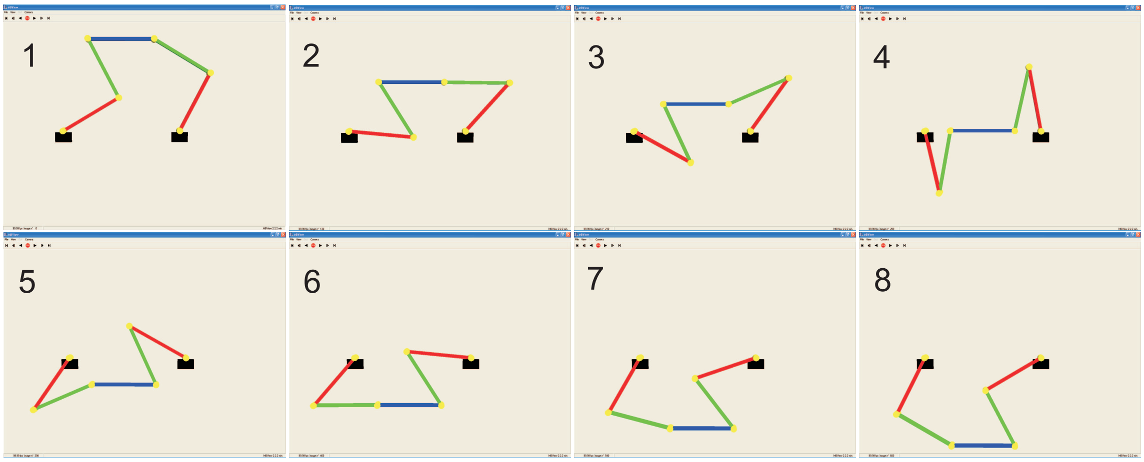



Figure 9.23: Trajectory of the planar parallel manipulator.

Of a 6-sec duration, it is characterized by a sinusoidal motion along the horizontal axis X and a constant-velocity translation along the vertical axis Y of the end-effector coordinate system. The third coordinate $\theta$ corresponding to the third degree of freedom in the plane, remains constant and equal to zero. The motion is accomplished in a horizontal (with respect to the gravity field) plane, therefore the effects of gravity are not accounted for in the modeling and the experimental procedure. The described combination of coordinate evolutions gives an end-effector motion (in the middle of the blue bar in Figure 9.23), which can be assimilated to a painting tool or window cleaning tool manipulation, for instance, and represents a reconfiguration of the manipulator, symmetric with respect to an axis, passing through the fixed revolute joints $q_1$ and $q_6$.

The trajectory reveals different force singularities, depending on the actuator configuration chosen. For instance, as se shall see later in this section, the force singularities of snapshots 2 and 6 are problematic, when actuating joints $q_1$, $q_2$ and $q_6$.

This time, five identical bars of 0.2m length and 0.15kg mass were used in the model and dry friction of the same friction coefficient of 0.015 was considered for the six revolute joints, in accordance with the prototype design. Viscous friction forces were again not taken into account for the principal part of control simulations.

An equivalent exhaustive simulation tuning procedure was performed using the same two problem cases: a non-perturbed trajectory case and a double perturbation case, using the same perturbation conditions. The actuator torque reference data is obtained again using the pseudoinverse-equivalent torque optimization solution.

### 9.5.2 Planar parallel manipulator: overcoming force singularities

Here again, the advantages of the redundant actuation in terms of control are demonstrated on the example of a PDFF-control, tuned by means of the exhaustive control simulation procedure. The used gain values correspond to the ones, given in Table 9.3, commented in the next section.

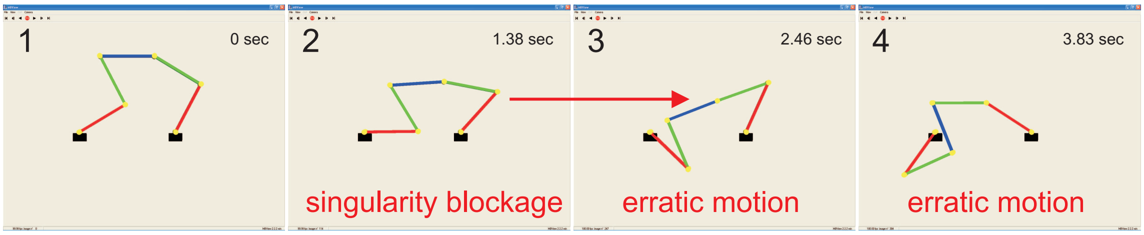



Figure 9.24: Snapshot sequence of the planar manipulator non-redundant PDFF-control in the presence of joint friction and damping torques: singularity blockage and subsequent bad tracking.

By analogy with the case of four-bar mechanism, the example consist in simulations of a non-redundant ($q_1$, $q_2$ and $q_6$ actuated) and a redundant ($q_1$, $q_2$, $q_5$ and $q_6$ actuated) PDFF-control of the four-bar mechanism for the same combination of joint dry friction and damping coefficients (0.015 and 0.0039 [Nm.s/rad], respectively). The joint damping is not accounted for in the dynamic model for this case either, recreating the practical situation, in which the joint friction terms of a given robot change in time (e.g. the lubrication conditions deteriorate).

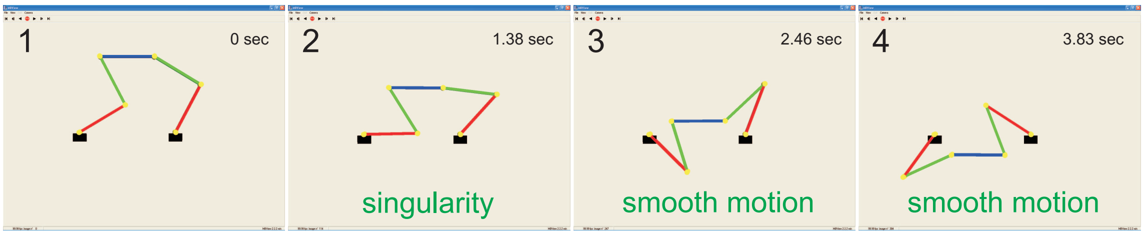



Figure 9.25: Snapshot sequence of the planar manipulator redundant PDFF-control in the presence of joint friction and damping torques: singularity effect elimination and smooth passage.

Sequences of snapshots are given in Figures 9.24 and 9.25 in order to visualize the virtual system behavior for the two cases – non-redundant and redundant actuation PDFF-control. The non-redundant PDFF algorithm, applied on joints $q_1$, $q_2$ and $q_6$ of the 3-d.o.f. planar parallel manipulator, does not perform successful tracking of the prescribed trajectory for the given combination of joint friction and damping. The tracking is perturbed by the singularity configuration of snapshot 2 (at about 1.38 sec), after which the manipulator demonstrates a completely erratic motion. The latter can be noticed as well in Figure

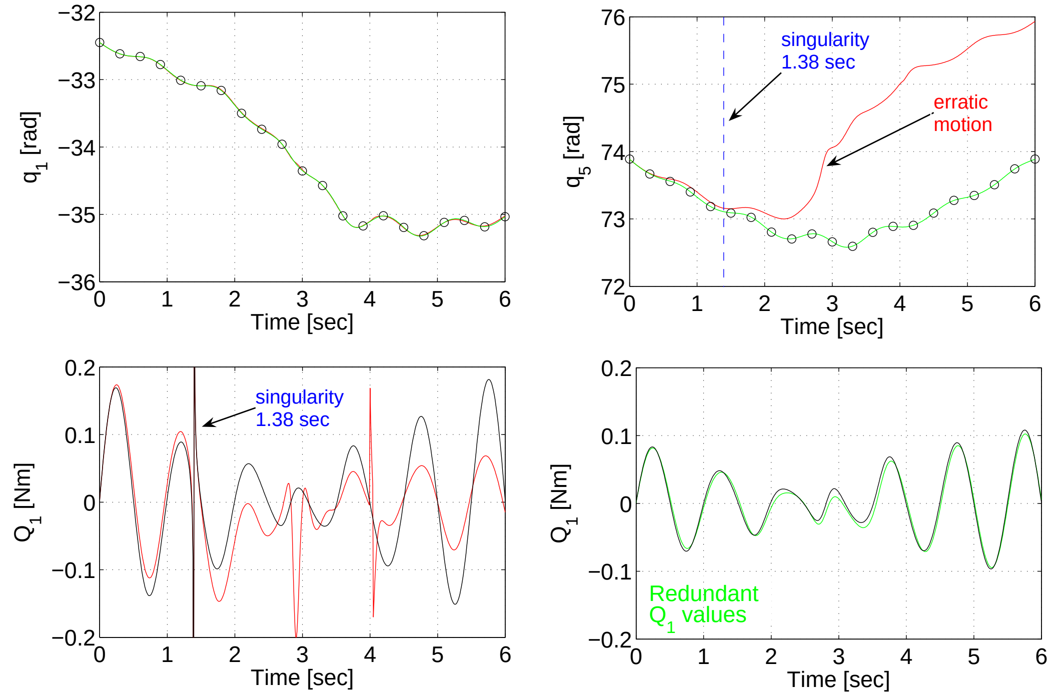



Figure 9.26: Planar manipulator PDFF-control in the presence of joint friction and damping torques: joint angle and torque responses. Non-redundant control ($q_1$, $q_2$ and $q_6$ actuated) in red, redundant control ($q_1$, $q_2$, $q_5$ and $q_6$ actuated) in green, reference data in black (circles and lines).

9.26 – by the time evolution of joint coordinate $q_5$ and joint torque $Q_1$, applied on $q_1$. For the case of redundant PDFF-control the prescribed trajectory is followed with very good precision and is free of force singularity problems. Thus, the conclusion on the advantage[10] of applying redundant control to parallel manipulators for trajectories that contain singularities, is reconfirmed and generalized by verifying it on a planar manipulator possessing all the degrees of freedom in the plane.

### 9.5.3 Comparison of the simulated control algorithms

Just as in the case of four-bar mechanism, exhaustive simulations over the non-perturbed trajectory and in the presence of the two perturbations were performed for the PD, PDFF and FFCT-control laws in order to tune them. The obtained gain values, settling times and integral errors for the planar parallel manipulator are given in Table 9.3.

Conclusions, similar to the simulation analysis of the four-bar mechanism, can be drawn here as well. In order to avoid unnecessary repetitions and the overflow of this text with graphical information, only the comparison between a non-redundant and a redundant

[10]That becomes a necessity in these examples!

| | TNPT | | RPT | | TPT | | RPT | |
|---|---|---|---|---|---|---|---|---|
| Controller | $K_p[Nm]$ | $K_v[Nms]$ | $T_{s_p}[s]$ | $I_{pc}[-]$ | $K_p[Nm]$ | $K_v[Nms]$ | $T_{s_p}[s]$ | $I_{pc}[-]$ |
| NRD PD | 5 | 1.6 | 0.08 | 3.74 | 5 | 1 | 0.08 | 3.38 |
| RD PD | 6 | 1.2 | 0.04 | 3.11 | 5 | 2.7 | 0.09 | 1.63 |
| NRD PDFF | 5 | 0.3 | 0.1 | 4.41 | 6 | 2.6 | 0.08 | 1.72 |
| RD PDFF | 2 | 0.19 | 0.04 | 3.22 | 5 | 3.2 | 0.1 | 1.59 |
| Controller | $K_p[s^{-2}]$ | $K_v[s^{-1}]$ | $T_{s_p}[s]$ | $I_{pc}[-]$ | $K_p[s^{-2}]$ | $K_v[s^{-1}]$ | $T_{s_p}[s]$ | $I_{pc}[-]$ |
| NRD FFCT | 1.8 | 4 | 1.35 | 2.66 | 1.4 | 4 | 1.3 | 2.69 |
| RD FFCT | 9 | 3 | 0.11 | 2.10 | 5 | 2 | 0.12 | 2.65 |

Table 9.3: Gain values, settling time and integral error for the planar manipulator control simulations. Legend: NRD - "non-redundant actuation", RD - "redundant actuation", TNPT - "Tuning for the non-perturbed trajectory", TPT - "Tuning for the perturbed trajectory", RPT - "Response for the perturbed trajectory".

PDFF-control for a case of successful tracking of the complete reference trajectory, is given in Figures 9.27-9.29. In order to have example variety, the non-perturbed controller tuning (TNPT) was chosen this time for the simulations. The position and velocity responses for joint $q_5$ are presented in Figures 9.27 and 9.28, whereas the joint torque $Q_1$ is shown in Figure 9.29. The reason for this choice of joints is that $q_5$ is not controlled (actuated) in the non-redundant control case, therefore it is interesting to compare the the force redundancy influence on its motion tracking, but in the same time a torque response graph can not be given for this joint under the non-redundant PDFF-control, so $q_1$ was chosen (as before) for that purpose.

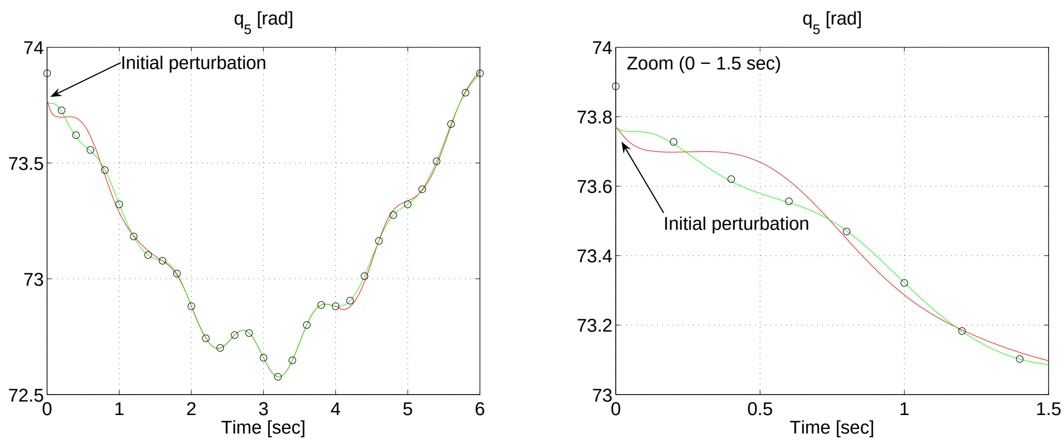



Figure 9.27: Planar manipulator non-redundant and redundant PDFF-control simulation: joint $q_5$ angle response, based on a non-perturbed trajectory tunning. Left: complete trajectory, right: excerpt (0 - 1.5 sec). Non-redundant control in red, redundant control in green, reference data in black circles.

The figures reconfirm the conclusion based on the numerical data of Table 9.3, of

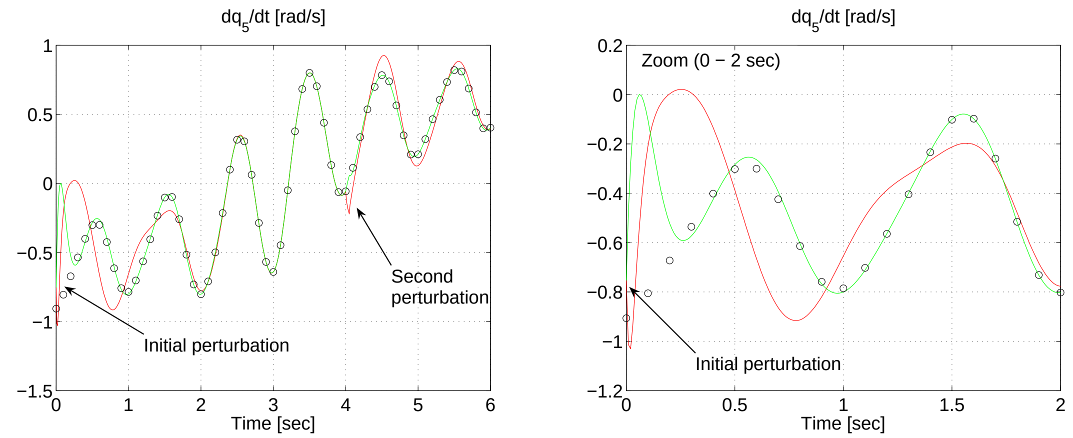



Figure 9.28: Planar manipulator non-redundant and redundant PDFF-control simulation: joint $q_5$ velocity response, based on a non-perturbed trajectory tunning. Left: complete trajectory, right: excerpt (0 - 1.5 sec). Non-redundant control in red, redundant control in green, reference data in black circles.

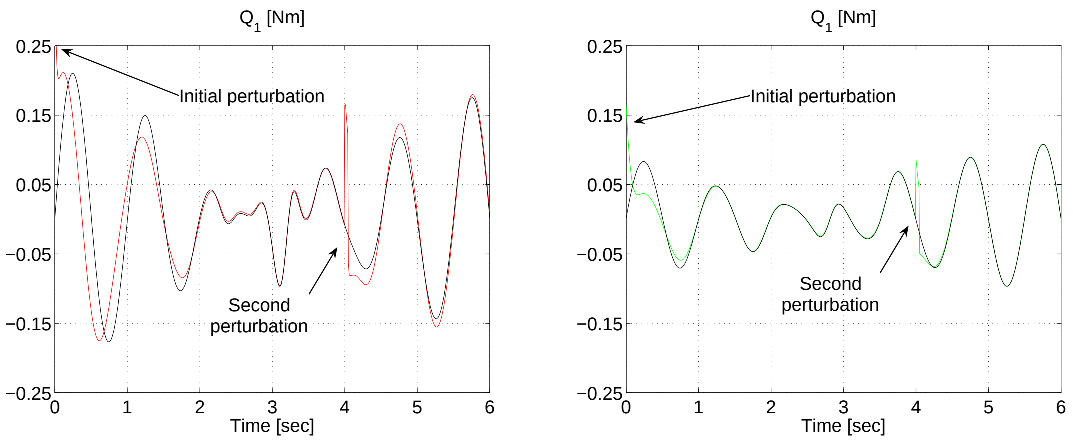



Figure 9.29: Planar manipulator non-redundant and redundant PDFF-control simulation: joint $q_1$ torque response, based on a non-perturbed trajectory tunning. Left: non-redundant control in red, right: redundant control in green. Reference data in black.

a better position settling time and lower integral errors, achieved with the redundant PDFF-control scheme.

Concluding this chapter, we would like to insist once more on the fact that in practical situations, one should not rely on simulation results of successful non-redundant control of parallel manipulators following singular trajectories, and thus has to apply redundant actuation in cases when this is physically possible for an acceptable cost. As it was demonstrated in this chapter, even the application of a *simple* degree of redundant actuation, i.e. only one motor more than the numbers of manipulator degrees of freedom, helps a lot in eliminating the force singularity problems for a rather complex parallel manipulator trajectory (see Figure 9.23).

Finally, the hybrid position/force control scheme of Figure 9.5 was not tested via simulation, because of the $\{a, p\}$-partition switching problems already mentioned and the necessity to artificially recreate on-line data from force/torque virtual "sensors", which induced further developments (commented in the prospects section), going beyond the time limits of this research work.

# Part IV

# Prototype Design and Experimentation

# Chapter 10

# Prototype Design

An important part of the present work consists in the design of a parallel manipulator prototype allowing an experimental validation of the theoretical developments we presented.

