Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Informatica, Gestionale e dell' Automazione

Kinematic Control Algorithms for Cooperative Dual-arm Systems via Redundancy Resolution

Ph.D. Dissertation of:
dott. Davide Ortenzi

Advisor:
Chiar.mo Prof. Sauro Longhi

Coadvisor:
dott. Andrea Monteriù
dott. Alessandro Freddi

Curriculum Supervisor:
Prof. Claudia Diamantini

XV edition - new series

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Informatica, Gestionale e dell' Automazione

Kinematic Control Algorithms for Cooperative Dual-arm Systems via Redundancy Resolution

Ph.D. Dissertation of:
dott. Davide Ortenzi

Advisor:
Chiar.mo Prof. Sauro Longhi

Coadvisor:
dott. Andrea Monteriù
dott. Alessandro Freddi

Curriculum Supervisor:
Prof. Claudia Diamantini

XV edition - new series

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Facoltà di Ingegneria
Via Brecce Bianche – 60131 Ancona (AN), Italy

"Fatti non foste a viver come bruti,
ma per seguir virtute e canoscenza"
Divina Commedia,
Dante Alighieri.

To my girlfriend Chiara,
because she has encouraged me to overcome my limits.

Acknowledgments

The present study has been supported by several people, who have helped to improve important skills.

First of all, I would thank my tutor, Prof. Sauro Longhi, for giving me the opportunity to carry out my research also at a prestigious foreign university, such as Aalto University of Helsinky.

Moreover, all my results obtained during the research activity have been made possible thanks to two fundamental people as my supervisors: Andrea and Alessandro. I would like to thank them for all their precious and tireless help for everything, especially during the writing of our publications, many of which completed together late at night. However, the most important thing that I received from their research activities concerns the passion for this work, which often requires a lot of sacrifice and motivation.

I would thank Prof. Ville Kyrki, who welcomed me into his research group and supported my research during my visiting time in Finland. From this experience, I have learned the organization and methodology that he has always demonstrated in his work. Moreover, I would thank Raj, with whom I pleasantly collaborated and all the colleagues in the office, as Mattia and Roel, who made my period in Finland really pleasant.

A great thank to Patrizia and her talent, because her illustrations help to clarify the basic concepts of my research topics.

I thank my colleagues of department, which have become also my friends. They have always supported me, with their professional experience in order to improve the quality of my research activity, especially at the beginning of this PhD course. I think that my research results would not be so pleasant, without the enthusiasm of (in alphabetic order) Francesco, Giovanni, both Luca, Lucia, Lucio, Luigi and Sabrina.

For last one, but most important one, I thank my girlfriend Chiara, because, in spite of the big distance that separates us, she still continues to support me tirelessly in all my changes, of today and tomorrow...

Ancona, Marzo 2017

dott. Davide Ortenzi

Abstract

This thesis tackles the problem of kinematic control for cooperative dual-arm systems via redundancy resolution. First the redundancy analysis of two cooperative manipulators is presented, which shows how they can be considered as a single redundant manipulator, through the use of the relative Jacobian matrix. Then the Jacobian null space technique is considered: in this way the kinematic redundancy can be resolved by applying the principal local optimization techniques used for the single manipulator case; moreover several tasks with different execution priority levels can be performed at the same time (e.g. obstacle avoidance, manipulability index maximization or joint position constraints satisfaction). The kinematic resolution is then specifically addressed to both prevent and take into account possible faults. In detail, since violation of joint position constraints may damage a manipulator, a joint position limit avoidance strategy is proposed. This strategy is able to satisfy all joint limits even when the number of redundant motions are no longer sufficient to ensure them with a classical approach, by locally and temporary changing the desired end-effectors motion when redundant motions are not available. Finally, a fault tolerance algorithm is proposed, which use the degree of redundancy both to maximize the local optimum fault tolerance configuration, and to overcome the loss of the end effector velocity due to fault occurrence, by use of the saturation null space method. Several simulated and experimental examples are illustrated, which demonstrate the high flexibility of the proposed algorithms, which can implemented on different kind of cooperative manipulators.

Sommario

Questa tesi affronta il problema del controllo cinematico di due manipolatori cooperanti, attraverso la risoluzione della ridondanza cinematica. In primo luogo, la tesi propone una descrizione dell'analisi della ridondanza dovuta alla cooperazione dei manipolatori, poichè essi possono essere considerati come un unico manipolatore ridondante, mediante il metodo dello Jacobiano relativo. Successivamente, viene studiata la tecnica di risoluzione della ridondanza basata sulla proiezione di diversi compiti nello spazio nullo dello Jacobian, in modo da eseguire contemporaneamente diversi compiti, aventi differenti livelli di priorità. Tali compiti possono migliorare la cooperazione dei due manipolatori, come ad esempio, evitare gli ostacoli posti in prossimità, massimizzare l'indice di manipolabilità o rispettare i vincoli di posizione presenti nei giunti. In particolare, la violazione di vincoli di posizione dei giunti è dettagliatamente studiata, poichè tale condizione può indurre a molteplici criticità, fra le quali il danneggiamento del sistema stesso. Pertanto, é proposta una strategia di controllo che permette di rispettare tutti i limiti dei giunti, anche qualora il numero di movimenti ridondanti non fosse più sufficiente a garantirli mediante un approccio classico. Tale tecnica degrada, parzialmente e temporaneamente, le prestazioni del movimento assegnato ai due terminali, al fine di rispettare tutti i vincoli di giunto, anche quando i moti ridondanti non sono disponibili. Infine, viene proposto un algoritmo di tolleranza del guasto, che impiega il grado di ridondanza del sistema sia per ottimizzare localmente una configurazione dei manipolatori robusta al guasto, che per limitarne i suoi effetti sulle loro prestazioni. Alcuni esempi di possibili applicazioni sono stati realizzati sia in simulazione che sperimentalmente. Essi dimostrano l'elevata flessibilità degli algoritmi proposti che possono essere adeguatamente implementati su diverse tipologie di manipolatori.

Contents

1 Introduction 1

1.1 Motivations 1

1.2 Literature review 3

1.3 Contributions 5

1.4 Thesis structure 5

2 Kinematic redundant manipulators 7

2.1 Manipulator mobility 7

2.2 Spaces definition 9

2.2.1 Joint space 9

2.2.2 Cartesian space 9

2.2.3 Workspace 10

2.2.4 Task space 10

2.3 Kinematic redundancy definition 10

2.4 Differential kinematics 12

2.4.1 Geometric Jacobian 13

2.4.2 Analytical Jacobian 15

2.4.3 Inverse differential kinematics 17

3 Redundancy resolution via local optimization 21

3.1 Jacobian pseudo-inverse method 21

3.2 Jacobian null space method 24

3.2.1 Jacobian null space projection matrix 24

3.2.2 Gradient projection method 26

3.2.3 Objective functions 27

3.2.4 Saturation in the null space method 27

3.2.5 Hierarchical priority execution task 29

4 Kinematic modelling of cooperative manipulators 33

4.1 Relative Jacobian method 33

4.2 Inverse kinematics solution for cooperative manipulators 36

5 Proposed kinematic control algorithms for obstacle and joint limits avoidance 39
5.1 Control algorithm for obstacle avoidance 39
5.1.1 Case study 39
5.1.2 Task Description 39
5.1.3 Simulation results 43
5.2 Control algorithm for joint limits avoidance 46
5.2.1 Proposed joint position limits avoidance strategy 47
5.2.2 First case study 49
5.2.3 Second study case 53
5.2.4 Supervisor controller for redundancy management . . . 58
5.2.5 Third study case 60

6 Proposed fault tolerant algorithms 65
6.1 Relative manipulability index 65
6.2 Optimization of fault tolerant configuration 67
6.3 Proposed fault tolerant configuration algorithm 68
6.4 Joint fault management with the SNS method 69
6.5 Case study 70
6.5.1 Results 71
6.5.2 Discussion 73

7 Conclusions and future works 75
7.1 Conclusions 75
7.2 Future works 76

List of Figures

2.1 Three-link planar arm (revised version of Figure 2.20 from [55]) 8
2.2 Calculation of $i-th$ joint velocity contribution to the end-effector velocity (revised version of Figure 3.2 from [55]) 14
2.3 Block diagram of the close loop inverse kinematics (partial reproduction of Figure 3.11 from [55]) 18

3.1 Example of non-repeatable motion in the joint space 23
3.2 Relation between joint velocity space and end-effector velocity space 24
3.3 Geometric view on Jacobian null space 25
3.4 Example of SNS application 29
3.5 Example of hierarchical priority execution task 30

4.1 Two cooperative manipulators (partial reproduction of Figure 1 from [26]) 34
4.2 Block diagram of the relative Jacobian pseudo-inverse algorithm 37

5.1 Concept of the wheelchair with two manipulators within an AAL scenario 40
5.2 Top view of the system (a) and initial planar manipulators configuration (b) 40
5.3 Block diagram of two cooperative manipulators 43
5.4 Final manipulators configuration without (a) and with (b) obstacle avoidance task. 44
5.5 Joint angle position without (a) and with (b) obstacle avoidance task 44
5.6 Relative end-effectors position error (a) and orientation error (b) 45
5.7 End-effector A position along the path (a) and the path following error (b) 45
5.8 Minimum distance between manipulators and obstacle 46
5.9 Activation function for one component w_i of W matrix. 48
5.10 Operating principle of the joint limits avoidance strategy. . . . 49
5.11 Manipulator configurations: (a) initial, (b) Case I and III final configuration, (c) Case II final configuration.) 50

5.12 A (B) manipulator joint position space relative to q_{A1}-q_{A2} (a) (q_{B1}-q_{B2} (b)). 52
5.13 A and B manipulators Cartesian path for the three considered simulation cases. 53
5.14 Joint velocity space with repulsive velocity for A manipulator and B manipulator. 54
5.15 Two robot manipulators holding a woodblock to perform tightly coordinated cooperative operations. 55
5.16 Heterogeneous two-robot system: coordinate frame transformation for the relative Jacobian formulation. 55
5.17 Coordinated manipulation results of tightly-coupled manipulators under joint constraints (Case B and C). 56
5.18 Cooperative performance of tightly-coupled manipulators for the Cases (Case A-C), refer to Table5.3. 57
5.19 Controller treats joint limits enforced in manipulators. 58
5.20 Operating principle of the supervisor controller. 59
5.21 Operating principle of the joints avoidance strategy. 60
5.22 Study case setup (a) and coordinate transformation of the Baxter arms (b) 61
5.23 Controller treats joint limits enforced in manipulators. 63
5.24 Coordinated manipulation results. 63
5.25 Cooperative performance of tightly-coupled manipulators. . . . 64

6.1 3 DoF planar manipulator with fault tolerant configuration with respect to joint one (q_1) (revised version of Figure 2 from [52]) 66
6.2 Starting configuration of the dual arm system 70
6.3 Relative manipulability indices for 50% reduction of maximum joint velocity 71
6.4 Relative manipulability indices for 80% reduction of maximum joint velocity 71
6.5 Joint velocity when the maximum $\dot{q}_1$ velocity is reduced of 50% at t=5s 72
6.6 Joint velocity when the maximum $\dot{q}_3$ velocity is reduced of 50% at t=5s 72
6.7 Final manipulators configuration, when a 80% maximum velocity loss on (a) $\dot{q}_1$ and (b) $\dot{q}_3$ occurs at t=5s 73

List of Tables

5.1 Joint limits. 50
5.2 Decomposition of the task into prioritized subtasks. 51
5.3 Experimental cases. 53
5.4 Joint limit parameters. 56
5.5 Description of the experimental considered cases. 62
5.6 Parameters used for the experimental cases. 62

Chapter 1

Introduction

Dual-arm manipulation systems have been attracting a growing from the scientific community over the last few years. Although a universal accepted definition does not exist, a good starting point is provided [1], where dual arm manipulation is presented as the cooperation of two robotic systems physically interacting with an object, moving or reshaping it. The use of multiple cooperative manipulators offers several advantages in many different contexts [2]. Potential applications for cooperative manipulation include, but are not limited to, the manufacturing industry [3, 4, 5], hazardous environments such as nuclear sites [6, 7, 8], underwater [9, 10] and space [11, 12]. Cooperation can bring economical benefits since a wide range of tasks can be accomplished through the use of multiple simpler and less expensive heterogeneous robots [13].