The prototype we designed was intended as providing a convenient and reliable means of validation of the theoretical developments of this research study. Therefore, we shall describe in the pages to follow those prototyping points that we consider as being of primary importance for the comprehension of the design concepts and solutions used. Through the latter, a prototype of a 3-d.o.f., light planar parallel manipulator is obtained, matching close enough the virtual models we treat.

## 10.1 Mechanical Design

The mechanical design of the prototype we designed targets the following main characteristics:

1. Design of a *light*[1] planar parallel manipulator;

2. Robustness of the construction;

3. Low mass and inertia of the manipulator constitutive elements (bodies, connection and transmission parts), in particular of those in motion;

4. Transmissions that do not engender friction torques, nor add significant masses to the moving bodies;

5. Possibility for bar length modification, in order to easily adjust the transmission belt tension;

6. Modularity (possibility to easy assemble a five-bar or six-bar mechanism from a four-bar mechanism);

[1]for which the actuators are fixed to the base, and not to the moving manipulator bars (see Section 1.2.2).

7. Element accessibility and easy mounting/dismounting manipulation for minor adjustments, modifications and repair.

As a prototyping solution, complying with these requirements and corresponding as well to the planar parallel manipulator topology (see Figure 9.22, right), the prototype shown in Figure 10.1 was designed.

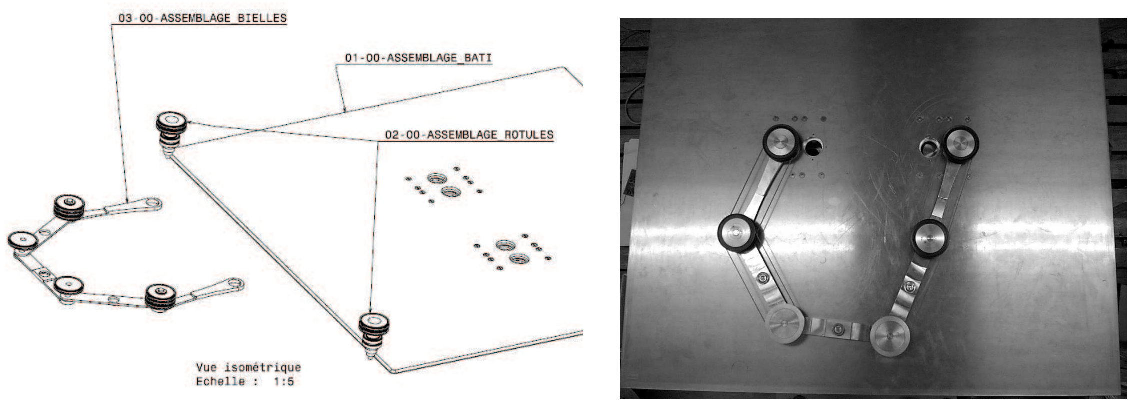



Figure 10.1: The 3-d.o.f. planar prototype (PRM Division, UCL). Left: excerpt from a drawing of the prototype assembly (3-D exploded view), right: a photograph of the real prototype.

Several design particularities have to be emphasized:

- Two pairs of at most three motors each are fixed beneath the base, by means of holders, specially designed for this purpose (Figure 10.2), in order to obtain a light parallel manipulator;

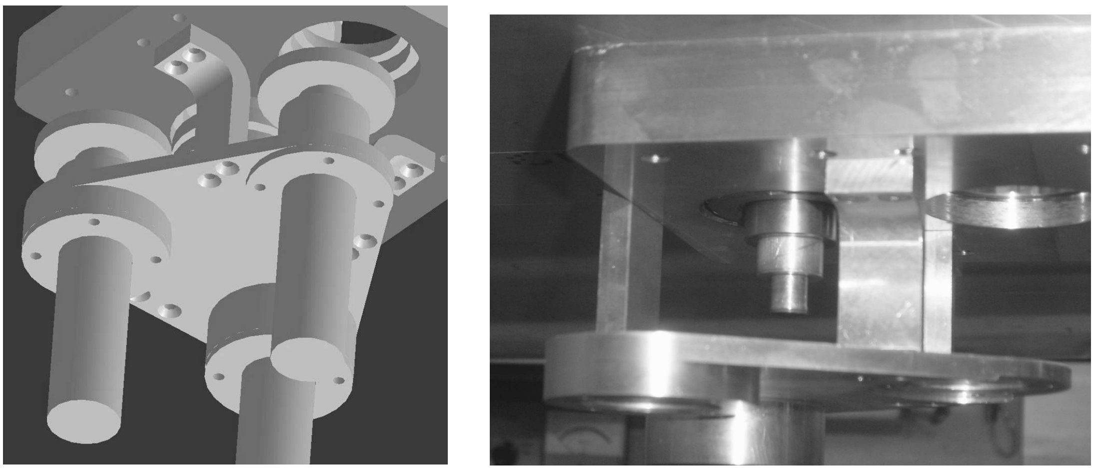


Figure 10.2: The prototype actuator block. Left: Catia 3-D view of the motor holder assembly (3 motors attached), right: the real motor holder (the lower extremities of the three concentric rotation transmission axes are clearly visible in the center of the photo).

- In each holder, three concentric axes form a rotation transmission block (Figure 10.2, right; Figure 10.3). The outer axis, fixed to the base via two ball bearings, ensures the rotation of the moving part of the first joint and the first bar of each parallel leg. The inner axes serve to transmit the rotation via pulleys and toothed belts to the rest of the joints;

- For each of the two motor holders, the motor/gear blocks are placed at 120 deg with respect to each other, and driving spur gears of equal diameter are directly coupled with the output shafts of the motor/gear blocks. The motor positioning allows for equal-distance access to the three concentric axes of each of the two joints, fixed to the base, as shown in Figures 10.2 and 10.3;

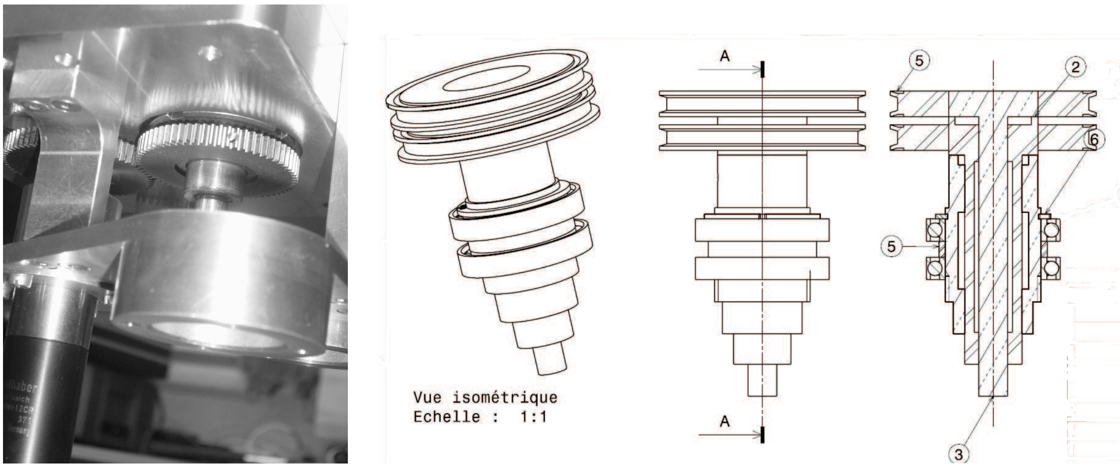



Figure 10.3: Transmission of the motor torques to the bar joints. Left: one of the motors mounted and its 1:1 ratio spur gear transmission, providing the motor torque to the corresponding axis. Right: excerpt from a drawing of the three concentric transmission axes.

- The transmissions used (Figure 10.4) are of toothed belt/pulley type, the diameter of the pulleys being chosen in standard correspondence with the toothed belts. The distances between the pulley axes correspond to the bar length, equal to 200mm.

## 10.2 Actuators, Sensors and Associated Hardware

### 10.2.1 Actuator and sensor choice

In order to have at our disposal a prototype, capable of performing rapid motion of large amplitudes while manipulating external loads, and be able to realize correct position/velocity tracking control over it, we chose as actuators four identical DC-motors with graphite commutation (Series 2657CR, Faulhaber: Annexe 10.3.2), produced by FAULHABER Group – one of the leading companies in the field of high quality, high performance drive systems. The motors are coupled with planetary gearheads of 1:66 gear

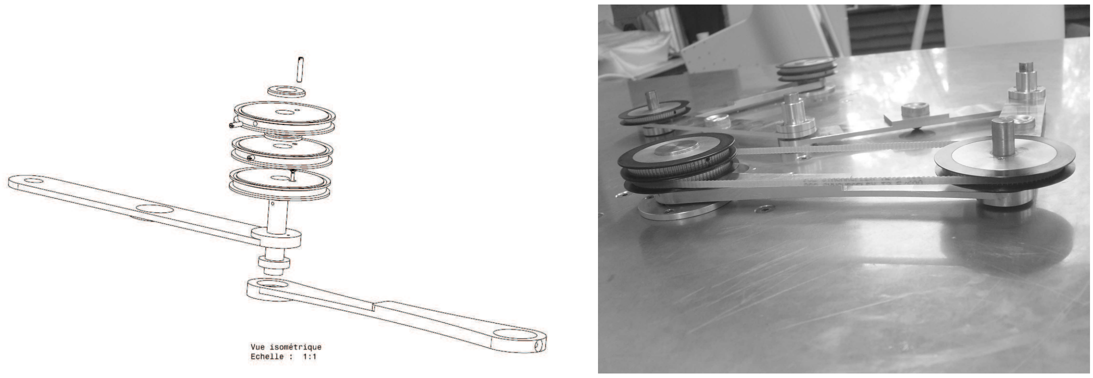



Figure 10.4: The prototype bars with their transmission parts. Left: Catia 3-D view of the bar assembly (2 bars with their common axis and pulleys), right: a view of the prototype bars with the transmission belts and pulleys.

ratio (Series 26/1, Faulhaber: Annexe 10.3.2) and 2-channel incremental magnetic shaft encoders (Series IE2-512, Faulhaber: Annexe 10.3.2) possessing 512 lines per revolution. This combinations provides actuators, delivering an output torque of 4.5 Nm (at the output shaft of the planetary gearhead) and sensors of a precision (0.01 deg), sufficient for correct position and velocity tracking.

## 10.2.2 Developed hardware components

Figure 10.5 represents the hardware conceptual solution we developed so as to create a convenient interface between the dedicated dSPACE controller PC-board, used for prototype real-time control, and the actuators.

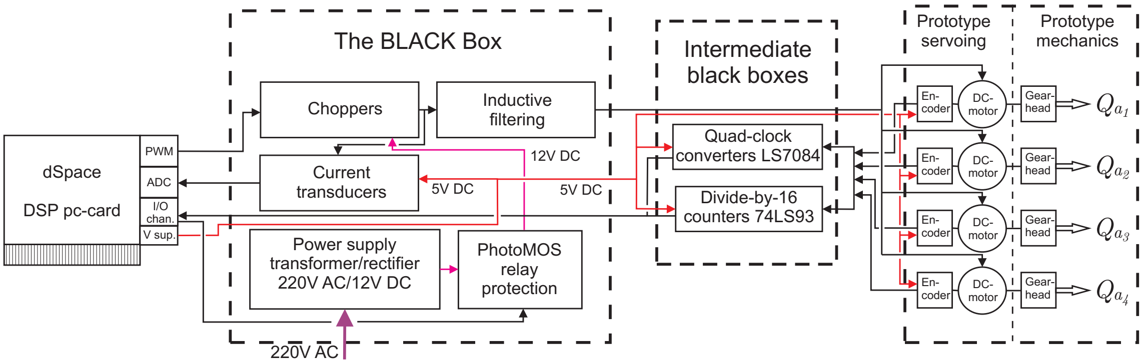



Figure 10.5: The developed hardware, interfacing the *dSPACE* DSP pc-card and the prototype servo-components.

The following hardware particularities have to be specified:

1. For the present prototype, a dedicated PC-station equipped with a dSPACE DS1102 DSP Controller Board was used. The DS1102 board has the following main characteristics:

- Texas Instruments TMS320C31 floating-point Digital Signal Processor (DSP) of 60 MHz clock rate;
- 130KB 32-bit RAM for user applications;
- 4 Analog-to-digital (ADC) input channels (2 parallel 16-bit channels of 4 $\mu s$ conversion time plus 2 parallel 12-bit channels of 1.25 $\mu s$ conversion time) of 10 V input voltage range;
- 4 parallel 12-bit DAC (digital-to-analog) channels (4 $\mu s$ typical settling time) of 10 V output voltage range;
- 16 digital I/O (input/output) bit-selectable lines, of which capture/compare unit with 8 channels (2 in, 4 out, 2 in/out) and PWM generation on up to 6 channels (40 $ns$ resolution); dedicated circuitry provides PWM frequency of up to 100 MHz;
- 2 incremental encoder interfaces - two parallel input channels for two phase lines and one index line each;
- Power supply: 5 V, 1.5 A and 12 V, 100 mA.

2. As the DS1102 board possesses only two incremental encoder interfaces, and for our prototype control applications up to 4 motors are controlled using feed-back data, in order to preserve the uniformity in the control synthesis 8 identical I/O channels were used for the encoder data (1 clock channel plus 1 sign channel per encoder). The encoder signals are transformed in a convenient form for the DS1002 I/O specifications, using LS7084 quad-clock converters (Annexe 10.3.2);

3. The control of the Faulhaber DC-motors is performed using choppers, providing effective current values according to a Pulse-width Modulation (PWM) over a supplied DC voltage of 12V; The 12V DC supply is delivered to the choppers using a separate high-quality stabilized transformer/rectifier, whereas the PWM signals are provided from the dSPACE controller board;

4. The on-line data on the currents through the motor armatures is obtained via current transducers and entered into the DS1102 board via its four ADC-channels. In our application, LEM LTS 15-NP transducers of 5V DC supply voltage (see Annexe 10.3.2) were used;

5. A protection PhotoMOS relay switch, controlled by a separate signal from the DS1102 board, was added to the main hardware interface box. The protection allows for avoiding extreme motor charging and possible subsequent physical damage of prototype when the choppers are at their passive state (absence of control signals when not in a real-time control motion), corresponding to 100% duty cycle. This problem is eliminated by the switch-on signal, sent to the PhotoMOS relay only for the duration of real-time control, during which appropriate PWM control signals are sent to the choppers from the DS1102 board.

6. Supplementary zener-diode protections were installed over the 5V and 12V DC supplies in the principal hardware circuitry, in order to protect the principal functional components - choppers, current transducers, incremental encoders, binary counters;

7. The design of print circuit boards (PCB) for the main and the intermediate interface cards (Figure 10.5) was carefully studied in order to have the different signals well separated and the shortest possible current ways. For the same reasons (avoiding signal interference and mutual noise generation), high quality screened cables were used for all the connections.

### Design and implementation problems – solutions adopted

Some problems that we encountered during the design of the described hardware required solutions, based on a compromise between prototype performances on one side, and overall cost and time on the other:

1. The limited memory for user applications of the DS1102 controller board (130KB) often caused problems when performing the real-time prototype control. These problems were indicated by the impossibility to compile the corresponding SIMULINK model and store it in the board for execution, or by the inability of the board to perform in real-time over the complete time duration of the trajectories considered, when the model was successfully compiled and loaded. In order to cope with this major hardware limitation problem, countermeasures were adopted:

   - The sampling frequency was limited to a range of frequencies from 4 to 4.5 KHz (SIMULINK integrator fixed steps from $2.5 \times 10^{-4}$ to $2.2 \times 10^{-4}$ sec);
   - The reference trajectory data was computed at a fixed step size of 0.01 sec (100 Hz) so as to reduce the size of data files to be stored in the DS1102 board;
   - The sampling frequency limitation necessitated a reduction of the large number of encoder pulses (33000 pulses/round, equal to 500 pulses/round at the motor shaft times the gearhead ratio of 66), in order to obtain a correspondence between the sampling frequency $f_s$ and the maximum number of encoder pulses $P_{enc}$ that can be detected without non-negligible error. According to the well-known *sampling theorem* (Nyquist [108], Kotelnikov [109]) of the information theory, the non-equality $f_s > 2B$ between a signal of limited bandwidth $B$ and a sampling frequency $f_s$ must be satisfied in order to have the signal reconstructed exactly via discrete sampling. Therefore, the number of encoder pulses was divided by a factor of 16, using the dividing binary counters 74LS93, shown in Figure 10.5. The obtained number of encoder pulses (2062.5) is correctly sampled for frequencies, greater than 4125 Hz, and still sufficient in terms of precision (0.003 rad).

2. Parasite noises were detected in the current measures, obtained from the transducers, during the initial phases of hardware testing. These noises were principally due

to the motor and chopper commutation, we eliminated them by means of:

- Hardware filtering – using self-inductances of appropriate characteristics, placed between the motors and the choppers, as well as between the transducer current measure signal and the ADC physical entries of the dSPACE board;
- Software filtering – using low-pass first-order filters at the output ports of the ADC-block in the corresponding SIMULINK/dSPACE models.
- The chopper power supply lines were separated via capacitor filters, so as to bring to a minimum the parasite noise emissions of their commutation.

## 10.3 Prototype Real-time Control

In the framework of the present research study, two principal closed-loop multibody mechanical systems were modeled: a four-bar mechanism and a planar parallel manipulator.

Modeling and control simulations of the two closed-loop MBS were presented in the previous chapter. Here, comments are given on experiments, carried out for a variant of the prototype that corresponds to the four-bar mechanism with two actuators, and for the complete prototype topology of the 3-d.o.f. light planar parallel manipulator with four actuators. The actuator number for the both cases corresponds to a single degree of actuation redundancy (see Chapter 4).

Two control algorithms were tested experimentally: a PD-control and a PDFF-control. The predictive computed-torque controller was not tested because of the memory limitations of the DS-1102 controller board.

Some additional experiments we did with a simplified (because of the same limitations) variant of a redundant hybrid controller turned out to be unsatisfactory in obtaining control results over the complete manipulator trajectory. We will comment more on this issue in the prospects of this work.

### 10.3.1 Software implementation

The software implementation of the prototype control algorithms was performed using SIMULINK and dedicated SIMULINK-compatible blocks of the dSPACE implementation software package *Real-Time Interface* (RTI), accompanying the DS1102 board. The structure of the models (under the form of connected SIMULINK blocks) and their main modules is briefly described hereunder.

Figure 10.6 shows the structure of the outer layer of all the plant control models that we developed.

Its main component sub-systems can be cited as follows:

- Controller sub-system

  In Figure 10.7 the structure of the designed SIMULINK sub-system representing a PDFF-controller, is shown. It corresponds to the PDFF-control algorithm, described

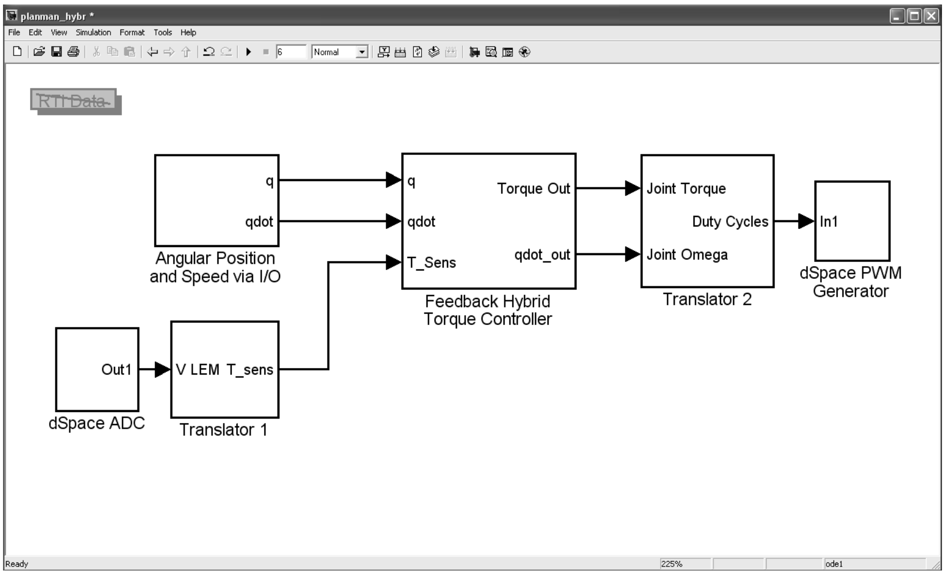



Figure 10.6: Main window with SIMULINK block diagram of the plant hybrid control model.

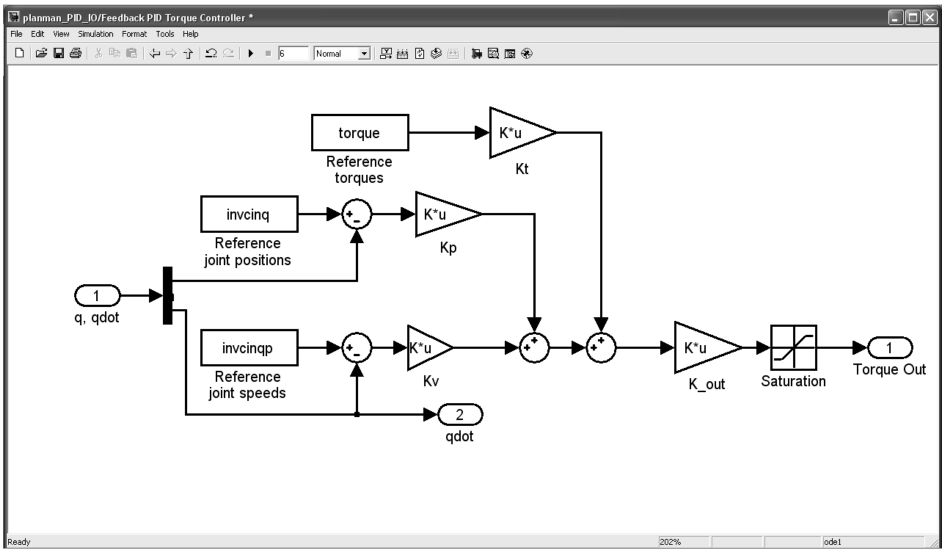



Figure 10.7: SIMULINK block-scheme of the PDFF-controller structure model.

in the previous chapter. Similar sub-systems can be found in the SIMULINK models of PD and FFCT-controller.

- Motor current/torque conversion sub-system

  Next important sub-system of the plant control model is the one converting the data on the motor armature currents (obtained in real-time from the current transducers) into torque values (Figure 10.8). The conversion is performed on the basis of corresponding DC-motor, gearhead and current transducer characteristics (see Annexe for details). The presence of low-pass first order software filters, smoothing the torque signals that still contain some current noise (already partially filtered by appropriate inductors), as well as the existence of "Emergency stop" block, containing the software control of the PhotoMOS protection relay (Figure 10.5), are to be noticed.

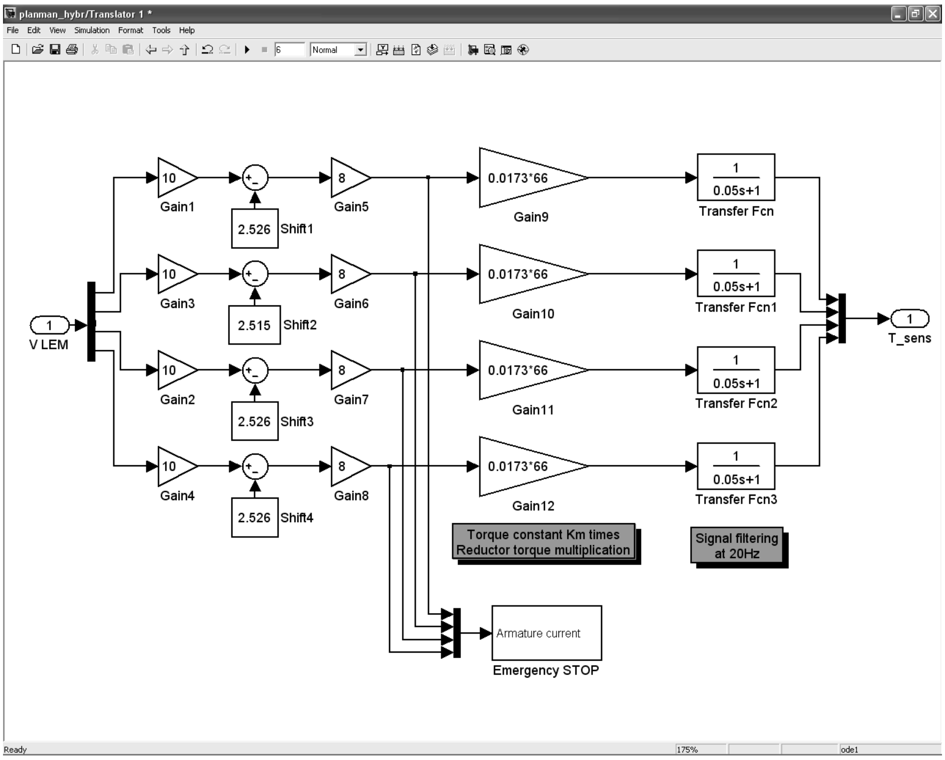



Figure 10.8: SIMULINK block diagram of the motor current/torque signal conversion and filtering. Input: motor armature currents (VLEM) [A], output: joint actuator torques (T_sens) [Nm].

- Position/velocity estimation sub-system

  As Input/Output ports of the DS1102 board were used to gather the data from the incremental encoders, serving as motion sensors attached to the motor shafts, an appropriate treatment of the data signals (encoder pulses in a rectangular form), received from the intermediate binary clock counter-and-divider interface was needed. The number of pulses entering the controller board is obtained using SIMULINK data-store blocks and crossing-value triggered switches, and data history is created using a "Transport-delay" block over an interval of 10 samples (0.01 sec) in order to have correct joint velocity estimates.

- Motor electro-mechanical model sub-system

  The electro-mechanical model of the four identical DC-motors is shown in Figure 10.10. The control torque inputs and the current joint velocities constitute its input data, and pulse-width modulation signals, sent to the corresponding PWM-entries of the dedicated dSPACE SIMULINK block, form the output of this SIMULINK model sub-system.

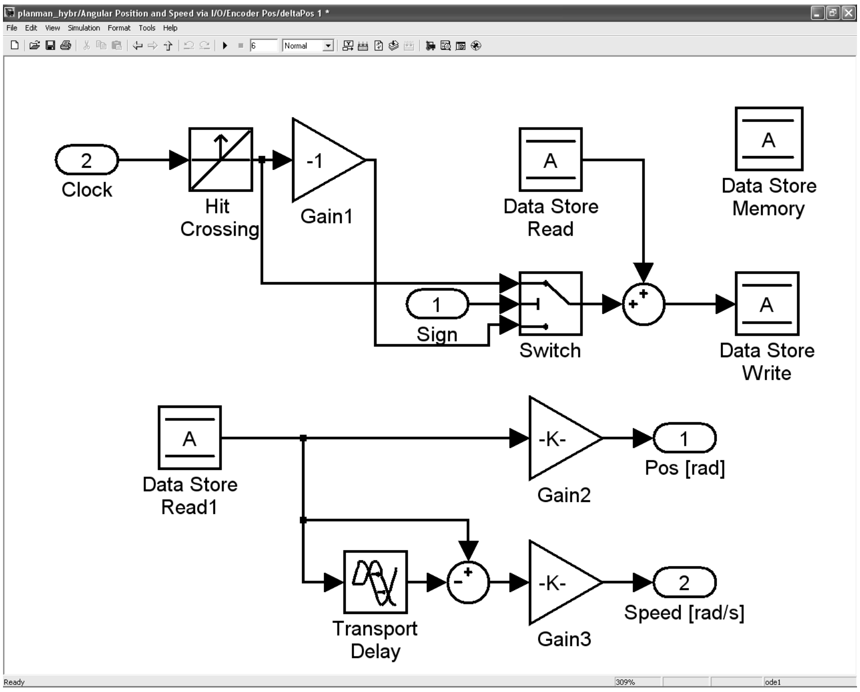



Figure 10.9: Position/Velocity estimate computation.

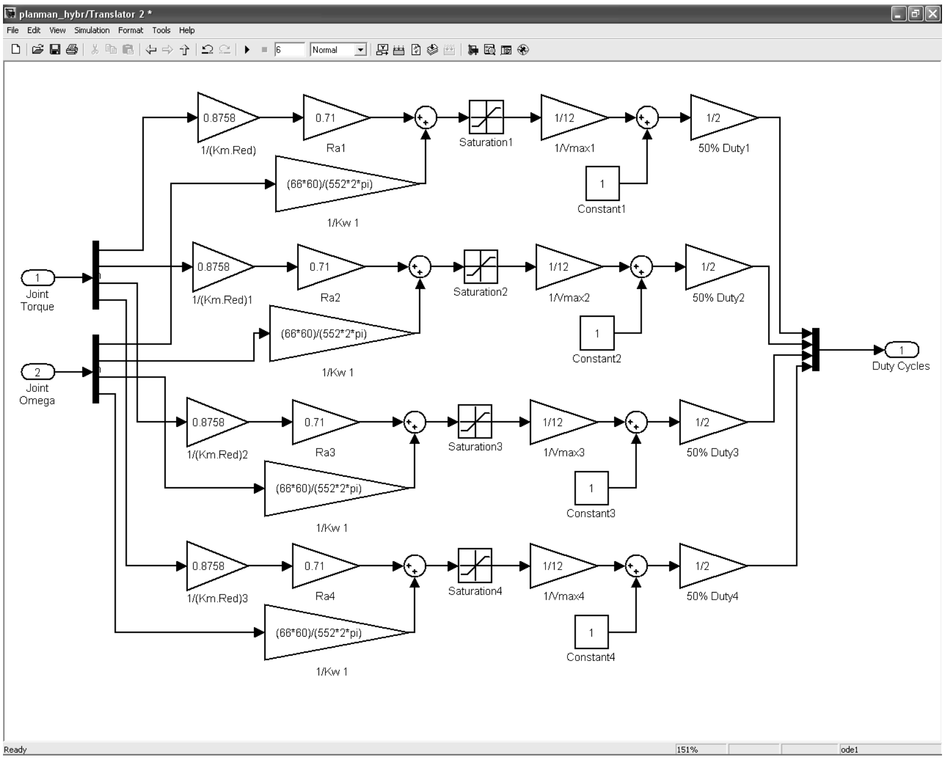



Figure 10.10: Block-diagram of the motor electro-mechanical model and PWM-signal construction. Input: joint control torques [Nm] and joint velocities [rad/s], output: PWM-values [-].

### 10.3.2 Experimental results

The trajectories, used for the experiments of the two prototype variants – the four-bar mechanism and the planar parallel manipulator, correspond exactly to the trajectories already described in Chapter 9 (Section 9.4, Figure 9.10 and Section 9.5, Figure 9.23).

#### "Virtual model – prototype" confrontation

The following important points, distinguishing the virtual models from the experimental ones, have to be emphasized:

- The DC-motor dynamics is not taken into account in the virtual models used for controlled system simulation analysis;
- Only joint friction torques, depending linearly on an approximated (via experiments on the prototype) friction coefficient and the normal components of the internal reaction forces are accounted for in the virtual models. Viscous friction components in the journal bearings, as well as friction components in the motors and all the gears are neglected, but exist in the real system;
- Non-negligible joint clearances are present in the real system.

#### Cases of singularity-perturbed non-redundant control: experimental demonstration

Impossibility to overcome force singularities and subsequent erratic motion deviating from the prescribed trajectories, were observed both for the four-bar mechanism and the planar parallel manipulator experiments, when applying non-redundant PD and PDFF-controllers.

In the previous chapter, simulation results were presented showing the real danger of relying on non-redundant control schemes to pass through force singularities, when the internal friction is underestimated (Sections 9.4.2, 9.5.2). Here, snapshot sequences taken from videos of the real mechanism behavior, are shown as a practical confirmation of the erratic manipulator behavior in such cases.

Figure 10.11 shows snapshots of the four-bar prototype motion for the non-redundant PDFF-control acting of joint $q_1$ (Figure 10.11, up) and the redundant PDFF-control, acting on $q_1$ and $q_4$ (Figure 10.11, down). In the case of non-redundant control, the force singularity at time t = 3.63 sec provoke temporary blockage of the prototype motion, and the prototype control continued unsuccessfully by an erroneous trajectory tracking. The redundant PDFF-controller completely eliminated this problem.

Let us recall, that the erroneous motion tracking of the prototype, due to underestimated joint friction and damping terms in the dynamic models, was quite realistically predicted by the simulation cases, imitating such specific functional conditions. In terms of visual comparison, the graphical results from the simulation of the four-bar erratic behavior is given here once more (Figure 10.12).

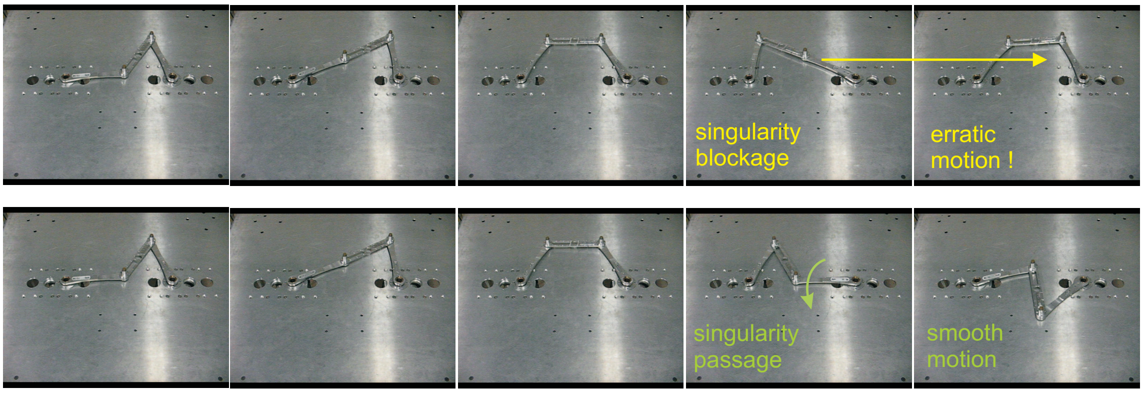



Figure 10.11: Comparison of four-bar prototype experiments with non-redundant and redundant PDFF-control (movie snapshots). Up: non-redundant control – force singularity perturbation, resulting in a bad trajectory tracking. Down: redundant control – precise tracking and smooth motion.

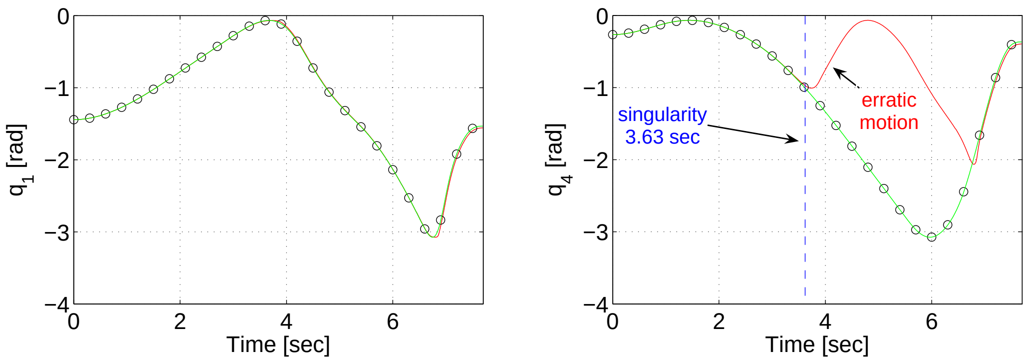



Figure 10.12: identical to Figure 9.13 (up), Section 9.4.2

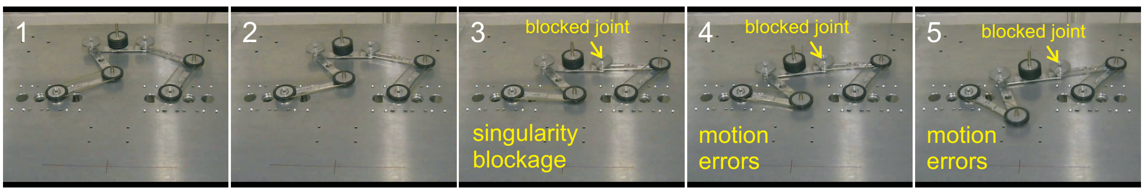



Figure 10.13: Erroneous tracking of the non-redundant PDFF-controller, applied on the planar manipulator prototype (snapshots).

The same phenomenon was observed when testing non-redundant and redundant PD and PDFF-control schemes on the planar parallel manipulator prototype. In Figure 10.13, identical problematic situation is given in several snapshots. The deviation from the prescribed trajectory, for which the intermediate bar (with the external load of three pulleys on it) has to remain horizontal, is clearly noticeable. The force singularity, represented by the alignment of two neighbor bars, blocks them in this configuration and causes deviation from the reference trajectory.

This major problem is successfully solved by using redundant control schemes, as it will be demonstrated by the experimental results in the next section.

### Planar parallel manipulator: redundant PD/PDFF-control comparison

The prototype control experiments on the basis of redundant controller schemes emphasized once more the advantages of applying redundant actuation to eliminate the effects of force singularities, present in the manipulator workspace. Moreover, good trajectory tracking over the complete trajectory was achieved using the redundant PD and PDFF-control schemes.

The experiment were carried out for gain coefficients $K_p = 350, K_v = 8$ for the PD-control and $K_p = 350, K_v = 8, K_t = 5$ in case of PDFF-control. These values were obtained manually (through experimental tuning) and are quite different (significantly higher) from the values, presented in the previous chapter and obtained via simulations. The difference is principally due to the discrepancies between the virtual models and the prototype, on which we already gave short comment in Section 10.3.2.