The dual-arm manipulation system was firstly introduced to replace workers in dangerous manufacturing processes. Early robotic manipulators were constructed by Goertz in the 1940s for handling radioactive goods [14]. Later, they were employed for marine and space exploration, where the dual arm manipulators were noticeably improved. In 1969, the NASA's Johnson Space Center introduced anthropomorphic dual arm teleoperators, due to the analogies with human operations [15].

Today, the modern technological progress and the increasing acceptance of technology by users have encouraged the scientific community to focus on the development of dual-arm manipulators able to work in user centered environments, such as surgery[16] and Ambient Assisted Living (AAL) applications [17]. An example of AAL application consists in dual-arm manipulators mounted on a wheelchair, which can assist disabled people to reach and handle objects, or to perform more complex tasks [18, 19, 20].

1.1 Motivations

Cooperation between two robotic arms manipulating an object is a challenging task from an engineering perspective, since their relative motions have to be properly controlled in order to perform the desired operations. Moreover, the two arms and the object realize a closed kinematic where the degrees of motion of the system are greater than those which are generally required to perform the task, therefore the inverse kinematics problem admits an infinite number of solutions [21]. For this reason the dual-arm manipulation context, the redundant variables can be employed to perform tasks, such as collision avoidance [22], or satisfy specific performance criteria, such as singularities avoidance [23], mechanical joint limits avoidance [24] or improvement of manipulability along a chosen direction [25]. Since the control of cooperative manipulators is more complex than the control of a single manipulator, I decided to employ a method based on the relative Jacobian matrix. This method permits to consider two redundant manipulators like a unique redundant manipulator, whose number of joints is equal to the sum of each manipulator's joints, while the end-effector motion variables correspond to the relative motion between the two end-effectors. This method presents several advantages with respect to the individual control of each manipulator. Firstly, a dual-arm system modelled according to the relative Jacobian method can be controlled by the same algorithms used for controlling single manipulators. Secondly, the compact expression of the relative Jacobian matrix [26] is simple to calculate when Jacobians of the individual manipulators are known and, if a manipulator is replaced with another one, it is sufficient to change its Jacobian without the need of complex calculus. Therefore the research activity presented in this thesis is focused on this method. In particular the main objective of this thesis is to extend the classical redundancy resolution algorithms for controlling dual-arm system, composed by homogeneous and heterogeneous manipulators.

Moreover, since some cooperative dual-arm manipulators have been designed to perform tasks in harsh environments, they have to complete an assigned task even in presence of faults. In such a scenario, a dual-arm system can be defined as fault tolerant [27]. There are several sources of faults that can compromise the correct motion of manipulators. The main faults can be classified into five categories: Free-Swinging Joint Faults (FSJFs) [28], Locked Joint Faults (LJFs) or joint failure [29], Partial Loss of Joint Torque Faults (PLJTFs) [30], incorrectly measured Joint Position Faults (JPFs) and incorrectly measured Joint Velocity Faults (JVFs) [31]. The first three categories consist in a partial or total loss of joints motion, while the other two comprises joint sensors faults. The probability of occurrence of one of these faults on a robot is inversely related to the reliability of its components [32]. For this reason, a first fault tolerant approach consists in using high quality components, i.e. perfectness [33], which dramatically increase the system cost. Regarding to the PLJTFs, a low-cost fault tolerant system is proposed in this thesis. The basic idea consists of designing a kinematic controller based on redundancy resolution algorithm, which allows both to maximizes the local optimum fault tolerance configuration (before joint fault) and to overcome the loss of the end effector velocity due to fault occurrence (after joint fault).

1.2 Literature review

Although only a few works focused on inverse kinematic control of dual arms motion, such problem has been addressed extensively in literature for the single manipulator case. The related research is mainly based on global and local optimization of a specific objective function. In particular, the global optimization permits to calculate the optimal path which satisfies a performance criteria (*off-line mode*) [34], like avoiding obstacles whose positions are known a priori, while the local optimization permits to calculate the current desired joints velocity in order to locally satisfy the performance criteria (*real time mode*) [35].

The simplest local optimization technique is represented by the pseudo-inverse solution [36], which provides the joint velocity with the minimum norm among those satisfying the task constraint. This techniques do not permit to use the redundant joints for any secondary task, which is instead possible by adopting the so called task augmentation method [37]. The task augmentation method consists in augmenting the task vector to tackle additional objectives by implementing two different methods: extended Jacobian [38, 39] and augmented Jacobian [40]. The disadvantage of these two techniques is due to occurrence of singularities when the Jacobians associated with the additional objectives are linearly dependant [38, 41]. In order to overcome these problems, the Jacobian null space technique was proposed in [42]. This method consists of projecting a specific objective function (*secondary task*) into the Jacobian null space of another task with higher execution priority, in order to obtain a hierarchical task priority structure, as described in [43, 44]. Since the joint velocities required by secondary tasks generate self-motions in the robot (changing the current manipulator configuration), they do not affect the performance of the task with highest priority, namely *primary task*.

A widely used technique of Jacobian null space is the Gradient Projection Method (GPM), which projects the gradient of a specific objective function in the null space of Jacobian associated with the desired task, in order to maximize/minimize it. The main drawback of GPM is the redundant motion needed to optimize the objective function (e.g. in the joint limits avoidance task, all joint positions are kept close to the mid-range joint position). In order to limit the number of redundant motions, other approaches operate only on those joints whose positions are close to the respective limits [45]. A recent work [46] projects a joint limit avoidance function based on Prescribed Performance

Control methodology (PPC) into the Jacobian null space of the desired task. However, since the dimension of the Jacobian null space depends on the number of the available redundant motions, PPC generally does not guarantee the execution of the secondary task. On the other hand, some algorithms based on Jacobian null space method ensure this execution by degrading the performance of primary task. One of these algorithms is described in [47] and consists of the division of the main task into several subtasks. Single subtasks can be removed and recovered by a supervisor controller in order to ensure a sufficient number of redundant motion to the secondary task. A different algorithm is the Saturation in the Null Space (SNS) algorithm [48], which considers the joint velocity limits avoidance as secondary task. In particular, this technique projects the exceeded joint velocity into the null space of the main task Jacobian matrix, namely partial Jacobian matrix, composed by the not-saturated joints velocity. When there are not sufficient redundant motions to guarantee the joint velocity constraints, the desired primary task velocity is reduced. Recently this algorithm was applied in the context of two cooperative manipulators affected by joint constraints [49]. The study involves the cooperation of two KUKA lightweight manipulators based on relative Jacobian method.

Focusing on the joint fault tolerance study, several method can be considered in the literature. In particular, a widely applied method consists in transforming a FSJF into a LJF, so that they are managed in the same way [50, 51], while the PLJTFs is kinematically modelled as a reduction of the maximum joints velocity due to a partial torque loss of its servomotor [30]. An efficient approach to compensate these types of faults in the end effector consists in the use of redundant degrees.

The manipulators' configuration during a fault is crucial in fault tolerant applications. In fact, the degradation of the system performances depends on which joint of the manipulators is affected by the fault [51]. If the joint that provides the greater contribution to the end effector motion is faulty, then the performance will heavily drop. A classical example is the human arm which has to execute the task “drink”. It is clearly a redundant system because it has 7 DoF, but it is not fault tolerant respect to an elbow joint failure, since it is the only joint that can reduce the distance between the shoulder and the wrist. These critical configurations can be avoided by projecting the gradient of a suitable objective function into the Jacobian null space, in order to avoid bad joint positions before a fault/failure occurs [34]. Another approach is to maximize the manipulability index by calculating a proper objective function [25]. This method allows to obtain a local optimum configuration, because it involves a minimal reduction of the manipulability index for any joint failure. A possible way to know the amount of manipulability after a failure is to compute the relative manipulability index [52] which is the ratio between the manipulability index after and before a failure. The research activity of this thesis about the fault tolerant applications is based on this last approach.

1.3 Contributions

Although several works focusing on the motion constraints and fault tolerant applications have been performed by several redundancy resolution algorithms in single manipulator case, only few works considered them in a cooperative manipulation scenario. Therefore the proposed research activity allowed to consider such methods for cooperative manipulators by using the relative Jacobian method.

Therefore, the research activity described in this thesis have led the following innovative contributions with respect to the state of the art:

- Implementation of the hierarchical execution task architecture based on relative Jacobian method [53];
- Analysis and development of a kinematic controller that provides coordinated motions and joint limit avoidance during the cooperation between a redundant and non-redundant manipulators;
- Development of a novel joint position limits avoidance strategy, which is able to satisfy all joint limits of homogeneous redundant manipulators, even when there are not available redundant motions;
- Development of a joint fault tolerance algorithm, which is able to guarantee the correct cooperation of two planar manipulators when the maximum velocity of a joint is reduced [54].

1.4 Thesis structure

The thesis has the following structure: Chapter II presents the knowledge about the kinematic redundancy with the introduction of the inverse differential kinematics problem; Chapter III summarizes the theoretical background about the redundancy resolution algorithm that are subsequently used in the proposed kinematic controller. The derivation of the compact relative Jacobian formulation and its implementation in the kinematic control is proposed in Chapter IV, while the proposed algorithms with respect to the motion constraints cases are described in Chapter V. Chapter VI describes the proposed fault tolerance algorithm, which has been tested on two planar cooperative manipulators. Conclusions and future works complete the thesis in Chapter VII.

Chapter 2

Kinematic redundant manipulators

This chapter focuses on some fundamental definitions necessary to well understand the kinematic redundancy problem in the cooperative manipulation control framework.

2.1 Manipulator mobility

The kinematically redundant manipulators are particular mechanical structures that possess more motion variables then those required to perform the specific task. This requires to first it is firstly necessary to analyse the manipulator mobility.