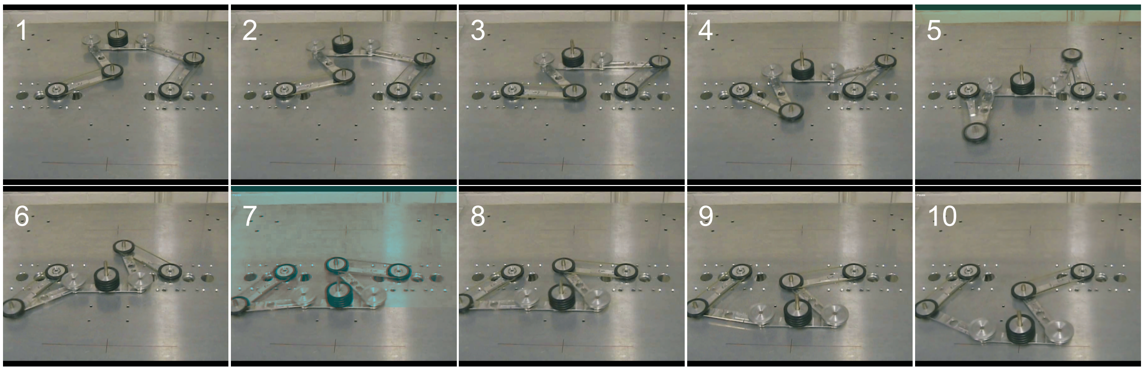



Figure 10.14: Successful redundant PDFF-control, applied on the planar manipulator prototype (snapshots). The complete trajectory is fully and precisely tracked by a smooth controlled motion.

Snapshots from the video taken during the experiments for the case of successful redundant PDFF-control, are shown in Figure 10.14. The prototype motion was smooth and free of force singularity disturbances along the prescribed path.

In Figure 10.15, the experimental results from the redundant PD and PDFF-control of the planar parallel manipulator prototype are presented in graphs. The following important observations can be made on the basis on the latter:

- The prototype demonstrates very good tracking precision at position level and acceptable precision at velocity level. Low integral errors of 0.2166 for the PD-control and 0.3975 for the PDFF-control were obtained, with $I_{pc}$ calculated on the basis of the position and velocity responses only (i.e. $W_p = W_v = 1, W_a = W_t = 0$, see Section 9.2);

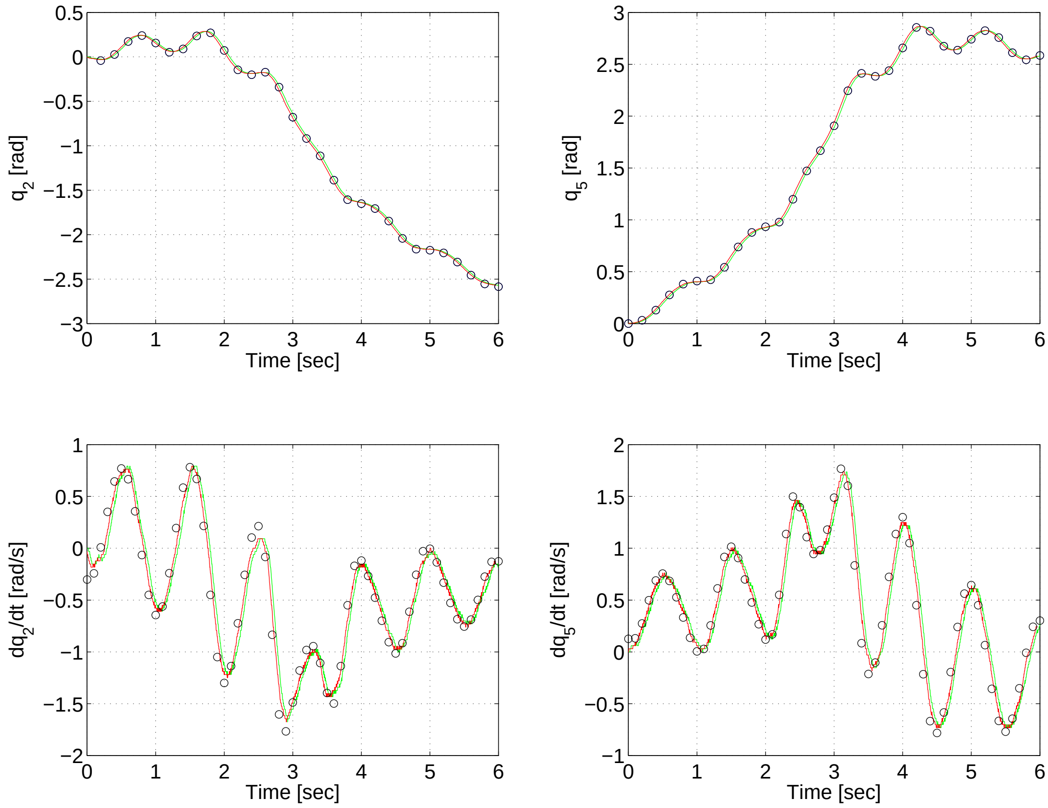



Figure 10.15: Prototype real-time redundant control: joint angle and velocity responses for $q_2$ and $q_5$. PD-redundant control in red, PDFF-control in green, reference data in black circles.

- For the chosen actuator configuration, the prototype follows the trajectory completely and without blocking or actuator saturation. The effects of force singularities (e.g. snapshots 3 and 8 in Figure 10.14) are eliminated, the controller applies joint torques that give smooth, continuous prototype motion.

Concluding this last chapter of the present research study, we would like to emphasize once again the fact that even applying a *simple* degree of actuation redundancy to parallel manipulators that follow complex trajectories with force singularities, can allow for stable real-time control of very good quality, for which the possible discontinuities in the multibody system motion, caused by the singularities and often accompanied with actuator blockage, saturation and even damage of motors or other mechanism components (gears, bearings, etc.), are successfully eliminated.

# Conclusions and prospects

## General conclusions

This work presents an approach for modeling and actuation of parallel manipulators that takes advantage from the force redundancy principles in order to eliminate the effects of parallel singularities existing in the manipulator workspace, and contribute to the effective manipulator control. The different stages of the approach were developed theoretically, then implemented in corresponding software applications, and validated by computer simulations and experiments.

The proposed approach principally targets:

- Elimination of force singularities that exist in the parallel manipulator workspace by means of redundant actuation and its application to the manipulator control, thus allowing for smooth motion over a workspace of increased volume;
- Satisfaction of optimal performance criteria (e.g. actuator torque minimization) when finding solutions for the force redundancy.

Several conclusions can be made on the basis of the accomplished research and development phases of the work:

1. The extensive utilization of the multibody formalisms [2, 3], of relative coordinate model formulation and effective symbolic generation tools [5, 3, 6] allowed to:
    - Deal with strict terminology and notations when describing closed-loop multibody systems (MBS), namely, parallel manipulators;
    - Generate parallel manipulator kinematic and dynamic models, convenient for the purposes of our work: numerically stable real-time control simulations that make use of coordinate partitioning techniques, specific actuation and control strategies, acting on the manipulator joints.
2. The reasoning of "actuating locally the joints, not subjected to force singularity problems", relying on practical "intuition" and hidden in the virtual actuation strategy, proved its effectiveness in eliminating the force singularities and produced *feasible*, realistic actuation solutions.

The *piecewise* actuation practically confirmed the need of using redundant actuation when continuous, singularity-free motion of the manipulator is sought. As a matter of fact, for the real treated manipulators the *virtual* changes in the actuator locations during the motion inevitably result in a permanent actuation of all the locally actuated joints, leading to actuator redundancy.

3. The choice of closed-loop MBS used for testing of the developed approach through simulations and experiments was adequate to the purposes of our work. The first "parallel manipulator"– a simple four-bar mechanism, allowed for very good visualization of the piecewise actuation approach and the elimination of force singularities using redundant actuation. The second system – a 3-d.o.f. planar parallel manipulator, offered more choices in terms of possible trajectories and actuator-dependent singularities to treat, and served very well to the computer and experimental validation of the redundant actuation and control strategies.

4. The actuator dynamics and better friction models have to be considered in order to emphasize the utility of control schemes that use reference torque values. We will give more comments on this when we give some prospects of our work.

5. As intended, the design of a parallel manipulator prototype constituted a convenient means of:

   - Assessment of the developed modeling and actuation approach. The prototype allowed for experimental validation of the force redundancy advantages when complex trajectories containing force singularities are followed;
   - Validation of the application of widely used controllers on redundantly actuated parallel manipulators.

Concerning the last point, we would like to mention that it was practically not possible to experiment with the predictive computed-torque control scheme, successfully tested via simulation. This was due to the hardware limitations we were confronted by. Furthermore, some additional experiments with a simplified scheme of hybrid control that did not bring valuable results revealed some modeling and control problems that still have to be solved. Therefore, we consider future experiments with these two controllers among the perspectives we cite hereunder.

# Main contributions

The following contributions of this research work could be pointed out:

1. A *unified* approach for modeling and actuation of redundantly actuated parallel manipulators was developed that allows for obtaining a *continuous*, smooth motion over the *complete* manipulator workspace by eliminating the effects of force (actuator-dependent) singularities, while satisfying actuator optimization criteria. The validity of this approach was tested via computer simulations and experiments;

2. For the treated parallel manipulators, kinematic and dynamic models that are stable with respect to numerical, kinematic loop closure problems were created. Together with the computation efficiency of the models, due to the symbolic generation tools used [5, 3, 6], this allowed to carry out real-time control simulations over any trajectory in the manipulator workspace and to develop a "pragmatic" controller tuning procedure, based on them.

3. The advantages of applying redundant actuation control to parallel manipulators following complex trajectories that contain parallel (force) singularities were demonstrated by means of computer simulations and experimentally.

   We showed that the problems caused by the force singularities (discontinuities in the applied joint torques, erroneous control, mechanism locking, actuator saturation ...) can be successfully eliminated by the redundant control inputs and that the optimized solutions for the actuator torques, used as an additional (feed-forward) reference by the controllers, have a positive influence on the control quality.

4. The topology and the particular design of our prototype allow for its use as an educational tool.

   The prototype in its feasible variants (four-bar, five-bar or six-bar) offers a very good practical illustration of parallel manipulators, possessing from one to all the three possible degrees of freedom of the planar motion. The corresponding workspaces contain multiple actuator-dependent singular configurations. Thus, this prototype will undoubtedly become an interesting educational tool, used for instance in the framework of courses in multibody dynamics and robotics. Moreover, it gives an excellent visualization of the redundant actuation advantages in terms of force singularity elimination, and its effectiveness with this regard when applied via known control schemes.

# Prospects

Several perspective research directions, based on the conclusions and the contributions of the present research, can be outlined:

1. Validation of the developed actuation and control strategies on complex 3-D parallel topologies, e.g. Stewart platform or Hunt manipulator topology, for families of trajectories containing force singularities.

   As already demonstrated in this work, even applying a single degree of redundant actuation gives promising results in terms of force singularity elimination and workspace enlargement. Therefore, it will be of great interest to broaden the family of parallel manipulators concerned, testing the developed approach on 3-D parallel manipulators that have already found or will find applications in different domains of the everyday life.

2. Improvement of the redundant control algorithms, tested by simulation and/or experiments.

   - In Chapter 9 (Section 9.4.3), we commented the particular additional disturbances in the feed-forward computed torque control simulations, due to the active/passive coordinate re-partitioning, performed during the real-time control and necessary for effective force singularity elimination. Possible solutions to this problem would, for instance, include the use of convenient gain scheduling techniques in order to ensure smooth passage from one set of controlled variables to another, or the application of sliding-mode control algorithms. Moreover, in Chapter 10 we explained that the experimental testing of this controller was not possible because of the limited memory of the hardware controller board. Successful experiments with this control algorithm on a DSP-board of higher performances would be of great interest to this work, e.g. as one more confirmation of its conclusions and contributions.
   - The redundant hybrid control scheme of Figure 9.5 was not tested via simulation, as it required further developments with respect to creating estimates for the feed-back data from torque sensors, which do not exist in the original direct dynamic models and thus are not present in the simulation loop. Such estimates are classically built using techniques of designing an additional dynamic system, called *state observer* [91, 92], that uses as input the system inputs and outputs in order to create in its turn outputs for the unmeasurable system states. Models of hybrid controller with state observer and their subsequent simulation validations were not planned in the framework of the present research thesis, because this required additional software and hardware developments that go beyond its limits.

     Some experimental tests, not interfering with these limits, were carried out though on the designed prototype. The tests were performed using a simplified hybrid control scheme, depicted in Figure 10.16, for which a PD-control instead of predictive computed-torque control was considered in the motion sub-space and the active/passive coordinate piecewise partitioning procedure was not applied. The first modification was imposed by the hardware limitations cited earlier, the second – by the fact that the *motion* sub-space part of the redundant hybrid controller of Figure 9.5 would also suffer from disturbances caused by the on-line changes of the $\{a, p\}$ partitions.

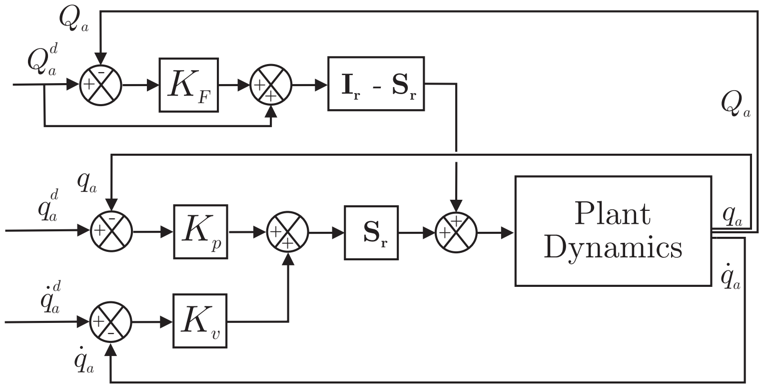



Figure 10.16: A "reduced" hybrid control scheme, tested through experiments on the developed prototype.

These last experiments demonstrated correct reference value tracking both in the motion and the force sub-space, but eventually lead to deviation from the reference trajectory for the joint coordinate controlled in the force sub-space, and manipulator leg collisions due to the control error. The experimental results revealed two important issues:

- The lack of the sequence of active/passive coordinate partitions that ensures actuation and motion stability for cases of trajectories with force singularities leads to low control quality;
- Better estimates for the actuator torques need to be computed on the basis of better models for the joint friction and the actuator dynamics, in order to have hybrid control of better quality. Moreover, other correction terms (e.g. integral term) could be added in the force sub-space, provided that hardware of better performances (higher amount of available memory) was employed.

Just as for the predictive computed-torque control, if applying the series of appropriate active/passive partitions, a convenient treatment (e.g. gain scheduling techniques) of the partition changes should be considered so as to avoid the appearance of controller disturbances for these changes. Moreover, modeling and testing of hybrid control schemes including state observer can also be envisaged in the prospects of this work.

3. Joint backlash cancellation using the redundant actuation.

We already mentioned in Chapter 4 that the elimination of joint clearances often existing in the parallel manipulator assembly, is an interesting and perspective domain of force redundancy applications. The productive work of A. Muller (e.g. [64, 97]) can be cited again with regard to this.

A preliminary research in this direction and a symbolic generation of some parallel manipulator models containing backlash were performed by the author of this work as well. The modeling was based on the techniques we used in the present research, and took advantage of the developed in [110] unilateral contact models.

The backlash modeling and elimination through redundant actuation was eventually left out of the scope of this study. Nevertheless, a sufficient basis for research in this direction was created. Therefore, we think that to continue the latter profiting from the present work contributions would be an excellent perspective. What is more, non-negligible backlash is present in the gears of the prototype we designed, therefore the latter would provide a convenient means of validation of future actuation and control strategies that eliminate not only force singularity problems, but the mechanism backlash as well.

4. Other prospects.

   Different applications could be figured out that would take advantage of the contributions of our work. For example, redundant parallel micro-manipulators, redundant surgical instrument mechanisms or redundant human prostheses could be imagined, characterized by high structure rigidity and slow motion properties, that allow for higher degrees of force redundancy and straightforward application of singularity elimination strategies. Reconfigurable hybrid or parallel manipulators, performing high motion precision at high speeds, and possessing high payload ratio, could be designed for use in specific environments that require multiple obstacle avoidance together with high performance criteria satisfaction, etc.

# Acknowledgments

This work has been partially supported by the Belgian Program "Interuniversity Poles of Attraction", initiated by the Belgian State – Prime Minister's Office – Science Policy Programme (IAP-P5/06). The scientific responsibility is assumed by the authors.

# Bibliography


[1] Willems P.Y. *Introduction à la mécanique.* Masson, Paris, 1979.

[2] Wittenburg J. *Dynamics of Systems of Rigid Bodies.* Stuttgart: Teubner, 1977.

[3] Samin J.C. and Fisette P. *Symbolic Modeling of Multibody Systems.* Solid mechanics and its applications, Volume 112. Kluwer Academic Publishers, 2003.

[4] Wehage R.A. and Haug E.J. Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems. *Journal of Mechanical Design*, 134:247–255, 1982.

[5] Fisette P. and Samin J.C. *Robotran Symbolic Generation of Multibody System Dynamic Equations.* W. Schiehlen (ed.): Advanced Multibody System Dynamics: Simulation and Software Tools, Dordrecht, the Netherlands, 1993.

[6] Postiau T. *Généralisation et parallélisation des équations du mouvement de systèmes multicorps par l'approche symbolique.* PhD thesis, Université catholique de Louvain, Louvain-la-Neuve, Septembre 2004.

[7] Asimov I. *I, Robot.* Gnome Press, 1950.

[8] Stewart D. A platform with six degrees of freedom. In *UK Institution of Mechanical Engineers*, volume 180(15), 1965-66.

[9] Clavel R. *Conception d'un robot parallèle rapide 4 degrés de liberté.* PhD thesis, EPFL, Lausanne, Switzerland, 1991.

[10] Clavel R. Device for the movement and positioning of an element in space. US Patent No. 4,976,582, December 11, 1990.

[11] Merlet J.P. *Parallel Robots.* Solid mechanics and its applications, Volume 74. Kluwer Academic Publishers, 2000.

[12] Gosselin Cl. and Hamel J.-F. The agile eye: a high-performance three-degree-of-freedom camera-orienting device. In *IEEE International Conference on Robotics and Automation*, pages 781–786, San Diego, California, USA, 1994.

[13] Dietmaier P. *"The Stewart-Gough Platform of General Geometry Can Have 40 Real Postures"*. Advances in Robot Kinematics : Analysis and Control. Kluwer Academic Publishers, July 1998.

[14] Gosselin C., Sefrioui J., and Richard M.J. Solution polynomiale au problème de la cinématique directe des manipulateurs parallèles plans à 3 degrés de liberté. *Mechanism and Machine Theory*, 27(2):107–119, March 1992.

[15] Wang S.L. and Waldron K.J. A study of the singular configurations of serial manipulators. *Transactions of ASME, Journal of Mechanism, Transmissions and Automation in Design*, 109:1420, 1987.

[16] Notash L. Uncertainty configurations of parallel manipulators. *Mechanism and Machine Theory*, 33:123–138, 1998.

[17] Zhao J.-S., Feng Z.-J., Zhou K., and Dong J.-X. Analysis of the singularity of spatial parallel manipulator with terminal constraints. *Mechanism and Machine Theory*, 40:275–284, 2005.