In robotics, the number of independent variables necessary to determine completely the system configuration in space is defined as Degrees of Freedom DoFs (or Degrees of Motion DoMs). This definition is different from that used to describe the pose of a rigid body (space's DoF) in a space n-dimensional, by means of independent coordinates. In order to calculate the DoFs of a generic mechanism, namely F, it is possible to implement the formula in (2.1), which depends on several kinematic parameters:

$$F = \lambda(j-1) - \sum_{i=1}^{n} c_i \tag{2.1}$$

where c_i is the imposed kinematic constraints of the $i-th$ joint, n and j are the number of joints and rigid links possessed by the mechanism (base link enclosed), while λ is the number of space's DoFs.

Since for each joint, the sum of the number of constraints c_i and permitted degree of freedom f_i have to be equal to the space's DoF ($\lambda = c_i + f_i$), it is possible to obtain the total number of system constraints as:

$$\sum_{i=1}^{n} c_i = \sum_{i=1}^{n} (\lambda - f_i) = j\lambda - \sum_{i=1}^{n} f_i \tag{2.2}$$

from which it is possible to obtain the Kutzbach formula (when $\lambda = 6$) or

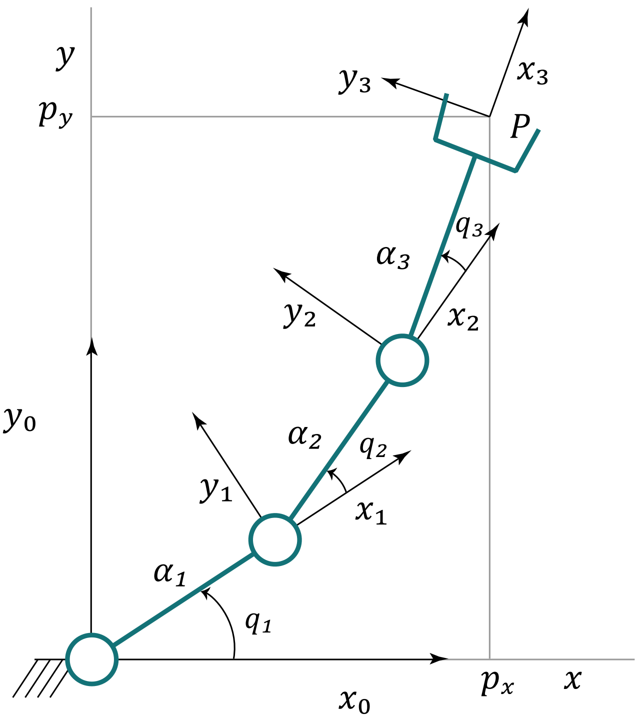


Figure 2.1: Three-link planar arm (revised version of Figure 2.20 from [55])

Grubler formula (when $\lambda = 3$):

$$F = \lambda(j - n - 1) + \sum_{i=1}^{n} f_i \tag{2.3}$$

A possible application about (2.3) is to calculate the DoF of the planar manipulator shown in Figure 2.1 in accordance with Grubler formulation

The planar manipulator in Figure 2.1 is composed by $j = 3$ links and $n = 3$ joints placed on a planar space ($\lambda = 3$). Since all joints are rotative, they allow only the relative rotation between two consecutive links $f_i = 1$, but not their relative translation $c_i = 2$, so that the manipulator DoFs is equals to three, $F = 3$.

This result implies an important property about the mobility of manipulators having open-chain structure. In fact, for open chain manipulators, the DoFs are equal to the number of joints n ($F = n$), hence in the following chapters the DoF of a manipulator will be implicitly expressed by its number of joints n.

A system composed by two planar cooperative manipulators having the same structure shown in Figure 2.1 has a DoF equals to six, $(F = 2{*}n = 6)$. However when they cooperate on a same object, a closed chain is realized and the DoF is smaller then the total number of joints, $F < n$. A possible demonstration can be given by applying (2.3) for a specific cooperation, in which the two end-effectors can change their relative orientation but not their relative positions. This type of cooperation can be kinematically modelled as a revolute joint belonging to the closed chain.

Therefore, the obtained structure has $n = 7$ but its DoF is equal to four, because 3 joint positions (passive joints) depend on the other 4 joints (active joints). Therefore, it is possible to assert that the loss of DoFs obtained during the cooperation depends on the number of kinematic constraints that are introduced from it. The motion of two cooperative manipulators will be examined in details in chapter 4.

2.2 Spaces definition

2.2.1 Joint space

The joint space (also called the configuration space) denotes the the space in which the n-dimensional vector of joint variables, $\boldsymbol{q}$, is defined as [55]:

$$\vec{\boldsymbol{q}} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix} \tag{2.4}$$

where the *i-th* component q_i of the vector $\boldsymbol{q}$ corresponds to the angular position of *i-th* joint θ_i, $q_i = \theta_i$. Since the dimension of $\boldsymbol{q}$ is equal to the total number of joints that are present in the structure, it is also equal to the DoFs of a open chair manipulator as discussed in the previous section. The joint space allows to obtain a unique description of the current configuration assumed by a manipulator and thus of its end effector pose via direct kinematics calculation.

2.2.2 Cartesian space

The Cartesian space is the space in which is defined the m-dimensional vector $\boldsymbol{x_e}$ (with $m \leq 6$), in which are reported the variables that describe the position and the orientation of the end-effector [55]:

$$\vec{\boldsymbol{x}}_{\boldsymbol{e}} = \begin{bmatrix} \vec{\boldsymbol{p}}_{\boldsymbol{e}} \\ \vec{\boldsymbol{\phi}}_{\boldsymbol{e}} \end{bmatrix} \tag{2.5}$$

where $\vec{\boldsymbol{p}}_{\boldsymbol{e}}$ and $\vec{\boldsymbol{\phi}}_{\boldsymbol{e}}$ indicate the Cartesian coordinate and minimum orientation representation (i.e. Euler angles) of the end-effector respect to the base frame.

The Cartesian space is very useful for trajectory planning (path and execution time) in the space, because the task variables are defined in this space, rather than joint space. However the end effector pose expressed in Cartesian space depends on joint positions (Joint space) via the direct kinematics relation, which can be expressed as

$$\vec{\boldsymbol{x}}_{\boldsymbol{e}} = \vec{\boldsymbol{k}}(\vec{\boldsymbol{q}}) \tag{2.6}$$

where the vectorial function $\vec{\boldsymbol{k}}$, which is generally not linear, allows to calculate the end effector pose in Cartesian space from the knowledge of joints position.

2.2.3 Workspace

When the joints position are varied across their entire operational range, the origin of the end-effector frame [55] describes a points region on the Cartesian space, which is called the workspace. Generally the workspace can be classified in reachable space and dexterous space. The latter defines the points region that the origin of the end effector frame can reach by several orientations, while the former is the region that the end effector frame can reach by at least orientation. Therefore the reachable space coincides with workspace, while the dexterous space is a subset of the previous one.

The workspace of a single manipulator depends on its geometry and joint position limits. However when two cooperative manipulators makes a closed kinematic chain the workspace is smaller than without cooperation, because it also depends on the type of cooperation and the distance between two manipulator bases.

2.2.4 Task space

The task space is the space in which is defined a r-dimensional vector $\vec{\boldsymbol{x}}_{\boldsymbol{r}}$, that is composed by the variables that are necessary to describe the end effector pose in order to perform the desired task [55]. The maximum number of variables is equal to the number of the Cartesian space variables ($r \leq m$). This definition is very important to determine when a manipulator is redundant, as described in the next section.

2.3 Kinematic redundancy definition

Given the previous definitions, it is now possible to define formally the kinematic redundancy of a robotic manipulator, as well as defined in [55].

Kinematic Redundancy A open-chain manipulator is *kinematically redundant* when the number of DoFs n is greater than the number of variables r that are necessary to describe a given task (dimension of the task space), $n \geq r$.

Intrinsic Redundancy A manipulator is defined as *intrinsically redundant* when the number of DoFs n is greater than the number of variables m relative to the Cartesian space in which the manipulator operates, $n > m$.

Functional Redundancy A manipulator is *functionally redundant* when the number of DoFs n is equal to the dimension of the Cartesian space m ($n = m$), but the dimension of task space r is smaller than that of the Cartesian Space m, (from which it follows $n > r$).

The first is the main definition of redundancy, which demonstrates the dependence of the redundancy on the dimension of the specific task. Therefore the same manipulator can be redundant with respect to a specific task and non-redundant with respect to another task. This definition can be classified in two sub-categories: intrinsic and functional redundancy. In detail, the intrinsic redundancy definition does not depend on the dimension of the task space, which can be equal or less than that of the Cartesian space, since it is sufficient that kinematic structure of the robot has a number of DoFs grater than those request from Cartesian. On the other hand, the functional redundancy definition refers to the specific case where the kinematic structure of the robot has a number of possible motions equal to the number of the Cartesian space variables, but some of these are not necessary to perform the specific task. The following example provides a better understanding of the differences among the three redundancies. Considering the planar manipulator in figure 2.1 with 3 joints, which has to only translate and rotate an object ($r = 3$) on a plane ($m = 3$). Such manipulator is neither intrinsically redundant ($n = m$) nor functionally redundant ($m = r$), but it can become functional redundant by obtaining ($m > r$). A possible way to consider the manipulator like redundant consists of specifying only the translation of the object on the plane and not its orientation. Therefore, the planar manipulator is not intrinsically redundant, because its kinematic structure is not changed ($n = m$), but since the specific task requires only two variable of the Cartesian Space ($r < m$), then the manipulator becomes functional redundant with respect to this task ($n > r$). This propriety is able to make a manipulator redundant by reducing the number of task variables, permits to satisfy other secondary tasks (e.g. objective function) and it will be used in the following chapter. Finally, it is possible to define the *Degree of redundancy* of a redundant manipulator as the number of redundant motions dr that are possessed by the system, which is calculated as shown in (2.7).

$$dr = n - r. \tag{2.7}$$

2.4 Differential kinematics

The inverse of the kinematic equation in (2.6) is very useful to determine the joint positions of a manipulator from the desired end-effector poses. Therefore, the resolution of the inverse kinematic problem is very important to translate a desired trajectory defined in the Cartesian space to motions defined in joint space, in order to actualize the desired motion. However, while the direct kinematic equation permits to obtain uniquely the end-effector pose by joint position variables, the inverse kinematic problem admits a closed form solution only for manipulators having simple kinematic structure. In fact, since the direct kinematic in (2.6) is not linear, it introduces some limitations when:

- The end effector assumes a particular position and orientation in the Cartesian space;
- It is not possible to combine the end-effector position and orientation to several joint variables set;
- The manipulator is redundant.

A possible solution concerns of introducing a linear transformation matrix that permits to project the joint velocity space into the Cartesian velocity space (direct differential kinematics) and vice-versa (inverse differential kinematics). Such linear transformation is a $r \times n$ dimensional matrix, which is called *Jacobian*, J, and it depends on the current configuration $\vec{\boldsymbol{q}}$ assumed by the manipulator. The Jacobian is of two different types:

Geometric Jacobian When the end effector velocity vector $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{e}}(t)$ is defined in terms of linear velocity $\dot{\vec{\boldsymbol{p}}}_{\boldsymbol{e}}(t)$ and angular velocity $\vec{\boldsymbol{\omega}}(t)$, the transformation matrix is called *geometric Jacobian*, $\boldsymbol{J}(\vec{\boldsymbol{q}}(t))$ [55].

Analytic Jacobian When the end effector orientation velocity $\dot{\vec{\boldsymbol{\phi}}}(t)$ is expressed as the time derivative of the Euler angles, the transformation matrix is called *analytic Jacobian* $\boldsymbol{J_A}(\vec{\boldsymbol{q}}(t))$ [55].

2.4.1 Geometric Jacobian

The direct differential kinematics can be calculated by geometric Jacobian matrix as shown in (2.8):

$$\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{e}}(t) = \begin{bmatrix} \dot{\vec{\boldsymbol{p}}}_{\boldsymbol{e}}(t) \\ \vec{\boldsymbol{\omega}}_{\boldsymbol{e}}(t) \end{bmatrix} = \begin{bmatrix} \boldsymbol{J_P}(\vec{\boldsymbol{q}}(t)) \\ \boldsymbol{J_O}(\vec{\boldsymbol{q}}(t)) \end{bmatrix} \dot{\vec{\boldsymbol{q}}}(t) = \boldsymbol{J}(\vec{\boldsymbol{q}}(t))\dot{\vec{\boldsymbol{q}}}(t) \tag{2.8}$$

where $\boldsymbol{J_P}$ is the $(dim(\dot{\vec{\boldsymbol{p}}}_{\boldsymbol{e}}) \times n)$ - dimensional matrix relative to the joint velocity vector contribution to the translation velocity vector $\dot{\vec{\boldsymbol{p}}}_{\boldsymbol{e}}$, while $\boldsymbol{J_O}$ is the $(dim(\dot{\vec{\boldsymbol{\omega}}}_{\boldsymbol{e}}) \times n)$ - dimensional matrix mapping the joint velocity vector into the angular velocity vector $\vec{\boldsymbol{\omega}}_{\boldsymbol{e}}$.

Since $\boldsymbol{J}$ depends on the kind of joints constituting the manipulators (revolute, prismatic, etc...), it is first of all necessary to choose them. In this study the joints considered are of revolute type, because the manipulators that will be described have only revolute joints.

In order to calculate the geometric Jacobian, it possible to considerate separately the linear velocity contribution $\boldsymbol{J_P}$ and the angular velocity contribution $\boldsymbol{J_O}$. The first one can be obtained by differentiating the end effector position $\vec{\boldsymbol{p}}_{\boldsymbol{e}}$ with respect to time as shown below:

$$\dot{\vec{\boldsymbol{p}}}_{\boldsymbol{e}} = \sum_{i=1}^{n} \frac{\partial \vec{\boldsymbol{p}}_{\boldsymbol{e}}}{\partial q_i} \dot{q}_i = \sum_{i=1}^{n} \boldsymbol{J_{P_i}} \dot{q}_i = \boldsymbol{J_P} \dot{\vec{\boldsymbol{q}}}(t) \tag{2.9}$$

where the subscript i indicates the $i - th$ joint. Therefore, (2.9) shows how $\dot{\vec{\boldsymbol{p}}}_{\boldsymbol{e}}$ is obtained as sum of the $\boldsymbol{J_{P_i}} \dot{q}_i$ terms, each of which represents the linear velocity contribution given by $i - th$ joint to the end-effector.

In accordance with Figure 2.2, the linear velocity contribution of each revolute joint, $\boldsymbol{J_{P_i}} \dot{q}_i$, can be obtained by following expression:

$$\boldsymbol{J_{P_i}} \dot{q}_i = \vec{\boldsymbol{\omega}}_{\boldsymbol{i-1,i}} \times \vec{\boldsymbol{r}}_{\boldsymbol{i-1,e}} = \dot{q}_i \vec{\boldsymbol{z}}_{\boldsymbol{i-1}} \times (\vec{\boldsymbol{p}}_{\boldsymbol{e}} - \vec{\boldsymbol{p}}_{\boldsymbol{i-1}}) \tag{2.10}$$

where the subscript $i - 1, i$ indicates a coordinate transformation from $i - th$ link frame to $(i - 1) - th$ link frame. Finally, from (2.10) it is possible to obtain the final $\boldsymbol{J_{P_i}}$ expression:

$$\boldsymbol{J_{P_i}} = \vec{\boldsymbol{z}}_{\boldsymbol{i-1}} \times (\vec{\boldsymbol{p}}_{\boldsymbol{e}} - \vec{\boldsymbol{p}}_{\boldsymbol{i-1}}) \tag{2.11}$$

where $\boldsymbol{z_i}$ indicates the versor relating to the $i - th$ revolute joint axes.

The calculation of angular velocity contribution $\boldsymbol{J_O}$ can be obtained by expressing the end-effector angular velocity as:

$$\boldsymbol{\omega_e} = \sum_{i=1}^{n} \boldsymbol{\omega_{i-1,i}} = \sum_{i=1}^{n} \boldsymbol{J_{O_i}} \dot{q}_i = \boldsymbol{J_O} \dot{\vec{\boldsymbol{q}}}(t) \tag{2.12}$$

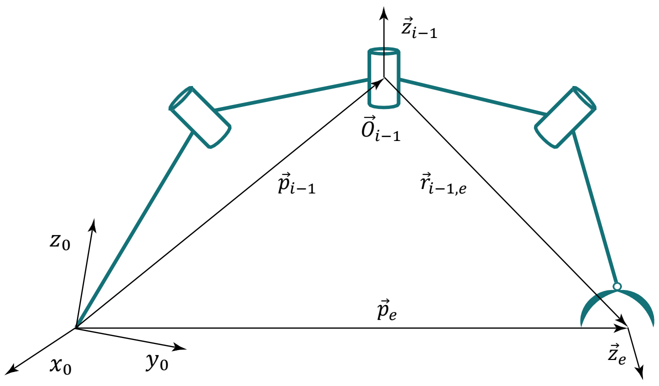


Figure 2.2: Calculation of $i-th$ joint velocity contribution to the end-effector velocity (revised version of Figure 3.2 from [55])

from which it is possible to characterize the angular velocity contribution $\boldsymbol{J_{O_i}}\dot{q}_i$ by using of simple kinematic relation from the rigid body motion theory in [55]

$$\boldsymbol{J_{O_i}}\dot{q}_i = \dot{q}_i\vec{\boldsymbol{z}}_{\boldsymbol{i-1}} \tag{2.13}$$

and thus:

$$\boldsymbol{J_{O_i}} = \vec{\boldsymbol{z}}_{\boldsymbol{i-1}} \tag{2.14}$$

Geometric Jacobian of a 3-DoFs planar manipulator

As practical example of (2.11) and (2.14), the geometric Jacobian of the three-links planar manipulator (Figure 2.1) is reported in below in accordance with [55]

$$\boldsymbol{J(q)} = \begin{bmatrix} \vec{\boldsymbol{z}}_{\boldsymbol{0}} \times (\vec{\boldsymbol{p}}_{\boldsymbol{3}} - \vec{\boldsymbol{p}}_{\boldsymbol{0}}) & \vec{\boldsymbol{z}}_{\boldsymbol{1}} \times (\vec{\boldsymbol{p}}_{\boldsymbol{3}} - \vec{\boldsymbol{p}}_{\boldsymbol{2}}) & \vec{\boldsymbol{z}}_{\boldsymbol{2}} \times (\vec{\boldsymbol{p}}_{\boldsymbol{3}} - \vec{\boldsymbol{p}}_{\boldsymbol{2}}) \\ \vec{\boldsymbol{z}}_{\boldsymbol{0}} & \vec{\boldsymbol{z}}_{\boldsymbol{1}} & \vec{\boldsymbol{z}}_{\boldsymbol{2}} \end{bmatrix} \tag{2.15}$$

where the position vectors of each link are obtained by direct kinematic equation and they result to be:

$$\vec{\boldsymbol{p}}_{\boldsymbol{0}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \vec{\boldsymbol{p}}_{\boldsymbol{1}} = \begin{bmatrix} a_1 cos(q_1) \\ a_1 sin(q_1) \\ 0 \end{bmatrix} \quad \vec{\boldsymbol{p}}_{\boldsymbol{2}} = \begin{bmatrix} a_1 cos(q_1) + a_2 cos(q_1 + q_2) \\ a_1 sin(q_1) + a_2 sin(q_1 + q_2) \\ 0 \end{bmatrix}$$
$$\vec{\boldsymbol{p}}_{\boldsymbol{3}} = \begin{bmatrix} a_1 cos(q_1) + a_2 cos(q_1 + q_2) + a_3 cos(q_1 + q_2 + q_3) \\ a_1 sin(q_1) + a_2 sin(q_1 + q_2) + a_3 sin(q_1 + q_2 + q_3) \\ 0 \end{bmatrix}, \tag{2.16}$$

while the versors of the revolute joint axes are equal to:

$$\vec{\boldsymbol{z}}_{\boldsymbol{0}} = \vec{\boldsymbol{z}}_{\boldsymbol{1}} = \vec{\boldsymbol{z}}_{\boldsymbol{2}} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \tag{2.17}$$

Therefore it is possible to obtain the geometric Jacobian as[1]:

$$\boldsymbol{J} = \begin{bmatrix} -a_1 s_1 - a2 s_{12} - a_3 s_{123} & -a2 s_{12} - a_3 s_{123} & -a_3 s_{123} \\ a_1 c_1 + a2 c_{12} + a_3 c_{123} & a2 c_{12} + a_3 c_{123} & a_3 c_{123} \\ 1 & 1 & 1 \end{bmatrix} \tag{2.18}$$

which is a (3×3)-dimensional matrix, because the manipulator has 3 joints, $n = 3$, while its end-effector on the plane has 2 translation motion variables along X, Y axes and 1 angular motion variable with respect to the revolute joint z-axis joint, so that $m = 3$.

If the end-effector orientation is not of interest for the specific task, it is possible to consider only the positional part $\boldsymbol{J_P}$, which is obtained from the $\boldsymbol{J}$ matrix by extracting the first two rows.

$$\boldsymbol{J} = \boldsymbol{J_P} = \begin{bmatrix} -a_1 s_1 - a2 s_{12} - a_3 s_{123} & -a2 s_{12} - a_3 s_{123} & -a_3 s_{123} \\ a_1 c_1 + a2 c_{12} + a_3 c_{123} & a2 c_{12} + a_3 c_{123} & a_3 c_{123} \end{bmatrix} \tag{2.19}$$

Therefore $\boldsymbol{J}$ in (2.19) is a lower rectangular matrix with dimension equals to (2×3) and the manipulator becomes *functional redundant* in accordance with the definition stated in section 2.3.

2.4.2 Analytical Jacobian

The calculation method of the previous Jacobian $\boldsymbol{J}$ is based on a geometric procedure that determinates the velocity contribution of each joint to the linear

[1] where the notations $c_{i...j}$ and $s_{i...j}$ represent $cos(q_i + \ldots + q_j)$ and $sin(q_i + \ldots + q_j)$, respectively.

and angular velocity components of the end-effector.