[18] Park F.C. and Kim J.W. Manipulability and singularity analysis of multiple robot systems: a geometric approach. In *IEEE International Conference on Robotics and Automation*, pages 1032–1037, Leuven, Belgium, May, 1998.

[19] Ghosal A. and Ravani B. Differential geometric analysis of singularities of point trajectories of serial and parallel manipulators. *Transactions of ASME, Journal of Mechanical Design*, 123:8089, 2001.

[20] Gosselin Cl. and Wang J. Singularity loci of planar parallel manipulators with revolute actuators. *Robotics and Autonomous Systems*, 21:377–398, 1997.

[21] Mohammadi Daniali H.R., Zsombor-Murray P.J., and Angeles J. Singularity analysis of platform-type multi-loop spatial mechanisms. *Mechanism and Machine Theory*, 30:665–678, 1995.

[22] Yang G., Chen I.-M., Lin W., and Angeles J. Singularity analysis of three-legged parallel robots based on passive-joint velocities. In *IEEE International Conference on Robotics and Automation*, pages 2407–2412, Seoul, Korea, May 21-26, 2001.

[23] Tchon K. Singularities of the euler wrist. *Mechanism and Machine Theory*, 35:505–515, 2000.

[24] Husty M.L. and Karger A. Architecture singular planar stewart-gough platforms. In *Proceedings of the 10th workshop RAAD*, Vienna, Austria, 2001.

[25] Choudhury P. and Ghosal A. Singularity and controllability analysis of parallel manipulators and closed-loop mechanisms. *Mechanism and Machine Theory*, 35:1455–1479, 2000.

[26] Basu D. and Ghosal A. Singularity analysis of platform-type multi-loop spatial mechanisms. *Mechanism and Machine Theory*, 35:375–389, 1997.

[27] McCarthy M.J. and Bodduluri R.M. Avoiding singular configurations in finite position synthesis of spherical 4r linkages. *Mechanism and Machine Theory*, 35:451–462, 2000.

[28] Wenger Ph. and Chablat D. Design of a three-axis isotropic parallel manipulator for machining applications: The orthoglide. In *WORKSHOP on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 16–24, Quebec City, Quebec, Canada, October 34, 2002.

[29] Zhou H. and Ting K.-L. Path generation with singularity avoidance for five-bar slider-crank parallel manipulators. *Mechanism and Machine Theory*, 40:371–384, 2005.

[30] Dash A.K., Chen I-M., Yeo S.H., and Yang G. Workspace generation and planning singularity-free path for parallel manipulators. *Mechanism and Machine Theory*, 40:776–805, 2005.

[31] Sen S., Dasgupta B., and Mallik A.K. Variational approach for singularity-free path-planning of parallel manipulators. *Mechanism and Machine Theory*, 38:1165–1183, 2003.

[32] Tchon K. Singularity avoidance in robotic manipulators: A differential form approach. *System and Control Letters*, 30:165–176, 1997.

[33] Beiner L. Singularity avoidance for articulated robots. *Robotics and Autonomous Systems*, 20:39–47, 1997.

[34] Gonzalez L.J. and Sreenivasan S.V. Representational singularities in the torque optimization problem of an active closed loop mechanism. *Mechanism and Machine Theory*, 35:871–886, 2000.

[35] Wen J.T. and Wilfinger L.S. Kinematic manipulability of general constrained rigid multibody systems. *IEEE Transactions on Robotics and Automation*, 15(3):558–567, 1999.

[36] Ganovski L., Fisette P., and Samin J.-C. Piecewise overactuation of parallel mechanisms following singular trajectories: modeling, simulation and control. *Multibody System Dynamics*, 12:317–343, 2004.

[37] Zlatanov D., Fenton R.G., and Benhabib B. Identification and classification of the singular configurations of mechanisms. *Mechanism and Machine Theory*, 33(6):743–760, 1998.

[38] Dasgupta B. and Mruthyunjaya T.S. Force redundancy in parallel manipulators: theoretical and practical issues. *Mechanism and Machine Theory*, 33(6):727–742, 1998.

[39] Birglen L. and Gosselin Cl. Force analysis of connected differential mechanisms: application to grasping. *The International Journal of Robotics Research*, 25(10):1033–1046, 2006.

[40] Nokleby S.B., Fisher R., Podhorodeski R.P., and Firmani F. Force capabilities of redundantly actuated parallel manipulators. *Mechanism and Machine Theory*, 40:578–599, 2005.

[41] Yi B.-J. and Freeman R.A. Simultaneous stiffness generation and internal load distribution in redundantly actuated mechanisms. In *IEEE International Conference of Advanced Robotics*, pages 802–807, 1991.

[42] Kock S. and Schumacher W. A parallel x-y manipulator with actuation redundancy for high-speed and active stiffness applications. In *IEEE International Conference on Robotics and Automation*, pages 2295–2300, Leuven, Belgium, 1998.

[43] Chakarov D. Study of the antagonistic stiffness of parallel manipulators with actuation redundancy. *Mechanism and Machine Theory*, 39:583–601, 2003.

[44] Notash L. and Huang L. On the design of fault tolerant parallel manipulators. *Mechanism and Machine Theory*, 38:85–101, 2003.

[45] Chen W., Chen I.-M., Chen P.C.Y., Yang G., and Kang I.-G. Fault-tolerant control for redundant robot based on joint torque redistribution. In *Fifth International Conference on Control, Automation, Robotics and Vision*, pages 1325–1329, Mandarin Singapore, Singapore, December 8-11, 1998.

[46] Zheng Y.F. and Luh J.Y.S. Optimal load distribution for two industrial robots handling a single object. In *IEEE International Conference on Robotics and Automation*, pages 344–349, Philadelphia, Pennsylvania, USA, April 24-29, 1988.

[47] Kerr D.R., Griffis M., Sanger D.J., and Duffy J. Redundant grasps, redundant manipulators, and their dual relationship. *Journal of Robotic Systems*, 9(7):973–1000, 1992.

[48] Lee J.H., Yi B.-J., Oh S.-R., and Suh I.H. Optimal design and development of a five-bar finger with redundant actuation. *Mechatronics*, 11:27–42, 2001.

[49] Ji Z. and Song Ph. Design of a reconfigurable platform manipulator. *Journal of Robotic Systems*, 15(6):341–346, 1998.

[50] Chen I-M., Yang G., and Kang I.-G. Numerical inverse kinematics for modular reconfigurable robots. *Journal of Robotic Systems*, 16(4):213–225, 1999.

[51] Yoon J. and Jeha R. Reconfigurability of a parallel manipulator: a case study. In *Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 94–97, Quebec City, Quebec, Canada, October 3-4, 2002.

[52] Fisher R., Podhorodeski R.P., and Nokleby S.B. Design of a reconfigurable planar parallel manipulator. *Journal of Robotic Systems*, 21(12):665–675, 2004.

[53] Firmani F. and Podhorodeski R.P. Force-unconstrained poses for a redundantly-actuated planar parallel manipulator. *Mechanism and Machine Theory*, 39:459–476, 2004.

[54] Krut S., Company O., and Pierrot F. Velocity performance indexes for parallel manipulators with actuation redundancy. In *Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 46–56, Quebec City, Quebec, Canada, October 3-4, 2002.

[55] Ganovski L., Fisette P., and Samin J.-C. Modeling of overactuated closed-loop mechanisms with singularities: simulation and control. In *ASME DETC and Computers and Information in Engineering Conference*, pages 215–223, Pittsburgh, Pennsylvania, USA, September 914, 2001.

[56] Dubowsky S. and Freudenstein F. Dynamic analysis of mechanical systems with clearances. *ASME Journal of Engineering for Industry*, 93:305–316, 1971.

[57] Sarkar N., Ellis R., and Moore Th. Backlash detection in geared mechanisms: modeling, simulation and experimentation. *Mechanical Systems and Signal Processing*, 11(3):391–408, 1997.

[58] Bauchau O. and Rodriguez J. Modeling of unilateral contact conditions with application to aerospace systems involving backlash, freeplay and friction. *Mechanics Research Communications*, 28(5):571–599, 2001.

[59] Garcia Orden J.C. Analysis of joint clearances in multibody systems. *Multibody System Dynamics*, 13:401–420, 2005.

[60] Schwab A.L., Meijaard J.P., and Meijers P. A comparison of revolute joint clearance models in the dynamic analysis of rigid and elastic mechanical systems. *Mechanism and Machine Theory*, 37:895–913, 2002.

[61] Zhu J. and Ting K.-L. Uncertainty analysis of planar and spatial robots with joint clearances. *Mechanism and Machine Theory*, 35:1239–1256, 2000.

[62] Parenti-Castelli V. and Venanzi S. On the joint clearance effects in serial and parallel manipulators. In *Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 215–223, Quebec City, Quebec, Canada, October 3-4, 2002.

[63] Chen D.-Z. and Yao K.-L. Drive train design of redundant-drive backlash-free robotic mechanisms. *Mechanism and Machine Theory*, 35:1269–1285, 2000.

[64] Müller A. Control of simply-redundant full-actuated parallel manipulators. In *ECCOMAS Thematic Conference MULTIBODY DYNAMICS 2005*, pages 215–223, Madrid, Spain, June 2124, 2005.

[65] Buchanan T., Lloyd D., Manal K., and Besier T. Neuromusculoskeletal modeling: estimation of muscle forces and joint moments and movements from measurements of neural command. *Journal of Applied Biomechanics*, 20:367–395, 2004.

[66] Raison M., Detrembleur C., Fisette P., and Samin J.C. Contribution to the quantification of muscle efforts: application to the human forearm flexion and extension. In *ECCOMAS Thematic Conference MULTIBODY DYNAMICS 2007*, pages 215–223, Milano, Italy, June 2528, 2007.

[67] Groote F., Pipeleers G., Demeulenaere B., Jonckers I., Spaepen P., Swevers J., and De Schutter J. A convex optimization approach to dynamic musculoskeletal analysis. In *National Congress of Theoretical and Applied Mechanics*, pages 232–246, Mons, Belgium, May , 2006.

[68] Yamaguchi G., Moran D., and Si J. A computationally efficient method for solving the redundant problem in biomechanics. *Journal of Biomechanics*, 28(8):999–1005, 1995.

[69] Nakamura Y. and Ghodoussi M. Dynamic computation of closed-link robot mechanisms with nonredundant and redundant actuators. *IEEE Transactions on Robotics and Automation*, 5(3):294–302, 1989.

[70] Zheng Y.F. and Luh J.Y.S. Coordination of two redundant manipulators. In *IEEE International Conference on Robotics and Automation*, pages 425–430, Scottsdale, Arizona, USA, May 14-19, 1989.

[71] Nahon M. and Angeles J. Real-time force optimization in parallel kinematic chains under inequality constraints. In *IEEE International Conference on Robotics and Automation*, pages 2198–2203, Sacramento, California, USA, 1991.

[72] Buttolo P. and Hannaford B. Advantages of actuation redundancy for the design of haptic displays. In *ASME International Mechanical Engineering Congress and Exposition - Part 2*, pages 623–630, San Francisco, California, USA, November 12-17, 1995.

[73] Merlet J.-P. Redundant parallel manipulators. *Laboratory Robotics and Automation*, 8(1):17–24, 1996.

[74] Kim H.S. and Choi Y.J. The kinetostatic capability analysis of robotic manipulators. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1241–1246, Kyongju, South Korea, 1999.

[75] Lee S.H., Yi B.-J., Kim S.H., and Kwak Y.K. Control of impact disturbance by redundantly actuated mechanisms. In *IEEE International Conference on Robotics and Automation*, pages 3734–3741, Seoul, South Korea, May 21-26, 2001.

[76] Cheng H., Liu G.F., Yiu Y.K., Xiong Z.H., and Li Z.X. Advantages and dynamics of parallel manipulators with redundant actuation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 171–176, Maui, Hawaii, USA, 2001.

[77] Dasgupta B. Redundancy resolution schemes for parallel manipulators with force redundancy. In *Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 257–260, Quebec City, Quebec, Canada, October 3-4, 2002.

[78] Ding H., Li Y.F., and Tso S.K. Dynamic optimization of redundant manipulators in worst case using recurrent neural networks. *Mechanism and Machine Theory*, 35:55–70, 2000.

[79] Zhang Y. Inverse-free computation for infinity-norm torque minimization of robot manipulators. *Mechatronics*, 16:177–184, 2006.

[80] Gorla B. and Renaud M. *Modèles de Robots Manipulateurs, applictaion à leur commande*. Toulouse: Cepadues-Editions, 1984.

[81] Luh J.Y., Walker N.W., and Paul R.P. On-line computational scheme for mechanical manipulators. *Journal of Dynamic Systems, Measurement and Control*, 102:69–76, 1980.

[82] Petzold L. Differential/algebraic equations are not ode's. *SIAM J. Sci. Stat.Comput.*, 3(3):367–384, 1982.

[83] Baumgarte J. Stabilization of constraints and integrals of motion. *Computer Methods in Applied Mechanics and Engineering*, 1:1–16, 1972.

[84] Petzold L. Methods and softwares for differential/algebraic systems. In *NATO ASI Series, Series F: Computer and Systems Sciences, Vol. 69*, pages 127–140, Berlin, 1989.

[85] Eichberger A., Fuhrer C., and Schwertassek R. The benefits of parallel multibody simulation and its application to vehicle dynamics. In *Advanced Multibody System Dynamics : Simulation and Software Tools*, pages 107–126, Dordrecht, the Netherlands, 1993.

[86] Fuhrer C. and Leimkuhler B. A new class of generalized inverses for the solution of discretized euler-lagrange equations. In *NATO ASI Series, Series F: Computer and Systems Sciences, Vol. 69*, pages 143–154, Berlin, 1989.

[87] Penrose R. A generalized inverse for matrices. In *Proceedings of the Cambridge Philosophical Society*, volume 51, pages 406–413, 1955.

[88] Deo A.S. and Walker I.D. Minimum-effort inverse kinematics for redundant manipulators. *IEEE Transactions on Robotics and Automation*, 3:767–775, 1997.

[89] Shim I.C. and Y.S. Yoon. Stabilized minimum infinity-norm torque for redundant manipulators. *Robotica*, 16:193–205, 1998.

[90] Dombre E. and Khalil W. *Modélisation et commande des robots.* Hermes, Paris, 1988.

[91] Bélanger P.R. *Control Engineering: A Modern Approach.* Saunders College Publishing, 1995.

[92] Ogata K. *Modern Control Engineering (3rd edition).* Prentice Hall, 1996.

[93] Nakamura Y. and Hanafusa H. Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of Dynamic Systems, Measurement and Control*, 108:163–171, 1986.

[94] Hanon M. A comparison of methods for the control of redundantly-actuated robotic systems. *Journal of Intelligent and Robotic Systems*, 14:3–21, 1995.

[95] Liu G.F., Wu Y.L., Wu X.Z., Kuen Y.Y., and Li Z.X. Analysis and control of redundant parallel manipulators. In *IEEE International Conference on Robotics and Automation*, pages 3748–3754, Seoul, South Korea, May 21-26, 2001.

[96] Cheng H., Yiu Y.K., and Li Z.X. Dynamics and control of redundantly actuated parallel manipulators. *IEEE Transactions on Mechatronics*, 8(4):483–491, 2003.

[97] Müller A. Internal preload control of redundantly actuated parallel manipulators. In *International Conference on Robotics and Automation*, pages 948–953, Barcelona, Spain, April, 2005.

[98] Hui S., Xu-Zhong W., Guan-Feng L., and Xe-Ziang L. Hybrid position/force adaptive control of redundantly actuated parallel manipulators. *Acta Automatica Sinica*, 29(4):567–572, 2003.

[99] Raibert M.H. and Craig J.J. Hybrid position/force control of manipulators. *Transactions of the ASME Journal of Dynamic Systems, Measurement and Control*, 103:126–133, 1981.

[100] Lipkin H. and Duffy J. Hybrid twist and wrench control for a robotic manipulator. *Transactions of ASME*, 110:138–144, 1988.

[101] Bruyninckx H. and De Schutter J. Specification of force-controlled actions in the task frame formalism: A survey. *IEEE Transactions on Robotics and Automation*, 12(5):581–589, 1996.

[102] De Schutter J., Torfs D., Dutre S., and Bruyninckx H. Invariant hybrid position/force control of a velocity controlled robot with compliant end effector using modal decoupling. *International Journal of Robotics Research*, 16(3):340–356, 1997.

[103] Salisbury J.K. Active stiffness control of a manipulator in cartesian coordinates. In *Proceedings of the 19th IEEE Conference on Decision and Control*, pages 95–100, Albuquerque, Canada, December, 1980.

[104] De Schutter J. Hybrid motion/internal reaction force control of redundantly actuated mechanisms. Technical Report 06RP005, Department of Mechanical Engineering, K.U.Leuven, April 2006.

[105] Ou L., Zhang W., and Gu D. Nominal and robust stability regions of optimization-based pid controllers. *Transactions of ISA - The Instrumentation, Systems, and Automation Society*, 45(3):361–371, 2006.

[106] Herreros A., Baeyens E., and Perán J.R. Design of pid-type controllers using multiobjective genetic algorithms. *Transactions of ISA - The Instrumentation, Systems, and Automation Society*, 41:457–472, 2002.

[107] Ghorbel F. Modeling and pd control of closed chain mechanical systems. In *Proceedings of the 34th IEEE Conference on Decision and Control*, pages 540–542, New Orleans, Louisiana, USA, December 13-15, 1995.

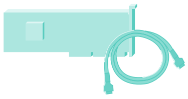
[108] Nyquist H. Certain topics in telegraph transmission theory. In *Proceedings of the IEEE, Vol. 90(2) (Reprinted from Transactions of the A. I. E. E., pp. 617-644, Feb. 1928)*, pages 143–154, Berlin, February 2002.

[109] Kotelnikov V. On the transmission capacity of the ether and of cables in electrical communications. In *Proceedings of the first All-Union Conference on the technological reconstruction of the communications sector and the development of low-current engineering.*, Moscow, 1933.

[110] Fisette P., Péterkenne J.M., Vaneghem B., and Samin J.C. A multibody loop constraints approach for modelling cam/follower devices. *Nonlinear Dynamics*, 22:335–359, 2000.

# Annexes

## DS1102 DSP Controller Board



### DS1102 DSP Controller Board

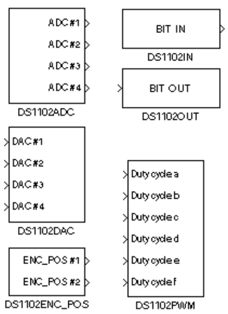
# Intelligent Single-Board Solution

- **Cost-effective single-board solution**
- **Ideally suited to rapid control prototyping**
- **TI's C31 DSP featuring 60 MFlops**
- **Comprehensive suite of on-board I/O**
- **Fully programmable from the Simulink block diagram environment**

**Software Support**

- DS1102 Real-Time Library (included)
- Graphical Hardware Management (included)
- Programming from Simulink with RTI (p. 64)
- Texas Instruments C Compiler (p. 78)
- Debugger (p. 79)
- Experiment Control with ControlDesk (p. 82) and MLIB/MTRACE (p. 98)

'All-in-one' is the motto of the DS1102 DSP Controller Board. It combines the computing performance of a TI TMS320C31 floating-point DSP with a set of I/O modules frequently required in control systems. It is therefore ideally suited to cost-sensitive applications that have a limited number of inputs and outputs but still require fast computation. Typical application fields are drives, automotives and robotics control, but customers from many other areas can also benefit from this powerful single-board solution.




Real-Time Interface provides Simulink blocks for convenient configuration of A/D, D/A, digital I/O lines, incremental encoder interface and PWM generation. The PWM dialog lets you enter the desired PWM frequency.

## Technical Details

| | |
|---|---|
| **Processor** | ■ TI's TMS320C31 floating-point DSP<br>■ 60 MHz clock rate, 33.3 ns cycle time<br>■ 4 external interrupts |
| **Memory** | ■ 128 K x 32-bit RAM, zero wait states<br>■ 2 K x 32-bit on-chip RAM |





Figure 17: Technical specifications of the dSpace DS1102 controller board.

## DS1102 DSP Controller Board

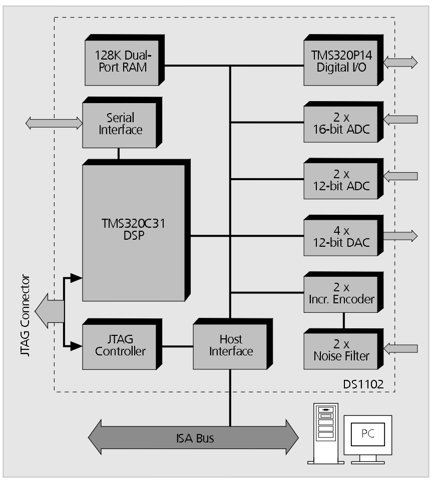
**Analog Input**
- 2 parallel 16-bit channels, 4 µs conversion time *)
- 2 parallel 12-bit channels, 1.25 µs conversion time *)
- Simultaneous sample & hold
- ±10 V input voltage range
- > 80 dB (16 bit) / 65 dB (12 bit) signal-to-noise ratio

**Analog Output**
- 4 parallel 12-bit channels
- 4 µs typical settling time
- ±10 V output voltage range

**Digital I/O**
- Programmable digital-I/O subsystem based on TI's 25 MHz TMS320P14 DSP
- 16 digital I/O lines (bit-selectable)
- Capture/compare unit with 8 channels (2 in, 4 out, 2 in/out)
- PWM generation on up to 6 channels (40 ns resolution)
- User interrupt

**Incremental Encoder Interface**
- Fourfold pulse multiplication
- Two parallel input channels for two phase lines and one index line each
- 8.3 MHz max. count frequency
- Noise filter
- 24-bit position counter

**Physical Characteristics**
- Power supply 5 V, 1.5 A and ±12 V, 100 mA
- Requires half-length 16-bit ISA slot
- 62-pin female high-density Sub-D connector




| Channels in use | F/D [kHz] | D/F [kHz] | PWM [kHz] |
|---|---|---|---|
| 1 | 60 | 12 | 100 |
| 2 | 20 | 12 | 100 |
| 3 | 15 | 4 | 100 |
| 4 | 10 | 4 | 100 |
| 5 | — | — | 100 |
| 6 | — | — | 100 |

**Application Frequencies**

The table above shows the maximum frequencies of the included timing applications for different numbers of channels in use, assuming that only one type of application is running at a time.

Single-Board Hardware

*) Speed and timing specifications describe the capabilities of the hardware components and circuits of our products. Depending on the software complexity, the attainable overall performance figures can deviate significantly from the hardware specifications.

**Order Number**

DS1102 DSP Controller Board
- DS1102

Connector Panel (p. 128)
- CP1102

Combined Connector/LED Panel (p. 128)
- CLP1102



Figure 18: Technical specifications of the dSpace DS1102 controller board.

**FAULHABER**

## DC-Micromotors

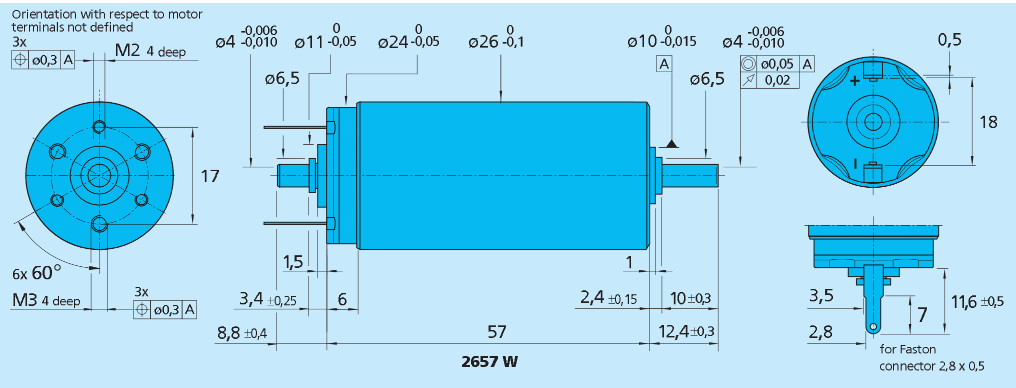
Graphite Commutation

## 44 mNm

**For combination with**
Gearheads:
26/1, 30/1
Encoders:
IE2, 5500, 5540

### Series 2657 ... CR

| | 2657 W | | 012 CR | 024 CR | 048 CR | |
|---|---|---|---|---|---|---|
| 1 | Nominal voltage | $U_N$ | 12 | 24 | 48 | Volt |
| 2 | Terminal resistance | $R$ | 0,71 | 2,84 | 12,50 | Ω |
| 3 | Output power | $P_{2\ max.}$ | 45,9 | 47,9 | 44,5 | W |
| 4 | Efficiency | $\eta_{max.}$ | 84 | 85 | 84 | % |
| 5 | No-load speed | $n_o$ | 6 300 | 6 400 | 6 400 | rpm |
| 6 | No-load current (with shaft ø 4,0 mm) | $I_o$ | 0,115 | 0,058 | 0,028 | A |
| 7 | Stall torque | $M_H$ | 278 | 286 | 265 | mNm |
| 8 | Friction torque | $M_R$ | 2 | 2 | 2 | mNm |
| 9 | Speed constant | $k_n$ | 552 | 274 | 136 | rpm/V |
| 10 | Back-EMF constant | $k_E$ | 1,81 | 3,65 | 7,37 | mV/rpm |
| 11 | Torque constant | $k_M$ | 17,3 | 34,8 | 70,4 | mNm/A |
| 12 | Current constant | $k_I$ | 0,058 | 0,029 | 0,014 | A/mNm |
| 13 | Slope of n-M curve | $\Delta n/\Delta M$ | 22,7 | 22,4 | 24,2 | rpm/mNm |
| 14 | Rotor inductance | $L$ | 95 | 380 | 1 550 | µH |
| 15 | Mechanical time constant | $\tau_m$ | 3,9 | 3,9 | 3,9 | ms |
| 16 | Rotor inertia | $J$ | 16 | 17 | 15 | $gcm^2$ |
| 17 | Angular acceleration | $\alpha_{max.}$ | 170 | 170 | 170 | $\cdot 10^3 rad/s^2$ |
| 18 | Thermal resistance | $R_{th1}$ / $R_{th2}$ | 1,9 / 9 | | | K/W |
| 19 | Thermal time constant | $\tau_{w1}$ / $\tau_{w2}$ | 10 / 580 | | | s |
| 20 | Operating temperature range: | | | | | |
| | – motor | | – 30 ... + 125 | | | °C |
| | – rotor, max. permissible | | + 155 | | | °C |
| 21 | Shaft bearings | | ball bearings, preloaded | | | |
| 22 | Shaft load max.: | | | | | |
| | – with shaft diameter | | 4,0 | | | mm |
| | – radial at 3000 rpm (3 mm from bearing) | | 20 | | | N |
| | – axial at 3000 rpm | | 2 | | | N |
| | – axial at standstill | | 20 | | | N |
| 23 | Shaft play: | | | | | |
| | – radial | ≤ | 0,015 | | | mm |
| | – axial | = | 0 | | | mm |
| 24 | Housing material | | steel, black coated | | | |
| 25 | Weight | | 156 | | | g |
| 26 | Direction of rotation | | clockwise, viewed from the front face | | | |
| **Recommended values** | | | | | | |
| 27 | Speed up to | $n_{e\ max.}$ | 6 000 | 6 000 | 6 000 | rpm |
| 28 | Torque up to | $M_{e\ max.}$ | 44 | 44 | 44 | mNm |
| 29 | Current up to (thermal limits) | $I_{e\ max.}$ | 3,10 | 1,54 | 0,73 | A |




For notes on technical data and lifetime performance refer to "Technical Information".

For options on DC-Micromotors refer to page 62.
Specifications subject to change without notice.
www.faulhaber.com

56

Figure 19: Technical specifications of the 2657 DC-motor series.

FAULHABER

## Planetary Gearheads

3,5 Nm

**For combination with**
DC-Micromotors:
2342, 2642, 2657
Brushless DC-Servomotors:
2444

### Series 26/1

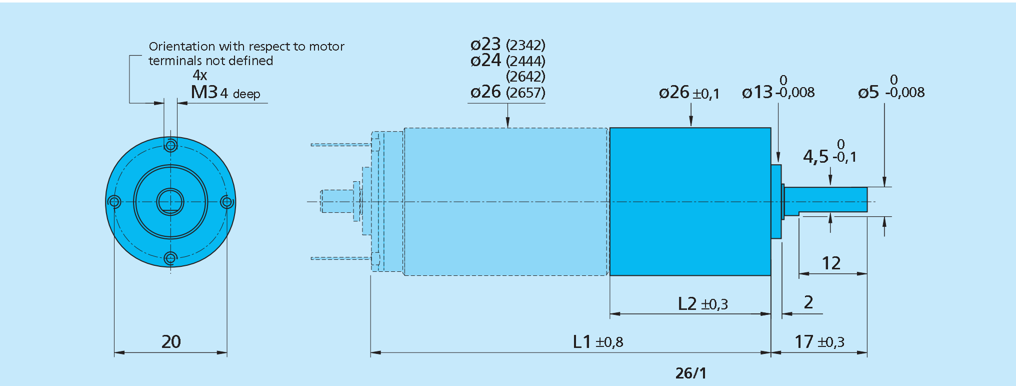
| | 26/1 |
|---|---|
| Housing material | inox steel |
| Geartrain material | steel [1] |
| Recommended max. input speed for: | |
| – for continuous operation | 4000 rpm |
| Backlash, typical, at no-load | ≤ 1° |
| Bearings on output shaft | preloaded ball bearings |
| Shaft load, max. | |
| – radial (10 mm from mounting face) | ≤ 150 N |
| – axial | ≤ 100 N |
| Shaft press fit force, max. | ≤ 150 N |
| Shaft play (on bearing output): | |
| – radial | ≤ 0,015 mm |
| – axial | ≤ 0,10 mm |
| Operating temperature range | – 30 ... + 100° C |

**Specifications**

| reduction ratio (nominal) | weight without motor | length without motor L2 | length with motor 2444 S L1 | length with motor 2342 S 2642 W L1 | length with motor 2657 W L1 | output torque continuous operation M max. | output torque intermittent operation M max. | direction of rotation (reversible) | efficiency |
|---|---|---|---|---|---|---|---|---|---|
| | g | mm | mm | mm | mm | Nm | Nm | | % |
| 3,71 :1 | 93 | 28,4 | 72,4 | 70,4 | 85,4 | 1,1 | 2,3 | = | 88 |
| 14 :1 | 116 | 36,4 | 80,4 | 78,4 | 93,4 | 0,3 (3,5) | 0,4 (4,5) | = | 80 |
| 43 :1 | 139 | 44,4 | 88,4 | 86,4 | 101,4 | 1,0 (3,5) | 1,2 (4,5) | = | 70 |
| 66 :1 | 139 | 44,4 | 88,4 | 86,4 | 101,4 | 1,5 (3,5) | 1,8 (4,5) | = | 70 |
| 134 :1 | 162 | 52,5 | 96,4 | 94,5 | 109,5 | 2,5 (3,5) | 3,5 (4,5) | = | 60 |
| 159 :1 | 162 | 52,5 | 96,4 | 94,5 | 109,5 | 3,5 (3,5) | 4,5 (4,5) | = | 60 |
| 246 :1 | 162 | 52,5 | 96,4 | 94,5 | 109,5 | 3,5 (3,5) | 4,5 (4,5) | = | 60 |
| 415 :1 | 185 | 60,5 | 104,4 | 102,5 | 117,5 | 3,5 (3,5) | 4,5 (4,5) | = | 55 |
| 592 :1 | 185 | 60,5 | 104,4 | 102,5 | 117,5 | 3,5 (3,5) | 4,5 (4,5) | = | 55 |
| 989 :1 | 185 | 60,5 | 104,4 | 102,5 | 117,5 | 3,5 (3,5) | 4,5 (4,5) | = | 55 |
| 1 526 :1 | 185 | 60,5 | 104,4 | 102,5 | 117,5 | 3,5 (3,5) | 4,5 (4,5) | = | 55 |

[1] Gearheads with ratio ≥ 14:1 have plastic gears in the input stage. For extended life performance, the gearheads are available with all steel gears and heavy duty lubricant as type 26/1 S.

The values for the torque rating indicated in parenthesis, are for gearheads, type 26/1 S with all steel gears.




For notes on technical data and lifetime performance refer to "Technical Information".

Specifications subject to change without notice.

115

www.faulhaber.com

Figure 20: Technical specifications of the 26 gearhead series.

FAULHABER

# Encoders

## Magnetic Encoders

**Features:**
64 to 512 Lines per revolution
2 Channels
Digital output

### Series IE2 – 512

| | | IE2 – 64 | IE2 – 128 | IE2 – 256 | IE2 – 512 | |
|---|---|---|---|---|---|---|
| Lines per revolution | N | 64 | 128 | 256 | 512 | |
| Signal output, square wave | | 2 | | | | channels |
| Supply voltage | $V_{DD}$ | 4,5 … 5,5 | | | | V DC |
| Current consumption, typical ($V_{DD}$ = 5 V DC) | $I_{DD}$ | typ. 6, max. 12 | | | | mA |
| Output current, max. 1) | $I_{OUT}$ | 5 | | | | mA |
| Pulse width | P | 180 ± 45 | | | | °e |
| Phase shift, channel A to B | Φ | 90 ± 45 | | | | °e |
| Signal rise/fall time, max. ($C_{LOAD}$ = 50 pF) | tr/tf | 0,1 / 0,1 | | | | µs |
| Frequency range 2), up to | f | 20 | 40 | 80 | 160 | kHz |
| Inertia of code disc | J | 0,09 | | | | gcm² |
| Operating temperature range | | – 25 … + 85 | | | | °C |

1) $V_{DD}$ = 5 V DC: Low logic level < 0,5 V, high logic level > 4,5 V: CMOS and TTL compatible
2) Velocity (rpm) = f (Hz) x 60/N

**Ordering information**

| Encoder | number of channels | lines per revolution | in combination with: |
|---|---|---|---|
| **IE2 – 64** | 2 | 64 | **DC-Micromotors series** |
| | | | **1336 … C,** |
| **IE2 – 128** | 2 | 128 | **1516 … SR, 1524 … SR,** |
| | | | **1717 … SR, 1724 … SR, 1727 … C,** |
| **IE2 – 256** | 2 | 256 | **2224 … SR, 2342 … CR,** |
| | | | **2642 … CR, 2657 … CR,** |
| **IE2 – 512** | 2 | 512 | **3242 … CR, 3257 … CR, 3863 … C** |
| | | | **brushless DC-Servomotors series** |
| | | | **1628 … B, 2036 … B, 2444 … B** |

**Features**

These incremental shaft encoders in combination with the FAULHABER DC-Micromotors and brushless DC-Servomotors are used for indication and control of both, shaft velocity and direction of rotation as well as for positioning.

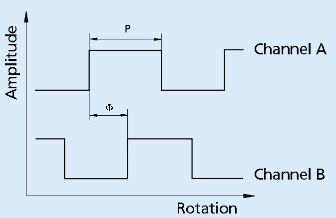
The encoder is integrated in the DC-Micromotors SR-Series and extends the overall length by only 1,4 mm and buildt-up option for DC-Micromotors and brushless DC-Servomotors.

Hybrid circuits with sensors and a low inertia magnetic disc provide two channels with 90° phase shift.

The supply voltage for the encoder and the DC-Micromotor as well as the two channel output signals are interfaced through a ribbon cable with connector.

Details for the DC-Micromotors and suitable reduction gearheads are on separate catalog pages.

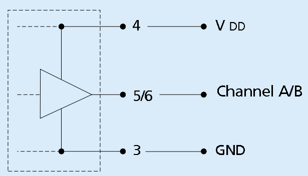
**Output signals / Circuit diagram / Connector information**



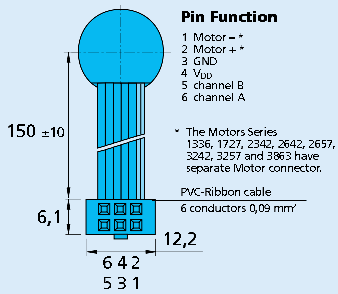

Admissible deviation of phase shift:

$$\Delta\Phi = \left| 90° - \frac{\Phi}{P} * 180° \right| \leq 45°$$

**Output signals**
with clockwise rotation as seen from the shaft end




**Output circuit**
Note: Motor terminal resistance increases by approx. 0,4 Ω




**Connector**
DIN-41651
grid 2,54 mm

For notes on technical data and lifetime performance refer to "Technical Information".

Specifications subject to change without notice.

127

www.faulhaber.com

Figure 21: Technical specifications of the Series IE2-512 incremental encoders.

LEM

# Current Transducer LTS 15-NP

$I_{PN}$ = 5 - 7.5 - 15 A

For the electronic measurement of currents : DC, AC, pulsed, mixed, with a galvanic isolation between the primary circuit (high power) and the secondary circuit (electronic circuit).