However, when the end-effector orientation is defined by a minimum number of parameters (Euler angles) $\vec{\boldsymbol{\phi}}_{\boldsymbol{e}}$, it is very advantageous to calculate the Jacobian matrix by differentiating the direct kinematic relation $\boldsymbol{k}(\vec{\boldsymbol{q}})$ with respect to the joint variable, as shown below

$$\boldsymbol{J_A}(\vec{\boldsymbol{q}}) = \frac{\partial \boldsymbol{k}(\vec{\boldsymbol{q}})}{\partial \vec{\boldsymbol{q}}} \tag{2.20}$$

where $\boldsymbol{J_A}(\vec{\boldsymbol{q}})$ is called the *Analytical Jacobian.* Therefore the direct differential kinematics in (2.8) can be rewritten as

$$\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{e}}(t) = \begin{bmatrix} \dot{\vec{\boldsymbol{p}}}_{\boldsymbol{e}}(t) \\ \dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{e}}(t) \end{bmatrix} = \begin{bmatrix} \boldsymbol{J_P}(\vec{\boldsymbol{q}}(t)) \\ \boldsymbol{J_\phi}(\vec{\boldsymbol{q}}(t)) \end{bmatrix} \dot{\vec{\boldsymbol{q}}}(t) = \boldsymbol{J_A}(\vec{\boldsymbol{q}}(t))\dot{\vec{\boldsymbol{q}}}(t) \tag{2.21}$$

where $\boldsymbol{J_P}$ and $\boldsymbol{J_\phi}$ are the Jacobian that provide the linear and angular velocities contribution to end effector, respectively.

$$\boldsymbol{J_P} = \frac{\partial \vec{\boldsymbol{p}}_{\boldsymbol{e}}}{\partial \boldsymbol{q}} \qquad \boldsymbol{J_\phi} = \frac{\partial \vec{\boldsymbol{\phi}}_{\boldsymbol{e}}}{\partial \boldsymbol{q}} \tag{2.22}$$

From 2.22, it is possible to affirm that the $\boldsymbol{J_A}$ is generally different from $\boldsymbol{J}$, because $\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{e}}$, that represents the rotation velocity vector of end-effector frame, does not coincides with the angular velocity vector $\vec{\boldsymbol{\omega}}_{\boldsymbol{e}}$. However the $\boldsymbol{J_\phi}(\vec{\boldsymbol{q}})$ is not easy to calculate because $\vec{\boldsymbol{\phi}}_{\boldsymbol{e}}$ is not directly accessible, but it is extracted from the rotation matrix of end-effector orientation frame with respect to manipulator base frame.

Therefore it is possible to establish the transformation matrix $\boldsymbol{T}$ between the geometric Jacobian $\boldsymbol{J_P}$ and analytic Jacobian $\boldsymbol{J_A}$ as shown in (2.23)

$$\boldsymbol{J} = \boldsymbol{T_A}(\vec{\boldsymbol{\phi}}_e)\boldsymbol{J_A} \qquad \textit{with} \;\; \boldsymbol{T_A}(\vec{\boldsymbol{\phi}}_e) = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{T}(\vec{\boldsymbol{\phi}}_e) \end{bmatrix} \tag{2.23}$$

where $\boldsymbol{T}(\vec{\boldsymbol{\phi}}_e)$ is the transformation matrix between angular velocity and the time derivative of the Euler angles as shown in (2.24).

$$\boldsymbol{\omega_e} = \boldsymbol{T}(\vec{\boldsymbol{\phi}}_e)\dot{\vec{\boldsymbol{\phi}}}_e \tag{2.24}$$

The transformation matrix $\boldsymbol{T}$ is affected by a *representation singularity* if the end-effector orientation angles make the determinant of $\boldsymbol{T}$ equals to zero, so that the matrix cannot be inverted. In other words, some angular velocity $\boldsymbol{\omega_e}$ cannot be expressed by means of $\dot{\vec{\boldsymbol{\phi}}}_e$, when the orientation of end effector assumes critical Euler angles.

However there are some type of manipulators for which the geometric and analytical Jacobians are completely equivalent. Such equivalence depends on the manipulator's geometry, and in detail, when the DoFs of the structure can assign rotation to the end-effector only respect to a unique fixed axes in the space. A typical example is the planar manipulator shown in Figure 2.1, whose geometry allows the end-effector to solely rotate with respect to $\boldsymbol{z_0}$.

In general the analytical Jacobian should be adopted whenever it is necessary to refer to differential quantities of variables defined in the Cartesian space, while geometric Jacobian should be used when it is necessary to refer to quantities of clear physical meaning.

2.4.3 Inverse differential kinematics

Since generally the task motion variables are defined in the operative space desired by using the $r-\textit{dimensional}$ end-effector vector $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}}$, kinematic inverse algorithms are necessary to obtain the manipulator joints velocity vector, which represents the reference input of the manipulator. If the manipulator is not redundant ($n = r$), the joints velocity vector $\dot{\vec{\boldsymbol{q}}}(t)$ can be calculated by inverting the Jacobian (note that the present study is focused on the analytic Jacobian, but it can be repeated similarly for the geometric Jacobian)

$$\dot{\vec{\boldsymbol{q}}}(t) = \boldsymbol{J_A}^{-1}(\vec{\boldsymbol{q}}(t))\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}}(t) \tag{2.25}$$

If the initial joint position $\vec{\boldsymbol{q}}(0)$ is known, it is possible to calculate the new manipulator configuration $\vec{\boldsymbol{q}}(t)$ by time integration of the joint velocity

$$\vec{\boldsymbol{q}}(t) = \int_0^t \dot{\vec{\boldsymbol{q}}}(s)ds + \vec{\boldsymbol{q}}(\boldsymbol{0}) \tag{2.26}$$

Since the inverse kinematic algorithms generally run on digital processors, the joint position is discretized into $\vec{\boldsymbol{q}}(t_k)$ (where k indicates the k-th discrete time instant) by discrete-time numerical approximation of the continuous-time integral shown in (2.26). In the following the mathematical approach will be based on discrete-time. In order to obtain an accurate discrete-time approximation of (2.26), an high-order algorithm is required. However, an high-order algorithm implicates an undesired large finite time delay in real time applications, which can be reduced by shortening the time step Δt [40]. A good choice consists in a first order algorithm as the Euler forward rectangular method, which can give an acceptable accuracy of the numerical integration with suitable Δt value. The Euler forward rectangular method is used to transform the integral shown in (2.26) in the following recursive form

$$\vec{\boldsymbol{q}}_k \approx \vec{\boldsymbol{q}}_{k-1} + \boldsymbol{J_A}^{-1}(\vec{\boldsymbol{q}}_{k-1})\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_{k-1}}\Delta t \tag{2.27}$$

where $\vec{\boldsymbol{q}}_k$ indicates the calculated joints position at the time instant k, while $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_{k-1}}$ is the desired end-effector motion at the time instant $k-1$. In spite of the kind of implemented interpolation, the obtained end effector pose $\vec{\boldsymbol{x}}_{e_k}$, by direct kinematic functions, is different from the desired end-effector pose $\vec{\boldsymbol{x}}_{\boldsymbol{d}_k}$, because any numerical integration is mainly affected by two sources of error. The first consists in an unavoidable drifting error, which increases at each numerical integration step, while the second depends on the uncertainty in the initial value of the joints position. Thus, the pose error $\vec{\boldsymbol{e}}_k$ can be defined as

$$\vec{\boldsymbol{e}}_k = \vec{\boldsymbol{x}}_{\boldsymbol{d}_k} - \vec{\boldsymbol{x}}_{e_k} \tag{2.28}$$

Defining the joint velocity vector as a function of the error of the end-effector pose $\dot{\vec{\boldsymbol{q}}}_k(\vec{\boldsymbol{e}}_k)$, it is possible to ensure the convergence of the pose error to zero. This choice permits to find several inverse kinematics algorithms which are based on the use of a feedback correction term called Closed-Loop Inverse Kinematics (CLIK) [56]. A well known first-order kinematic algorithm is the Jacobian inverse, which permits to calculate $\dot{\vec{\boldsymbol{q}}}_k$ as formally defined in [53, 55]

$$\dot{\vec{\boldsymbol{q}}}_k = \boldsymbol{J_A}^{-1}(\vec{\boldsymbol{q}}_k)(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k} + \boldsymbol{K}\vec{\boldsymbol{e}}_k) \tag{2.29}$$

therefore, (2.29) can be inserted in the control algorithm, whose block diagram representation is shown in Figure 2.3.

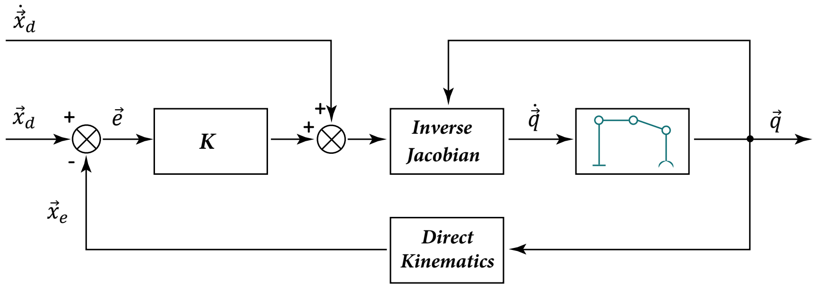


Figure 2.3: Block diagram of the close loop inverse kinematics (partial reproduction of Figure 3.11 from [55])

in which $\boldsymbol{K}$ is a constant positive-define gain matrix. Since $\dot{\vec{\boldsymbol{q}}}_k = \boldsymbol{J_A}^{-1}(\vec{\boldsymbol{q}}_k)\dot{\vec{\boldsymbol{x}}}_k$, (2.29) can be rewritten as follows

$$\boldsymbol{J_A}^{-1}(\vec{\boldsymbol{q}}_k)(\dot{\vec{\boldsymbol{e}}}_k + \boldsymbol{K}\vec{\boldsymbol{e}}_k) = \mathbf{0} \rightarrow \lim_{k\rightarrow\infty} \|\vec{\boldsymbol{e}}_k\| = 0 \tag{2.30}$$

where the velocity of the pose error convergence depends on the eigenvalues of matrix $\boldsymbol{K}$, so that larger eigenvalues values correspond to faster pose error convergence. However, the convergence velocity cannot be chosen arbitrarily due to the limitations imposed by the Δt. Finally, if the initial error pose is equal to zero, i.e., $\vec{\boldsymbol{e}}_{k=0} = \mathbf{0}$, then the feed-forward action ensures a zero error along the whole trajectory.

When the manipulator is redundant, the inverse differential kinematics in (2.29) cannot be implemented, because the Jacobian matrix is a lower rectangular matrix (its dimension is $r \times n$ with $r < n$). Therefore its inversion admits multiple solutions and a criteria of solution choice should be adopted, as it is described in the next chapter.

Chapter 3

Redundancy resolution via local optimization

This chapter presents the redundancy resolution methods that were considered during my research activity to solve the inverse differential kinematic problems of the redundant manipulator. Indeed when a manipulator is redundant, its Jacobian matrix $\boldsymbol{J_A}$ has more columns than rows (r < n) and thus infinite solutions exist to (2.29), hence a chosen solution criteria is required. The redundancy resolution methods allows to obtain the desired solution among all those possible, via local optimization of a performance criterion or other tasks having lower priority execution than the main task (or primary tasks).