5 YEARS WARRANTY

## Electrical data

| | | | |
|---|---|---|---|
| $I_{PN}$ | Primary nominal r.m.s. current | 15 | At |
| $I_P$ | Primary current, measuring range | 0 .. ± 48 | At |
| $\hat{I}_{PDC}$ | Overload capability | 250 | A |
| $V_{OUT}$ | Analog output voltage @$I_P$ | 2.5 ± (0.625·$I_P$/$I_{PN}$) | V |
| | $I_P$ = 0 | 2.5 [1] | V |
| $N_S$ | Number of secondary turns (± 0.1 %) | 2000 | |
| $R_L$ | Load resistance | ≥ 2 | kΩ |
| $R_{IM}$ | Internal measuring resistance (± 0.5 %) | 83.33 | Ω |
| $TCR_{IM}$ | Thermal drift of $R_{IM}$ | < 50 | ppm/K |
| $V_C$ | Supply voltage (± 5 %) | 5 | V |
| $I_C$ | Current consumption @ $V_C$ = 5 V Typ | 28+$I_S$[2]+($V_{OUT}$/$R_L$) | mA |
| $V_d$ | R.m.s. voltage for AC isolation test, 50/60 Hz, 1 mn | 3 | kV |
| $V_e$ | R.m.s. voltage for partial discharge extinction @ 10 pC | > 1.5 | kV |
| $\hat{V}_w$ | Impulse withstand voltage 1.2/50 µs | > 8 | kV |

## Accuracy - Dynamic performance data

| | | | Typ | Max | |
|---|---|---|---|---|---|
| X | Accuracy @ $I_{PN}$, $T_A$ = 25°C | | | ± 0.2 | % |
| | Accuracy with $R_{IM}$ @ $I_{PN}$, $T_A$ = 25°C | | | ± 0.7 | % |
| $\varepsilon_L$ | Linearity error | | | < 0.1 | % |
| $TCV_{OUT}$ | Thermal drift of $V_{OUT}$ @ $I_P$ = 0 | - 10°C .. + 85°C | 65 | 120 | ppm/K |
| | | - 40°C .. - 10°C | | 170 | ppm/K |
| $TC\varepsilon_G$ | Thermal drift of the gain | - 40°C .. + 85°C | | 50 [3] | ppm/K |
| $V_{OM}$ | Residual voltage @ $I_P$ = 0, after an overload of | 3 x $I_{PN}$ | | ± 0.5 | mV |
| | | 5 x $I_{PN}$ | | ± 2.0 | mV |
| | | 10 x $I_{PN}$ | | ± 2.0 | mV |
| $t_{ra}$ | Reaction time @ 10 % of $I_{PN}$ | | | < 100 | ns |
| $t_r$ | Response time @ 90 % of $I_{PN}$ | | | < 400 | ns |
| di/dt | di/dt accurately followed | | | > 35 | A/µs |
| f | Frequency bandwidth (0 .. - 0.5 dB) | | | DC .. 100 | kHz |
| | (- 0.5 .. 1 dB) | | | DC .. 200 | kHz |

## General data

| | | | |
|---|---|---|---|
| $T_A$ | Ambient operating temperature | - 40 .. + 85 | °C |
| $T_S$ | Ambient storage temperature | - 40 .. + 100 | °C |
| | Insulating material group | III a | |
| m | Mass | 10 | g |
| | Standards | EN 50178 (97.10.01) | |
| | | IEC 60950-1(01.10.26) | |

Notes : [1] Absolute value @ $T_A$ = 25°C, 2.475 < $V_{OUT}$ < 2.525
[2] Please see the operation principle on the other side
[3] Only due to $TCR_{IM}$

### Features

- Closed loop (compensated) multi-range current transducer using the Hall effect
- Unipolar voltage supply
- Compact design for PCB mounting
- Insulated plastic case recognized according to UL 94-V0
- Incorporated measuring resistance
- Extended measuring range.

### Advantages

- Excellent accuracy
- Very good linearity
- Very low temperature drift
- Optimized response time
- Wide frequency bandwidth
- No insertion losses
- High immunity to external interference
- Current overload capability.

### Applications

- AC variable speed drives and servo motor drives
- Static converters for DC motor drives
- Battery supplied applications
- Uninterruptible Power Supplies (UPS)
- Switched Mode Power Supplies (SMPS)
- Power supplies for welding applications.

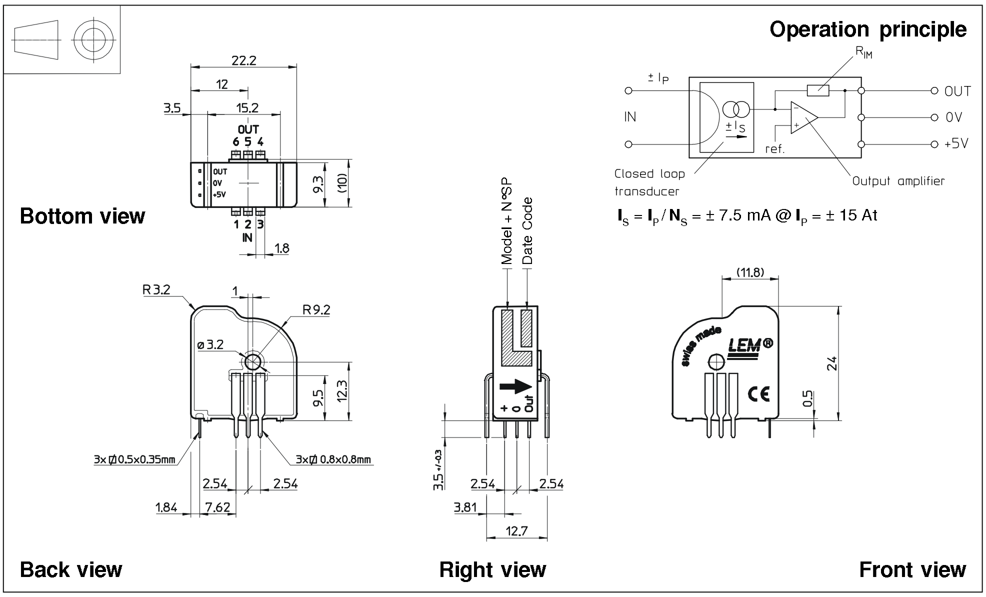


040319/12

LEM Components www.lem.com

Figure 22: Technical specifications of the current transducers used.

**Dimensions LTS 15-NP** (in mm. 1 mm = 0.0394 inch)




| Number of primary turns | Primary nominal r.m.s. current $I_{PN}$ [A] | Nominal output voltage $V_{OUT}$ [V] | Primary resistance $R_P$ [mΩ] | Primary insertion inductance $L_P$ [μH] | Recommended connections |
|---|---|---|---|---|---|
| 1 | ± 15 | 2.5 ± 0.625 | 0.18 | 0.013 | 6 5 4 OUT / IN 1 2 3 |
| 2 | ± 7.5 | 2.5 ± 0.625 | 0.81 | 0.05 | 6 5 4 OUT / IN 1 2 3 |
| 3 | ± 5 | 2.5 ± 0.625 | 1.62 | 0.12 | 6 5 4 OUT / IN 1 2 3 |

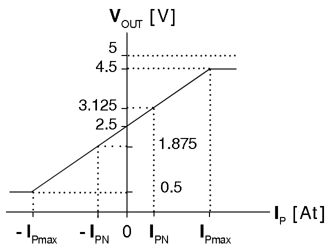
**Mechanical characteristics**

- General tolerance ± 0.2 mm
- Fastening & connection of primary 6 pins 0.8 x 0.8 mm
  Recommended PCB hole 1.3 mm
- Fastening & connection of secondary 3 pins 0.5 x 0.35 mm
  Recommended PCB hole 0.8 mm
- Additional primary through-hole ∅ 3.2 mm

**Remark**

- $V_{OUT}$ is positive when $I_P$ flows from terminals 1, 2, 3 to terminals 6, 5, 4.

**Output Voltage - Primary Current**




LEM reserves the right to carry out modifications on its transducers, in order to improve them, without previous notice.

Figure 23: Technical specifications of the current transducers used.

MOTOROLA

# DECADE COUNTER; DIVIDE-BY-TWELVE COUNTER; 4-BIT BINARY COUNTER

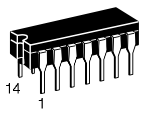
**SN54/74LS90**
**SN54/74LS92**
**SN54/74LS93**

**DECADE COUNTER; DIVIDE-BY-TWELVE COUNTER; 4-BIT BINARY COUNTER**

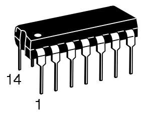
**LOW POWER SCHOTTKY**

The SN54/74LS90, SN54/74LS92 and SN54/74LS93 are high-speed 4-bit ripple type counters partitioned into two sections. Each counter has a divide-by-two section and either a divide-by-five (LS90), divide-by-six (LS92) or divide-by-eight (LS93) section which are triggered by a HIGH-to-LOW transition on the clock inputs. Each section can be used separately or tied together (Q to CP) to form BCD, bi-quinary, modulo-12, or modulo-16 counters. All of the counters have a 2-input gated Master Reset (Clear), and the LS90 also has a 2-input gated Master Set (Preset 9).

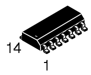
- Low Power Consumption . . . Typically 45 mW
- High Count Rates . . . Typically 42 MHz
- Choice of Counting Modes . . . BCD, Bi-Quinary, Divide-by-Twelve, Binary
- Input Clamp Diodes Limit High Speed Termination Effects




**J SUFFIX**
CERAMIC
CASE 632-08




**N SUFFIX**
PLASTIC
CASE 646-06




**D SUFFIX**
SOIC
CASE 751A-02

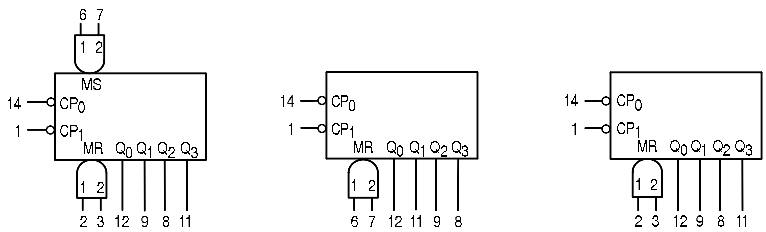
**ORDERING INFORMATION**

| | |
|---|---|
| SN54LSXXJ | Ceramic |
| SN74LSXXN | Plastic |
| SN74LSXXD | SOIC |

| PIN NAMES | | LOADING (Note a) HIGH | LOW |
|---|---|---|---|
| $\overline{CP_0}$ | Clock (Active LOW going edge) Input to ÷2 Section | 0.5 U.L. | 1.5 U.L. |
| $\overline{CP_1}$ | Clock (Active LOW going edge) Input to ÷5 Section (LS90), ÷6 Section (LS92) | 0.5 U.L. | 2.0 U.L. |
| $\overline{CP_1}$ | Clock (Active LOW going edge) Input to ÷8 Section (LS93) | 0.5 U.L. | 1.0 U.L. |
| $MR_1$, $MR_2$ | Master Reset (Clear) Inputs | 0.5 U.L. | 0.25 U.L. |
| $MS_1$, $MS_2$ | Master Set (Preset-9, LS90) Inputs | 0.5 U.L. | 0.25 U.L. |
| $Q_0$ | Output from ÷2 Section (Notes b & c) | 10 U.L. | 5 (2.5) U.L. |
| $Q_1$, $Q_2$, $Q_3$ | Outputs from ÷5 (LS90), ÷6 (LS92), ÷8 (LS93) Sections (Note b) | 10 U.L. | 5 (2.5) U.L. |

NOTES:
a. 1 TTL Unit Load (U.L.) = 40 μA HIGH/1.6 mA LOW.
b. The Output LOW drive factor is 2.5 U.L. for Military, (54) and 5 U.L. for commercial (74) Temperature Ranges.
c. The $Q_0$ Outputs are guaranteed to drive the full fan-out plus the $\overline{CP_1}$ input of the device.
d. To insure proper operation the rise ($t_r$) and fall time ($t_f$) of the clock must be less than 100 ns.

**LOGIC SYMBOL**




| LS90 | LS92 | LS93 |
|---|---|---|
| $V_{CC}$ = PIN 5 | $V_{CC}$ = PIN 5 | $V_{CC}$ = PIN 5 |
| GND = PIN 10 | GND = PIN 10 | GND = PIN 10 |
| NC = PINS 4, 13 | NC = PINS 2, 3, 4, 13 | NC = PIN 4, 6, 7, 13 |

FAST AND LS TTL DATA

5-1

Figure 24: Technical specifications of the 74LS series binary counters.

## SN54/74LS90 • SN54/74LS92 • SN54/74LS93

### FUNCTIONAL DESCRIPTION

The LS90, LS92, and LS93 are 4-bit ripple type Decade, Divide-By-Twelve, and Binary Counters respectively. Each device consists of four master/slave flip-flops which are internally connected to provide a divide-by-two section and a divide-by-five (LS90), divide-by-six (LS92), or divide-by-eight (LS93) section. Each section has a separate clock input which initiates state changes of the counter on the HIGH-to-LOW clock transition. State changes of the Q outputs do not occur simultaneously because of internal ripple delays. Therefore, decoded output signals are subject to decoding spikes and should not be used for clocks or strobes. The $Q_0$ output of each device is designed and specified to drive the rated fan-out plus the $CP_1$ input of the device.

A gated AND asynchronous Master Reset ($MR_1 \bullet MR_2$) is provided on all counters which overrides and clocks and resets (clears) all the flip-flops. A gated AND asynchronous Master Set ($MS_1 \bullet MS_2$) is provided on the LS90 which overrides the clocks and the MR inputs and sets the outputs to nine (HLLH).

Since the output from the divide-by-two section is not internally connected to the succeeding stages, the devices may be operated in various counting modes.

**LS90**

A. BCD Decade (8421) Counter — The $\overline{CP_1}$ input must be externally connected to the $Q_0$ output. The $CP_0$ input receives the incoming count and a BCD count sequence is produced.

B. Symmetrical Bi-quinary Divide-By-Ten Counter — The $Q_3$ output must be externally connected to the $CP_0$ input. The input count is then applied to the $CP_1$ input and a divide-by-ten square wave is obtained at output $Q_0$.

C. Divide-By-Two and Divide-By-Five Counter — No external interconnections are required. The first flip-flop is used as a binary element for the divide-by-two function ($CP_0$ as the input and $Q_0$ as the output). The $CP_1$ input is used to obtain binary divide-by-five operation at the $Q_3$ output.

**LS92**

A. Modulo 12, Divide-By-Twelve Counter — The $\overline{CP_1}$ input must be externally connected to the $Q_0$ output. The $CP_0$ input receives the incoming count and $Q_3$ produces a symmetrical divide-by-twelve square wave output.

B. Divide-By-Two and Divide-By-Six Counter —No external interconnections are required. The first flip-flop is used as a binary element for the divide-by-two function. The $CP_1$ input is used to obtain divide-by-three operation at the $Q_1$ and $Q_2$ outputs and divide-by-six operation at the $Q_3$ output.

**LS93**

A. 4-Bit Ripple Counter — The output $Q_0$ must be externally connected to input $CP_1$. The input count pulses are applied to input $CP_0$. Simultaneous divisions of 2, 4, 8, and 16 are performed at the $Q_0$, $Q_1$, $Q_2$, and $Q_3$ outputs as shown in the truth table.

B. 3-Bit Ripple Counter— The input count pulses are applied to input $CP_1$. Simultaneous frequency divisions of 2, 4, and 8 are available at the $Q_1$, $Q_2$, and $Q_3$ outputs. Independent use of the first flip-flop is available if the reset function coincides with reset of the 3-bit ripple-through counter.

FAST AND LS TTL DATA

5-3

Figure 25: Technical specifications of the 74LS series binary counters.

## SN54/74LS90 • SN54/74LS92 • SN54/74LS93

**AC CHARACTERISTICS** ($T_A$ = 25°C, $V_{CC}$ = 5.0 V, $C_L$ = 15 pF)

| Symbol | Parameter | Limits | | | | | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | LS90 | | | LS92 | | | LS93 | | | |
| | | Min | Typ | Max | Min | Typ | Max | Min | Typ | Max | |
| $f_{MAX}$ | $\overline{CP}_0$ Input Clock Frequency | 32 | | | 32 | | | 32 | | | MHz |
| $f_{MAX}$ | $\overline{CP}_1$ Input Clock Frequency | 16 | | | 16 | | | 16 | | | MHz |
| $t_{PLH}$ <br> $t_{PHL}$ | Propagation Delay, $\overline{CP}_0$ Input to $Q_0$ Output | | 10 <br> 12 | 16 <br> 18 | | 10 <br> 12 | 16 <br> 18 | | 10 <br> 12 | 16 <br> 18 | ns |
| $t_{PLH}$ <br> $t_{PHL}$ | $\overline{CP}_0$ Input to $Q_3$ Output | | 32 <br> 34 | 48 <br> 50 | | 32 <br> 34 | 48 <br> 50 | | 46 <br> 46 | 70 <br> 70 | ns |
| $t_{PLH}$ <br> $t_{PHL}$ | $\overline{CP}_1$ Input to $Q_1$ Output | | 10 <br> 14 | 16 <br> 21 | | 10 <br> 14 | 16 <br> 21 | | 10 <br> 14 | 16 <br> 21 | ns |
| $t_{PLH}$ <br> $t_{PHL}$ | $\overline{CP}_1$ Input to $Q_2$ Output | | 21 <br> 23 | 32 <br> 35 | | 10 <br> 14 | 16 <br> 21 | | 21 <br> 23 | 32 <br> 35 | ns |
| $t_{PLH}$ <br> $t_{PHL}$ | $\overline{CP}_1$ Input to $Q_3$ Output | | 21 <br> 23 | 32 <br> 35 | | 21 <br> 23 | 32 <br> 35 | | 34 <br> 34 | 51 <br> 51 | ns |
| $t_{PLH}$ | MS Input to $Q_0$ and $Q_3$ Outputs | | 20 | 30 | | | | | | | ns |
| $t_{PHL}$ | MS Input to $Q_1$ and $Q_2$ Outputs | | 26 | 40 | | | | | | | ns |
| $t_{PHL}$ | MR Input to Any Output | | 26 | 40 | | 26 | 40 | | 26 | 40 | ns |

**AC SETUP REQUIREMENTS** ($T_A$ = 25°C, $V_{CC}$ = 5.0 V)

| Symbol | Parameter | Limits | | | | | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | LS90 | | LS92 | | LS93 | | |
| | | Min | Max | Min | Max | Min | Max | |
| $t_W$ | $\overline{CP}_0$ Pulse Width | 15 | | 15 | | 15 | | ns |
| $t_W$ | $\overline{CP}_1$ Pulse Width | 30 | | 30 | | 30 | | ns |
| $t_W$ | MS Pulse Width | 15 | | | | | | ns |
| $t_W$ | MR Pulse Width | 15 | | 15 | | 15 | | ns |
| $t_{rec}$ | Recovery Time MR to $\overline{CP}$ | 25 | | 25 | | 25 | | ns |

RECOVERY TIME ($t_{rec}$) is defined as the minimum time required between the end of the reset pulse and the clock transition from HIGH-to-LOW in order to recognize and transfer HIGH data to the Q outputs

### AC WAVEFORMS

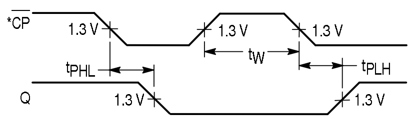



**Figure 1**

*The number of Clock Pulses required between the $t_{PHL}$ and $t_{PLH}$ measurements can be determined from the appropriate Truth Tables.

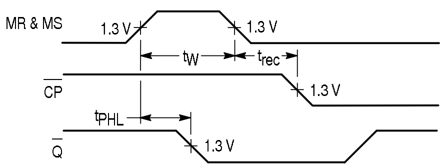



**Figure 2**

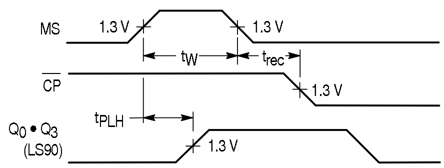



**Figure 3**

FAST AND LS TTL DATA

5-6

Figure 26: Technical specifications of the 74LS series binary counters.