The great advantage of local optimization methods consists in their simple redundancy resolution form, which permits the real-time calculation of the $k-th$ desired joint velocity vector $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{k}}$ in order to optimize locally a function objective $w(\vec{\boldsymbol{q}}_k)$. Since the solutions obtained are based only on current manipulator information, such as configuration and desired end-effector velocity, the local optimal solution is not guaranteed over long tasks (e.g., if the performance criterion is based on the manipulability measure, the singularity avoidance is not guaranteed along the whole manipulator motion). However, the local optimization methods have a low computational effort that permits their use in the inverse differential kinematic scheme of real-time application.

3.1 Jacobian pseudo-inverse method

The first local redundancy resolution method consists of finding the solution that locally minimize the norm of joint velocity vector, $||\dot{\vec{\boldsymbol{q}}}||$. The reason of this choice is due to high joint velocity values that occur when a manipulator assumes a configuration close to the singularity. In accordance with [55], it is possible to find a particular solution $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{k}}$, which minimizes a quadratic cost functional of joint velocities $g(\dot{\vec{\boldsymbol{q}}})$ in (3.1) by using the end-effector velocity $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}}$

and analytic Jacobian J_A associated with a given manipulator configuration.

$$g(\dot{\vec{\boldsymbol{q}}}) = \frac{1}{2}\dot{\vec{\boldsymbol{q}}}^T \boldsymbol{W}\dot{\vec{\boldsymbol{q}}} \tag{3.1}$$

In particular, the (3.1) shows a suitable $(n \times n)$ weighting matrix, which is a symmetric positive definite matrix. The (3.1) can be solved by using of *Lagrange multipliers method*, by introducing the following modified cost functional

$$g(\dot{\vec{\boldsymbol{q}}}, \vec{\boldsymbol{\lambda}}) = \frac{1}{2}\dot{\vec{\boldsymbol{q}}}^T \boldsymbol{W}\dot{\vec{\boldsymbol{q}}} + \vec{\boldsymbol{\lambda}}^T(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}} - \boldsymbol{J_A}\dot{\vec{\boldsymbol{q}}}) \tag{3.2}$$

where $\vec{\boldsymbol{\lambda}}$ is a unknown $(r \times 1)$ vector of multipliers that includes the constraint (2.29) in the functional to minimize. Therefore the desired solution has to satisfy the following conditions:

$$\left(\frac{\partial g}{\partial \dot{\vec{\boldsymbol{q}}}}\right)^T = \mathbf{0} \qquad \left(\frac{\partial g}{\vec{\boldsymbol{\lambda}}}\right)^T = \mathbf{0}. \tag{3.3}$$

The solution obtained from the first condition is equal to $\boldsymbol{W}\dot{\vec{\boldsymbol{q}}} - \boldsymbol{J}^T\vec{\boldsymbol{\lambda}} = \mathbf{0}$, which can be rewritten as

$$\dot{\vec{\boldsymbol{q}}} = \boldsymbol{W}^{-1}\boldsymbol{J_A}^T\vec{\boldsymbol{\lambda}} \tag{3.4}$$

because $\boldsymbol{W}$ is invertible. From (3.4), it is possible observe that this solution is a minimum, indeed $\frac{\partial^2 g}{\partial^2 \dot{\vec{\boldsymbol{q}}}} = \boldsymbol{W}$ is positive definite. On the other side, the second condition in (3.3) provides the following constraint:

$$\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}} = \boldsymbol{J_A}\dot{\vec{\boldsymbol{q}}} \tag{3.5}$$

which corresponds to direct kinematics. Combining the two solutions, it is possible to obtain the following relation:

$$\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}} = \boldsymbol{J_A}\boldsymbol{W}\boldsymbol{J_A}^T\vec{\boldsymbol{\lambda}} \tag{3.6}$$

By assuming that analytical Jacobian $\boldsymbol{J_A}$ has full rank, the $\boldsymbol{J_A}\boldsymbol{W}\boldsymbol{J_A}^T$ is an square matrix with dimension equals to $(r \times r)$ and rank r, so that, it is invertible. Finally, by solving the (3.6) respect with $\vec{\boldsymbol{\lambda}}$, it is possible to determine the expression shown in (3.7).

$$\vec{\boldsymbol{\lambda}} = (\boldsymbol{J_A}\boldsymbol{W}\boldsymbol{J_A}^T)^{-1}\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}} \tag{3.7}$$

Therefore, the desired optimal solution is obtained by replacing the 3.7 into

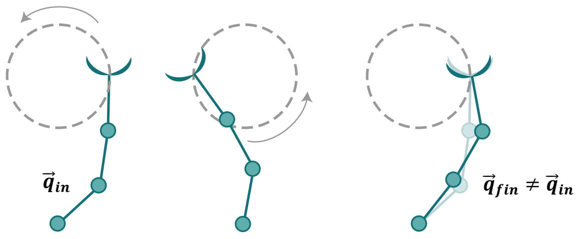


Figure 3.1: Example of non-repeatable motion in the joint space

(3.4) as shown bellow:

$$\dot{\vec{q}} = \boldsymbol{W}^{-1}\boldsymbol{J_A}^T(\boldsymbol{J_A}\boldsymbol{W}\boldsymbol{J_A}^T)^{-1}\dot{\vec{x}}_{\boldsymbol{d}}. \tag{3.8}$$

Pre-multiplying both sides of (3.8) by $\boldsymbol{J_A}$, it is possible to demonstrate that it satisfies the differential kinematics defined in (2.29). In order to locally minimize the norm of solution, $||\dot{\vec{q}}||$, it is possible to impose the weighting matrix $\boldsymbol{W}$ equals to the identity matrix $\boldsymbol{I}$:

$$\dot{\vec{q}} = \boldsymbol{J_A}^{\dagger}\dot{\vec{x}}_{\boldsymbol{d}} \tag{3.9}$$

As shown in (3.9), the minimum solution is obtained by $\boldsymbol{J_A}^{\dagger}$ matrix, which is defined as the right pseudo-inverse of $\boldsymbol{J_A}$ in accordance with Moore-Penrose properties. It is obtained as follows

$$\boldsymbol{J_A}^{\dagger} = \boldsymbol{J_A}^T(\boldsymbol{J_A}\boldsymbol{J_A}^T)^{-1} \tag{3.10}$$

However the pseudo-inverse of the Jacobian can be computed only when the matrix has full rank, so that, it becomes meaningless when the manipulator is at a singular configuration (the Jacobian matrix contains linearly dependent equations) or also in the neighbourhood of a singularity. Indeed, since it is well known that the computation of the inverse Jacobian matrix requires the determinant matrix value, it becomes a relatively small value in the neighbourhood of a singular configuration, so that, the joint velocity take high velocity values. In this case, the Singular Value Decomposition (SVD) of $\boldsymbol{J_A}$ can be a valid approach to overcome the problem above described in [55]. An alternative method consists of calculating the Damped Least-Squares (DLS) inverse of the Jacobian matrix $\boldsymbol{J_A}^{\#}$, as following defined:

$$\boldsymbol{J_A}^{\#} = \boldsymbol{J_A}^T(\boldsymbol{J_A}\boldsymbol{J_A}^T + k^2\boldsymbol{I})^{-1} \tag{3.11}$$

where k is a damping factor. However, even when the Jacobian has full rank, the (3.10) does not guarantee the repeatability of the end-effector motion. In particular, the example shown in Figure 3.1) demonstrates the loss of motion repeatability (in the joint space) of a planar manipulator, whose final manipulator configuration $\boldsymbol{q_{fin}}$ assumed after one tour is different from the initial manipulator configuration $\boldsymbol{q_{in}}$. Moreover, it is yet affected by drift error due to Euler integration (it is directly proportional to simple time value δt). For this reason it is possible to introduce the Jacobian pseudo-inverse $\boldsymbol{J_A}^{\dagger}(\vec{\boldsymbol{q}}_k)$ into the (2.29) in order to obtain the following CLIK expression based on the Jacobian pseudo inverse formulation.

$$\dot{\vec{\boldsymbol{q}}}_k = \boldsymbol{J_A}^{\dagger}(\vec{\boldsymbol{q}}_k)(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k} + \boldsymbol{K}\vec{\boldsymbol{e}}_k) \tag{3.12}$$

3.2 Jacobian null space method

3.2.1 Jacobian null space projection matrix

The simple CLIK pseudo-inverse in (3.12) does not allow to manage the redundant joint motions (motion variables that are not required by the task), which could be used to perform some performance criteria (or secondary tasks), such as: joint limits avoidance, obstacle avoidance and manipulability index optimization.

In order to ensure the performance of the primary task, a possible strategy consists in projecting the redundant joints velocity vector requested by the secondary tasks, namely $\dot{\vec{\boldsymbol{q}}}_k^{+}$, in the Jacobian null space, $N(\boldsymbol{J_A})$, which is a subspace of $\boldsymbol{J_A}$ having a dimension equals to the degree of redundancy dr, as defined in (2.7). Since the Jacobian null space is an orthogonal space of the

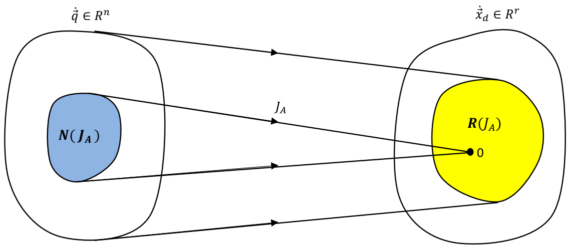
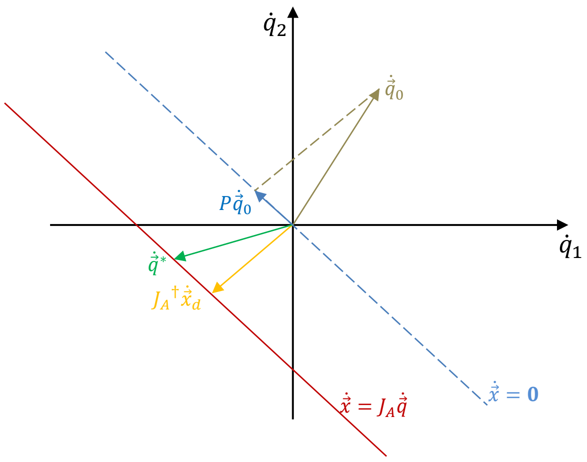


Figure 3.2: Relation between joint velocity space and end-effector velocity space



$\dot{q}_2$
$\dot{\vec{q}}_0$
$P\dot{\vec{q}}_0$
$\dot{q}_1$
$\dot{\vec{q}}^*$
$J_A{}^\dagger\dot{\vec{x}}_d$
$\dot{\vec{x}} = \mathbf{0}$
$\dot{\vec{x}} = J_A\dot{\vec{q}}$

Figure 3.3: Geometric view on Jacobian null space

Jacobian image space, namely $R(\boldsymbol{J_A})$, the joint velocities in Jacobian null space do not produce any velocity on the end-effector. Indeed these joint velocities are linearly mapped by the analytical Jacobian to the end-effector velocity value equals to zero in the Jacobian image space (or Jacobian rank space), as demonstrated in Figure 3.2.

In order to project a secondary task defined by $\dot{\vec{\boldsymbol{q}}}_k^+$ vector into the Jacobian null space of the primary task, it is possible to use the $n \times n$ dimensional projector matrix $\boldsymbol{P}$, such that

$$\boldsymbol{J_A}(\vec{\boldsymbol{q}}_k)\boldsymbol{P}_k\dot{\vec{\boldsymbol{q}}}_k^+ = \mathbf{0} \tag{3.13}$$

In particular the $\boldsymbol{P}$ can be obtained at each time instant by:

$$\boldsymbol{P}_k = \boldsymbol{I} - \boldsymbol{J_A}^\dagger(\vec{\boldsymbol{q}}_k)\boldsymbol{J_A}(\vec{\boldsymbol{q}}_k) \tag{3.14}$$

where $\boldsymbol{I}$ indicates the identity matrix of suitable dimensions, while the rank of the $\boldsymbol{P}$ matrix is equal to the redundancy degree dr.

The $\boldsymbol{P}$ matrix possesses several properties, as being symmetric ($p_{ji} = p_{ij}$), idempotent ($\boldsymbol{P}^2 = \boldsymbol{P}$) and Hermitian ($\boldsymbol{P}^{-1} = \boldsymbol{P}$).

By adding the orthogonal projection of $\dot{\vec{\boldsymbol{q}}}_{\mathbf{0}}$ in (3.12), it is possible to obtain the new CLIK expression

$$\dot{\vec{\boldsymbol{q}}}_k^* = \boldsymbol{J_A}^\dagger(\vec{\boldsymbol{q}}_k)(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k} + \boldsymbol{K}\vec{\boldsymbol{e}}_k) + \boldsymbol{P}_k\dot{\vec{\boldsymbol{q}}}_{0_k} \tag{3.15}$$

where the $\dot{\vec{\boldsymbol{q}}}_k^*$ solution is different from the one obtained from (3.12), because now it is the minimal norm joints velocity solution that satisfies both the primary task and the secondary tasks. In particular, a better understanding of (3.12) can be given by the geometric representation in Figure 3.3 (where the $\boldsymbol{J_A}^{\dagger}$ is replaced with $\boldsymbol{J}^{\#}$). In detail, the Figure 3.3 shows a 2-dimensional Joint space, in which the final solution $\dot{\vec{\boldsymbol{q}}}^{*}$ (red vector) is obtained by the vectorial sum of two vector: the first one is the joint velocity vector with minimum norm (orange vector), while the second one is the orthogonal projection of the secondary task velocity into the Jacobian null space (green vector). Therefore, it is possible to note that the final solution is closest to $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{0}}$.

3.2.2 Gradient projection method

In the previous section, the $\dot{\vec{\boldsymbol{q}}}_0$ vector has been assumed to be arbitrary. A typical choice of $\dot{\vec{\boldsymbol{q}}}_{0_k}^{+}$ value is based on the Gradient Projection Method (GPM) [42]. In detail, this method permits of projecting the gradient of a specific objective function $w(\vec{\boldsymbol{q}}_k)$ (in the joints space), into the Jacobian null space relative to the primary task. Therefore the $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{0}}$ results to be equal to:

$$\dot{\vec{\boldsymbol{q}}}_{0_k}^{+} = k_a \nabla w(\vec{\boldsymbol{q}}_k) = k_a \left(\frac{\partial w(\vec{\boldsymbol{q}}_k)}{\partial \vec{\boldsymbol{q}}} \right)^T \tag{3.16}$$

where k_a is a real scalar value which indicates the gain, and it is a positive value when $w(\vec{\boldsymbol{q}}_{\boldsymbol{k}})$ has to be maximized, otherwise is a negative value if $w(\vec{\boldsymbol{q}}_{\boldsymbol{k}})$ has to be minimized [42]. However, the choice of the k_a is critical for the performance of the redundancy resolution. In particular, a small value of the step size may slow down the minimization of the performance criteria $w(\vec{\boldsymbol{q}}_k)$, while a its large value may even lead to an increase it. An appropriate value of k_a could be chosen by using of in a simplified line search technique as described in [57].

By replacing the (3.16) in (3.15), the complete GPM expression is obtained:

$$\dot{\vec{\boldsymbol{q}}}_k^{*} = \boldsymbol{J_A}^{\dagger}(\vec{\boldsymbol{q}}_k)(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k} + \boldsymbol{K}\vec{\boldsymbol{e}}_k) + \boldsymbol{P}_k(k_a \nabla_q w(\vec{\boldsymbol{q}}_k)) \tag{3.17}$$

However the Jacobian pseudo-inverse $\boldsymbol{J_A}^{\dagger}$ is yet present in (3.17), hence the GPM can be computationally intensive, because the SVD has to be generally used. In case the rank of the Jacobian is equal to the task dimension, an alternative method for solving the (3.17) is based on the observation that the redundancy degree is dr. Therefore, the search of the optimum solution $\dot{\vec{\boldsymbol{q}}}_k^{*}$ can be more efficiently performed within a reduced space of the joint variables, which lead to the Reduced Gradient Method RGM ([58]), when $rank(\boldsymbol{J_A}) = r$. The great advantage of RGM is the use of the simple Jacobian inversion $\boldsymbol{J_A}^{-1}$ that makes it analytically simpler and numerically faster than GPM, but requires the search for a non-singular minor of the robot Jacobian.

3.2.3 Objective functions

Typical objective functions $w(\vec{\boldsymbol{q}}_k)$ that can be locally optimized in (3.17) are now presented in accordance with the definition reported in [55]

Manipulability measure

The manipulability measure can be defined as:

$$w(\vec{\boldsymbol{q}}_k) = \sqrt{det(\boldsymbol{J_A}(\vec{\boldsymbol{q}}_{\boldsymbol{k}})\boldsymbol{J_A}(\vec{\boldsymbol{q}}_{\boldsymbol{k}})^T)} \tag{3.18}$$

which measures the distance between the current manipulator configuration and its singular configuration. Therefore, it is possible to maximize this measure in order to move it away from singular configurations.

Joint limits avoidance

The maximum (or minimum) joint limits q_{iM} (q_{im}) represents critical motion constraints for any type of robotic application, because they can degrade the performance of the system and also damage it. Hence the joint limits can be avoided by minimizing the distance $w(\vec{\boldsymbol{q}}_k)$ between the current joint positions and their middle value of the joint position range ($\bar{q}_i$) .

$$w(\vec{\boldsymbol{q}}_k) = -\frac{1}{2n}\sum_{i=1}^{n}\left(\frac{q_i - \bar{q}_i}{q_{iM} - q_{im}}\right)^2 \tag{3.19}$$

The particular solution $\dot{\vec{\boldsymbol{q}}}^{+}_{0_k}$ that minimize the measure $w(\vec{\boldsymbol{q}}_k)$ is obtained by setting a negative value to k_a gain.

Obstacle avoidance

Finally, the obstacle avoidance task is very important for those applications that are executed in unstructured environment, because it permits to avoid accidental collision with obstacle placed into the workspace. This task is obtained by maximizing the distance $w(\vec{\boldsymbol{q}}_k)$ between the position vector of any point on the obstacle, defined as $\vec{\boldsymbol{o}}$, and the position vector of a generic point on the manipulator, namely $\vec{\boldsymbol{p}}$.

$$w(\vec{\boldsymbol{q}}_k) = min\|\vec{\boldsymbol{p}}(\vec{\boldsymbol{q}}) - \vec{\boldsymbol{o}}\| \tag{3.20}$$

3.2.4 Saturation in the null space method

The kinematic inversion obtained by means of pseudo-inverse does not explicitly take into account the limits of the joint, such as speed and acceleration limits.

Therefore the considered assumption is that the assigned task is compatible with the robot limits. Indeed the redundancy resolution with optimization tasks (typically with the GPM algorithm) does permit to guarantee the robots limits, because they are considered as secondary task, and thus, they have a lower execution priority than the main task. Therefore it is necessary to examine whether there are alternative motions of the joints that do not violate the joints limits and, at the same time, can perform correctly the specified task.

A possible solution concerns the algorithm proposed in ([48]),which is called Saturation Null Space algorithms (SNS). In detail the saturation of $l \leq (n - r)$ joints that exceed their limits is compensated as far as possible from the remaining joint still not saturated. In particular, the unperformed velocity of the most critical joint is projected into the reduced relative Jacobian null space, which is composed by the healthy joints. The new obtained manipulator pose allows to complete the task without affecting the relative end effectors motion.

Therefore the Jacobian pseudo-inverse based on SNS method can be so expressed:

$$\dot{\vec{\boldsymbol{q}}}_k^* = s(\boldsymbol{J_A W})^\dagger(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k} + \boldsymbol{K}\vec{\boldsymbol{e}}_k) + (\boldsymbol{I} - (\boldsymbol{J_A W})^\dagger \boldsymbol{J_A})\dot{\vec{\boldsymbol{q}}}_{f_k} \tag{3.21}$$

where $W \in [0;1]$ is a diagonal binary matrix, which is used for zeroing the column of the Jacobian associated to the saturated joint presents in $\dot{\vec{\boldsymbol{q}}}_{f_k}$ vector. Moreover, $s \in [0,1]$ is the trajectory scaling factor, which is used to reduce the end-effector velocity when the number of healthy joints is no longer sufficient to compensate the saturated joints velocity. Therefore, the scaling factor to be kept equal to 1 in order to preserve the desired end-effector motion, otherwise it is reduced just enough to make the task feasible considering the limitations. The following is an illustrative example of the operation of the algorithm with velocity saturation in the null-space.

Let consider the planar manipulator shown in Figure 3.4 having equal links of unitary length, that has to perform a task specified by a desired end-effector linear velocity, $\dot{\vec{\boldsymbol{x}}} = [2.5, -1]^T$ (with $r = 2$). Since the dimension of the joint velocity vector $\dot{\vec{\boldsymbol{q}}}$ is equal $n = 4$, the planar manipulator results to have a degree of redundancy equals to $dr = n - r = 2$.