# LSI/CSI LS7083/7084

LSI Computer Systems, Inc. 1235 Walt Whitman Road, Melville, NY 11747 (631) 271-0400 FAX (631) 271-0405

ISO 9001 UL REGISTERED FIRM A3800

## QUADRATURE CLOCK CONVERTER

October 2000

**FEATURES:**
- x1 and x4 mode selection
- Up to 16 MHz output clock frequency
- Programmable output clock pulse width
- On-chip filtering of inputs for optical or magnetic encoder applications.
- TTL and CMOS compatible I/Os
- +4.5V to +10.0V operation ($V_{DD}$-$V_{SS}$)
- LS7083, LS7084 (DIP)
  LS7083-S, LS7084-S (SOIC) - See Figure 1

**DESCRIPTION:**
The LS7083 and LS7084 are monolithic CMOS silicon gate quadrature clock converters. Quadrature clocks derived from optical or magnetic encoders, when applied to the A and B inputs of the LS7083/LS7084, are converted to strings of Up Clocks and Down Clocks (LS7083) or to a Clock and an Up/Down direction control (LS7084). These outputs can be interfaced directly with standard Up/Down counters for direction and position sensing of the encoder.

**INPUT/OUTPUT DESCRIPTION:**

**RBIAS** (Pin 1)
Input for external component connection. A resistor connected between this input and $V_{SS}$ adjusts the output clock pulse width (Tow). For proper operation, the output clock pulse width must be less than or equal to the A,B pulse separation ($T_{OW} \leq T_{PS}$).

**$V_{DD}$** (Pin 2)
Supply Voltage positive terminal.

**$V_{SS}$** (Pin 3)
Supply Voltage negative terminal.

**A** (Pin 4)
Quadrature Clock Input A. This input has a filter circuit to validate input logic level and eliminate encoder dither.

**B** (Pin 5)
Quadrature Clock Input B. This input has a filter circuit identical to input A.

**x4/$\overline{\text{x1}}$** (Pin 6)
This input selects between x1 and x4 modes of operation. A high-level selects x4 mode and a low-level selects the x1 mode. In x4 mode, an output pulse is generated for every transition at either A or B input. In x1 mode, an output pulse is generated in one combined A/B input cycle.
(See Figure 2.)

PIN ASSIGNMENT - TOP VIEW

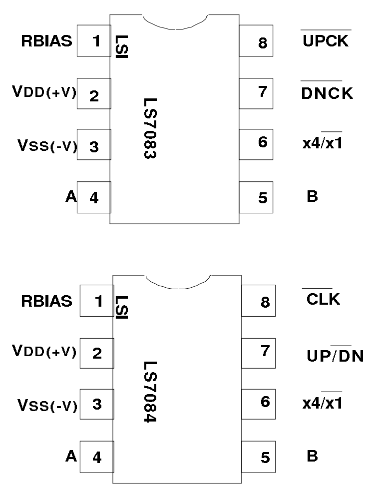



FIGURE 1

**LS7083 - $\overline{\text{DNCK}}$** (Pin 7)
In LS7083, this is the DOWN Clock Output. This output consists of low-going pulses generated when A input lags the B input.

**LS7084 - UP/$\overline{\text{DN}}$** (Pin 7)
In LS7084, this is the count direction indication output. When A input leads the B input, the UP/$\overline{\text{DN}}$ output goes high indicating that the count direction is UP. When A input lags the B input, UP/$\overline{\text{DN}}$ output goes low, indicating that the count direction is DOWN.

**LS7083 - $\overline{\text{UPCK}}$** (Pin 8)
In LS7083, this is the UP Clock output. This output consists of low-going pulses generated when A input leads the B input.

**LS7084 - $\overline{\text{CLK}}$** (Pin 8)
In LS7084, this is the combined UP Clock and DOWN Clock output. The count direction at any instant is indicated by the UP/$\overline{\text{DN}}$ output (Pin 7).

**NOTE**: For the LS7084, the timing of $\overline{\text{CLK}}$ and UP/$\overline{\text{DN}}$ requires that the counter interfacing with LS7084 counts on the rising edge of the $\overline{\text{CLK}}$ pulses.

7083/84-100600-1

Figure 27: Technical specifications of the LS7083/84 series quadrature clock converters.

**ABSOLUTE MAXIMUM RATINGS:**

| PARAMETER | SYMBOL | VALUE | UNITS |
|---|---|---|---|
| DC Supply Voltage | $V_{DD}$ - $V_{SS}$ | 11.0 | V |
| Voltage at any input | $V_{IN}$ | $V_{SS}$ -.3 to $V_{DD}$ +.3 | V |
| Operating temperature | $T_A$ | 0 to +70 | °C |
| Storage temperature | $T_{STG}$ | -55 to +150 | °C |

**DC ELECTRICAL CHARACTERISTICS:**

(All voltages referenced to $V_{SS}$, $T_A$ = 0°C to 70°C.)

| PARAMETER | SYMBOL | MIN | MAX | UNITS | CONDITION |
|---|---|---|---|---|---|
| Supply voltage | $V_{DD}$ | 4.5 | 10.0 | V | - |
| Supply current | $I_{DD}$ | - | 6.0 | μA | $V_{DD}$ = 10.0V, All input frequencies = 0 Hz RBIAS = 2MΩ |
| **x4/$\overline{\text{x1}}$** Logic Low | $V_{IL}$ | 0.3$V_{DD}$ | - | V | |
| **A,B** Logic Low | $V_{IL}$ | - | 0.6 | V | $V_{DD}$ = 4.5V |
| | | - | 1.0 | V | $V_{DD}$ = 9V |
| | | - | 1.1 | V | $V_{DD}$ = 10.0V |
| **x4/$\overline{\text{x1}}$**Logic High | $V_{IH}$ | 0.7$V_{DD}$ | - | V | - |
| **A,B** Logic High | $V_{IH}$ | 3.1 | - | V | $V_{DD}$ = 4.5V |
| | | 5.0 | - | V | $V_{DD}$ = 9V |
| | | 5.6 | - | V | $V_{DD}$ = 10.0V |
| **ALL OUTPUTS:** | | | | | |
| Sink Current | $I_{OL}$ | 1.75 | - | mA | $V_{DD}$ = 4.5V |
| $V_{OL}$ = 0.4V | | 5.0 | - | mA | $V_{DD}$ = 9V |
| | | 5.7 | - | mA | $V_{DD}$ = 10.0V |
| Source Current | $I_{OH}$ | 1.0 | - | mA | $V_{DD}$ = 4.5V |
| $V_{OH}$ = $V_{DD}$ - 0.5V | | 2.5 | - | mA | $V_{DD}$ = 9V |
| | | 3.0 | - | mA | $V_{DD}$ = 10.0V |

**TRANSIENT CHARACTERISTICS:**

($T_A$ = 0°C to 70°C)

| PARAMETER | SYMBOL | MIN | MAX | UNITS | CONDITION |
|---|---|---|---|---|---|
| **A,B** inputs: Validation Delay | $T_{VD}$ | - | 85 | ns | $V_{DD}$ = 10.0V |
| | | - | 100 | ns | $V_{DD}$ = 9V |
| | | - | 160 | ns | $V_{DD}$ = 4.5V |
| **A,B** inputs: Pulse Width | $T_{PW}$ | $T_{VD}$+$T_{OW}$ | Infinite | ns | - |
| **A to B** or **B to A** Phase Delay | $T_{PS}$ | $T_{OW}$ | Infinite | ns | - |
| **A,B** frequency | $f_{A,B}$ | - | $\frac{1}{2T_{PW}}$ | Hz | - |
| Input to Output Delay | $T_{DS}$ | - | 120 | ns | $V_{DD}$ = 10.0V |
| | | - | 150 | ns | $V_{DD}$ = 9V |
| | | - | 235 | ns | $V_{DD}$ = 4.5V Includes input validation delay |
| Output Clock Pulse Width | $T_{OW}$ | 50 | - | ns | See Fig. 4 & 5 |

7083/84-100100-2

Figure 28: Technical specifications of the LS7083/84 series quadrature clock converters.

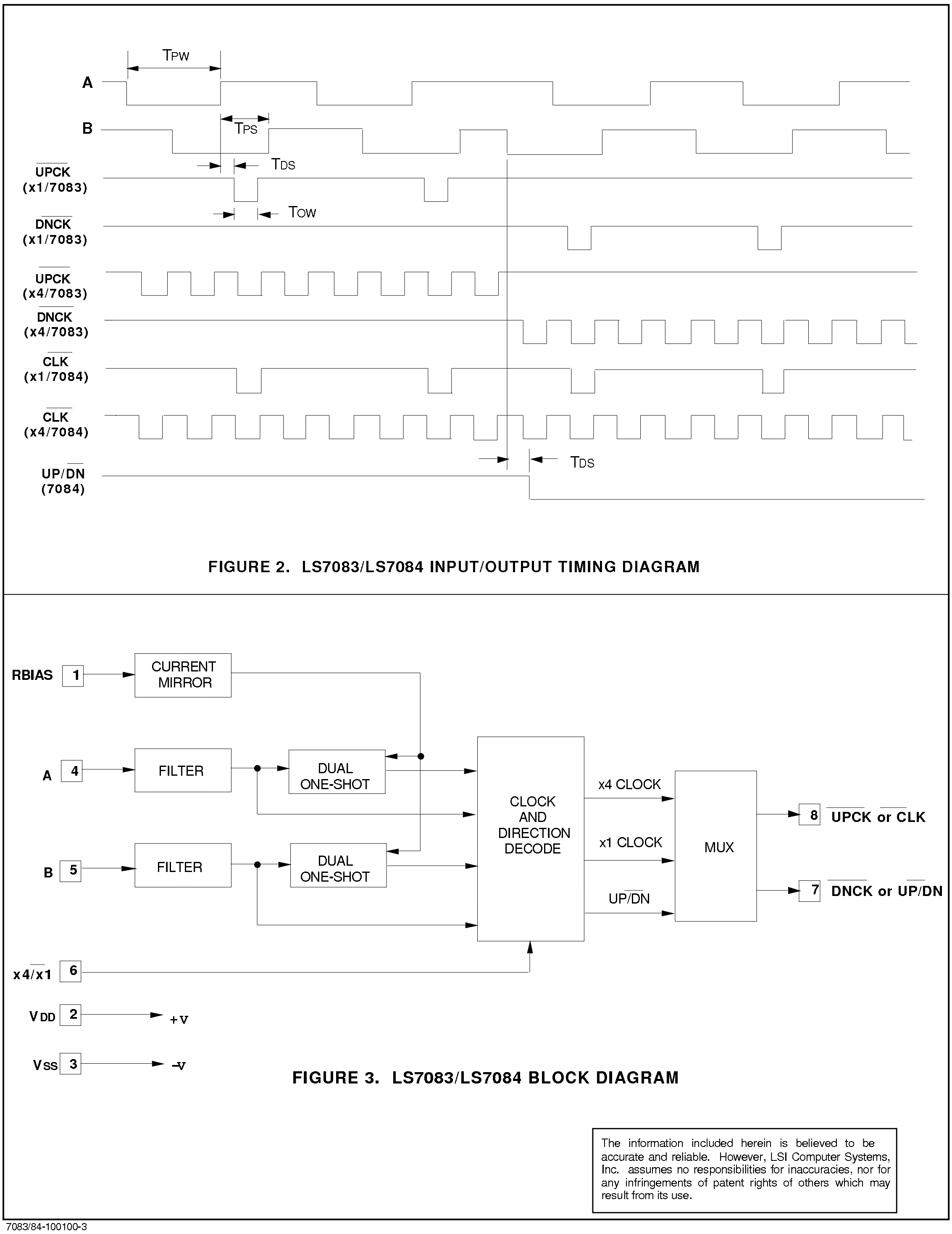


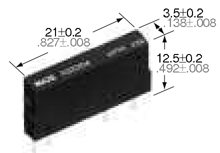

Figure 29: Technical specifications of the LS7083/84 series quadrature clock converters.

**NAiS**

**POWER PhotoMOS RELAYS**
**1-Channel (Form A) Type**

# PhotoMOS RELAYS

**UL File No.: E43149**
**CSA File No.: LR26550**




mm inch

## FEATURES

1. **High capacity PhotoMOS Relay in a compact and slim 4-pin SIL**
2. **Extremely low ON resistance**
3. **Control low-level signal**
Power PhotoMOS relays feature extremely low closed-circuit offset voltage to enable control of low-level analog signals without distortion.
4. **Low-level off state leakage current**
5. **High I/O isolation voltage 2,500 V**
6. **Eliminates the need for a counter electromotive protection diode in the drive circuit on the input side**
7. **Eliminate the need for a power supply to drive the power MOSFET**
8. **PC board layout is simplified**
9. **No restriction on mounting direction**
10. **Varistor incorporated type is also available.**

## APPLICATION

- High speed inspection machines
- IC checker
- NC machine, Robots
- Office machines
- Telecommunication
- Automotive
- Industrial control

## TYPES

| | Output rating | | Part No. | Packing quantity | |
|---|---|---|---|---|---|
| | Load voltage | Load current | | Inner carton | Outer carton |
| AC/DC type | 60 V | 3.0 A | AQZ202 | 25 pcs. | 500 pcs. |
| | 100 V | 2.0 A | AQZ205 | | |
| | 200 V | 1.0 A | AQZ207 | | |
| | 400 V | 0.5 A | AQZ204 | | |
| DC type | 60 V | 4.0 A | AQZ102 | | |
| | 100 V | 2.6 A | AQZ105 | | |
| | 200 V | 1.3 A | AQZ107 | | |
| | 400 V | 0.7 A | AQZ104 | | |

Notes: Load voltage and current of AC/DC type: Peak AC/DC
Load voltage and current of DC type: DC

Figure 30: Technical specifications of the PhotoMOS protection relay.

## RATING

**1. AC/DC type**

1) Absolute maximum ratings (Ambient temperature: 25°C 77°F)

| Item | | | Symbol | AQZ202 | AQZ205 | AQZ207 | AQZ204 | Remarks |
|---|---|---|---|---|---|---|---|---|
| Input | LED forward current | | $I_F$ | 50 mA | | | | |
| | LED reverse voltage | | $V_R$ | 3 V | | | | |
| | Peak forward current | | $I_{FP}$ | 1 A | | | | f = 100 Hz,Duty factor = 0.1 % |
| | Power dissipation | | $P_{in}$ | 75 mW | | | | |
| Output | Load voltage (peak AC) | | $V_L$ | 60 V | 100 V | 200 V | 400 V | |
| | Continuous load current | | $I_L$ | 3.0 A | 2.0 A | 1.0 A | 0.5 A | |
| | Peak load current | | $I_{peak}$ | 9.0 A | 6.0 A | 3.0 A | 1.5 A | 100 ms (1shot), $V_L$ = DC |
| | Power dissipation | | $P_{out}$ | 1.6 W | | | | |
| Total power dissipation | | | $P_T$ | 1.6 W | | | | |
| I/O isolation voltage | | | $V_{iso}$ | 2,500 V AC | | | | |
| Temperature limits | | Operating | $T_{opr}$ | –20°C to +85°C –4°F to +185°F | | | | Non-condensing at low temperatures |
| | | Storage | $T_{stg}$ | –40°C to +100°C –40°F to +212°F | | | | |

2) Electrical characteristics (Ambient temperature: 25°C 77°F)

| Item | | | | Symbol | AQZ202 | AQZ205 | AQZ207 | AQZ204 | Condition |
|---|---|---|---|---|---|---|---|---|---|
| Input | LED operate current | | Minimum<br>Typical<br>Maximum | $I_{Fon}$ | <br>1.0 mA<br>3.0 mA | | | | $I_L$ = 100 mA<br>$V_L$ = 10 V |
| | LED turn off current | | Minimum<br>Typical<br>Maximum | $I_{Foff}$ | 0.4 mA<br>0.9 mA | | | | $I_L$ = 100 mA<br>$V_L$ = 10 V |
| | LED dropout voltage | | Minimum<br>Typical<br>Maximum | $V_F$ | <br>1.25 V (1.6 V at $I_F$ = 10 mA)<br>1.5 V | | | | $I_F$ = 50 mA |
| Output | On resistance | | Minimum<br>Typical<br>Maximum | $R_{on}$ | <br>0.11 Ω<br>0.18 Ω | <br>0.23 Ω<br>0.34 Ω | <br>0.7 Ω<br>1.1 Ω | <br>2.1 Ω<br>3.2 Ω | $I_F$ = 10 mA<br>$I_L$ = max.<br>Within 1 s on time |
| | Off state leakage current | | Minimum<br>Typical<br>Maximum | — | <br><br>10 µA | | | | $I_F$ = 0<br>$V_L$ = max. |
| Transfer characteristics | Switching speed | Turn on time* | Minimum<br>Typical<br>Maximum | $T_{on}$ | <br>2.46 ms<br>5.0 ms | <br>2.40 ms<br>5.0 ms | <br>1.12 ms<br>5.0 ms | <br>1.65 ms<br>5.0 ms | $I_F$ = 10 mA<br>$I_L$ = 100 mA<br>$V_L$ = 10 V |
| | | | Minimum<br>Typical<br>Maximum | | <br>5.64 ms<br>10.0 ms | <br>5.65 ms<br>10.0 ms | <br>2.57 ms<br>10.0 ms | <br>3.88 ms<br>10.0 ms | $I_F$ = 5 mA<br>$I_L$ = 100 mA<br>$V_L$ = 10 V |
| | | Turn off time* | Minimum<br>Typical<br>Maximum | $T_{off}$ | <br>0.22 ms<br>3.0 ms | <br>0.21 ms<br>3.0 ms | <br>0.10 ms<br>3.0 ms | <br>0.08 ms<br>3.0 ms | $I_F$ = 5 mA or 10 mA<br>$I_L$ = 100 mA<br>$V_L$ = 10 V |
| | I/O capacitance | | Minimum<br>Typical<br>Maximum | $C_{iso}$ | <br>0.8 pF<br>1.5 pF | | | | f = 1 MHz<br>$V_B$ = 0 |
| | Initial I/O isolation resistance | | Minimum<br>Typical<br>Maximum | $R_{iso}$ | 1,000 MΩ | | | | 500 V DC |
| | Maximum operating speed | | Minimum<br>Typical<br>Maximum | — | <br><br>0.5 cps | | | | $I_F$ = 10 mA<br>Duty factor = 50%<br>$I_L$ = Max., $V_L$ = Max. |
| Vibration resistance | | | Minimum<br>Typical<br>Maximum | — | 10 to 55 Hz at double amplitude of 3 mm | | | | 2 hours for 3 axes |
| Shock resistance | | | Minimum<br>Typical<br>Maximum | — | 4,900 m/s² {500 G} 1 ms | | | | 3 times for 3 axes |

Note: Recommendable LED forward current $I_F$ = 5 to 10 mA.

*Turn on/off time

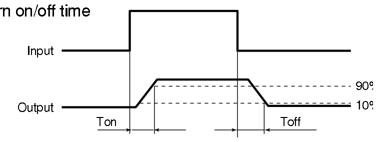



Figure 31: Technical specifications of the PhotoMOS protection relay.