By supposing that the joint velocities are bounded as $|\dot{q}_i| \leq V_i$, $i = 1, \ldots, 4$ with $V_1 = V3 = 1$, $V_2 = V_4 = 4[rad/s]$, and that the current configuration is equal to $\vec{\boldsymbol{q}} = [\pi/2, -\pi/4, -\pi/3, \pi/4]^T$, the joint velocity vector calculated by simple Jacobian pseudo-inverse (see 3.12), results to be:

$$\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{PS}} = [-1.6, 0.4, 1.2, -0.2]^T[rad/s] \tag{3.22}$$

The (3.22) shows that the first and third joint are saturated. A better solution can be found by applying the SNS algorithm, in which is saturated only

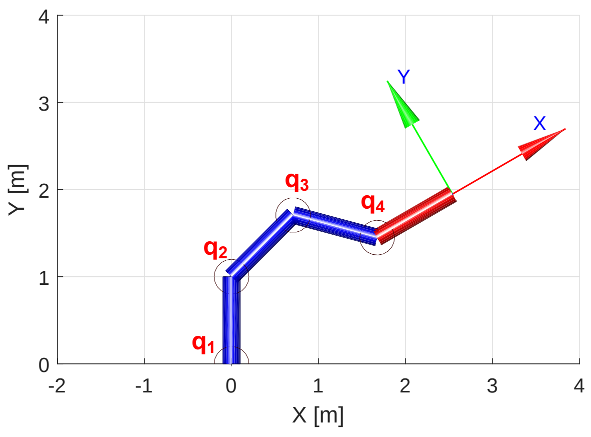


Figure 3.4: Example of SNS application

the most violating joint velocity, in this case $\dot{q}_f = [V_1, 0, 0, 0]$, therefore it is possible to calculate the joint velocity vector based on SNS method, $\dot{\vec{q}}_{SNS}$, by using (3.21):

$$\dot{\vec{q}}_{SNS} = [-1, -0.5, 2.1, -1.1]^T [rad/s] \tag{3.23}$$

in this case the third joint velocity is increased so that its limit results to be violated yet. Since the system has two redundant motions, it is possible to guarantee the third joint limit, yet. Therefore it is necessary to set the $\dot{q}_f$ vector as $\dot{q}_f = [V_1, 0, V_3, 0]$, so that, the new solution is equal to:

$$\dot{\vec{q}}_{SNS} = [-1, 1.2, 1, -3.9]^T [rad/s] \tag{3.24}$$

Now all joints respect their limits. However, the forth joint velocity is much increased, so that, it is close to its limit. Since now the number of violated joints is equal to the degree of redundancy, an another violated joint could not be guaranteed. In this case, the sns method permits to scale the original task velocity by using the scaling factor s.

3.2.5 Hierarchical priority execution task

Redundancy can be exploited to perform one or more secondary tasks, which possess lower execution priority with respect to the main task. The maximum number of tasks l that can be simultaneously executed by a robot, depends on the DoF possessed by manipulator n and from rank rk of the Jacobian associated with each task ([43]). Therefore, when choosing tasks in a nonconflicting way, it is possible to add tasks until

$$\sum_{i=1}^{l} rk_i = n \tag{3.25}$$

In order to avoid task conflicts, [44] introduced a hierarchic prioritized task architecture, in which lower priority tasks are directly projected on the null space of the higher priority ones. In this way, the lower priority tasks do not affect performance of the higher priority tasks and the performance of the highest priority task is always guaranteed. Moreover this architecture is a singularity-robust approach, because the singularity are limited to secondary tasks (with consequently degradation of their performance) without affect the primary task, as long as the sole primary-task Jacobian matrix is full rank.

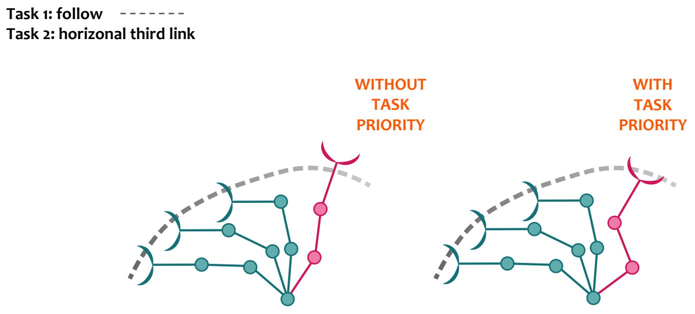


Figure 3.5: Example of hierarchical priority execution task

Therefore, given three generic prioritized tasks $\dot{\boldsymbol{x}}_{\boldsymbol{1}}$, $\dot{\boldsymbol{x}}_{\boldsymbol{2}}$ and $\dot{\boldsymbol{x}}_{\boldsymbol{3}}$ (where the subscript having highest value indicates the highest priority task), it is possible to obtain the following desired joint velocity vector $\dot{\vec{\boldsymbol{q}}}_{des}$

$$\dot{\vec{\boldsymbol{q}}}_{des} = \boldsymbol{J}_3{}^{\dagger}\dot{\boldsymbol{x}}_{\boldsymbol{3}} + \boldsymbol{P}_3\left(\boldsymbol{J}_2{}^{\dagger}\dot{\boldsymbol{x}}_{\boldsymbol{2}} + \boldsymbol{P}_2\boldsymbol{J}_1{}^{\dagger}\dot{\boldsymbol{x}}_{\boldsymbol{1}}\right) \tag{3.26}$$

where $\boldsymbol{J}_i{}^{\dagger}$ $(i = 1, 2, 3)$ is the pseudo inverse of each task Jacobian, while $\boldsymbol{P}_k$ $(k = 2, 3)$ indicates the orthogonal projector on the null space $\boldsymbol{N}_k$, that is obtained by $\boldsymbol{I} - \boldsymbol{J}_k^{\dagger}\boldsymbol{J}_k$ (where $\boldsymbol{I}$ indicates the identity matrix).

In order to have a better understanding about the priority execution task, the following simple example is described. In particular, a 3-DoF planar manipulator has to follow a specific path by keeping the third link in horizontal orientation. Therefore, this task could be divided in two sub-tasks with different priory execution level. In detail, the path-following is considered as primary task, while the orientation of the third link is considered as secondary task, and it is projected in the Jacobian null space of the primary task. Therefore, the Figure 3.5 shows two cases illustrating the end-effector motion without and with task priority architecture, respectively. In particular, when the assigned path does not permit physically to obtain an horizontal orientation of the third link, the manipulator is not able to track correctly the path in the first case. On the other hand, the priority task architecture decomposes the task into two separate sub-tasks, in order to guarantee the performance of the highest priority task, even when the lowest task cannot be performed.

Chapter 4

Kinematic modelling of cooperative manipulators

When two manipulators interact on an object at the same time, they make a closed-chain among the three parts, so that generally this task requires less motion variables respect to those provided by two independent manipulators. Therefore the two cooperative manipulators can be seen as a single redundant manipulator having the same degree of motion of the dual-arm and the same number of end-effector motion variables required by the task. In this way, it is possible to implement the same control algorithm adopted in the single redundant manipulator case, as discussed in the previous chapter, with the only difference of redefining the Jacobian matrix. The Jacobian matrix associated with the equivalent manipulator depends on the two individual analytic Jacobians possessed by each manipulator, and it is call *relative Jacobian*.

This chapter presents a the derivation of a compact relative Jacobian formulation and the inverse kinematic solution based on the projection into the relative Jacobian null space.

4.1 Relative Jacobian method

This section describes the derivation of the Relative Jacobian method in accordance with the compact expression introduced by ([26]). In detail, it is possible to consider two generic cooperative manipulators shown in Figure 4.1 [53], namely A and B. They posses a number of joints respectively equal to n_a and n_b, with corresponding Jacobians $\boldsymbol{J_A} = [J_{pA}, J_{oA}]^T$ and $\boldsymbol{J_B} = [J_{pB}, J_{oB}]^T$, that are expressed in terms of the position and orientation velocity components.

The manipulator A possesses the role of master, thus all coordination frames have to be expressed with respect to end-effector A, which is denoted as A_e. In detail the coordinate transformations of the frame bases, A_b and B_b, are performed by the rotation matrices $\boldsymbol{R^{A_e}_{A_b}}$ and $\boldsymbol{R^{A_e}_{B_b}}$, in accordance with the generic notation $\boldsymbol{R^j_i}$ that defines the rotation matrix of the frame i with respect to frame j. Regarding the frame of the end-effector B (denoted as B_e), it is

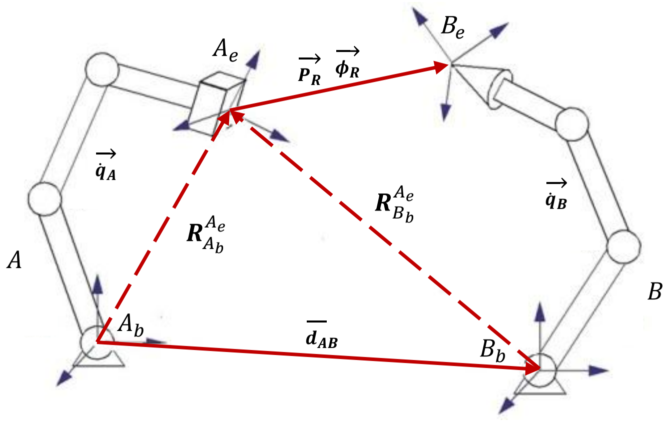


Figure 4.1: Two cooperative manipulators (partial reproduction of Figure 1 from [26])

expressed with respect to the A_e frame by vectors $\vec{\boldsymbol{P}}_{\boldsymbol{R}}$ and $\vec{\boldsymbol{\phi}}_{\boldsymbol{R}}$, which define the end effectors' relative position by using the Cartesian coordinates and relative orientation by using a minimum orientation representation, respectively.

Differentiating these two vectors with respect to time, it is possible to define a column vector $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k}$, which contains the end-effectors' relative velocity components at time instant k:

$$\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k} = \begin{bmatrix} \dot{\vec{\boldsymbol{P}}}_{\boldsymbol{R}} \\ \dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{R}} \end{bmatrix} \tag{4.1}$$

where $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k}$ has a dimension equals to r_{ab} (where $r_{ab} \leq 6$ in a three dimensional space).

In order to obtain the differential kinematics, $\dot{\vec{\boldsymbol{x}}} = \boldsymbol{J_A}\dot{\vec{\boldsymbol{q}}}$ (see 2.21), it is possible to define an equivalent manipulator having a number of joints equal to $n_{ab} = n_a + n_b$ and end-effector motion variables defined by the $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k}$ vector.

At this point, it is necessary to define the Jacobian matrix associated to the equivalent manipulator. A possible solution consists of expressing the relative end-effectors motions vectors, $\dot{\vec{\boldsymbol{P}}}_{\boldsymbol{R}}$ and $\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{A}}$, as a function of the joint velocity vectors, $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a}}$ and $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{b}}$.

In detail, the $\dot{\vec{\boldsymbol{P}}}_{\boldsymbol{R}}$ is obtained as the difference between the two independent end-effector velocity $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{A_k}}$ and $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{B_k}}$, which are referred to A_e frame, by means of the two rotation matrices $\boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}}$ and $\boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}}$. Moreover, it is present a term that permits of compensating the linear velocity contribution due to the A_e frame orientation velocity $\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{A}}$, which is obtained by the cross product of $\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{A}}$ with $\vec{\boldsymbol{P}}_{\boldsymbol{R}}$

vector.

$$\dot{\vec{\boldsymbol{P}}}_{\boldsymbol{R}} = \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}} \dot{\vec{\boldsymbol{x}}}_{\boldsymbol{B_k}} - \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \dot{\vec{\boldsymbol{x}}}_{\boldsymbol{A_k}} + \vec{\boldsymbol{P}}_{\boldsymbol{R}} \times \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{A}}. \tag{4.2}$$

Since a cross product of a vector can be expressed as a skew-symmetric matrix with input the same vector, it is possible to define the following skew-symmetric matrix $\boldsymbol{S}(\vec{\boldsymbol{P}}_{\boldsymbol{R}})$:

$$\boldsymbol{S}(\vec{\boldsymbol{P}}_{\boldsymbol{R}}) = \begin{bmatrix} 0 & -P_{R_z} & P_{R_y} \\ P_{R_z} & 0 & -P_{R_x} \\ -P_{R_y} & P_{R_x} & 0 \end{bmatrix}. \tag{4.3}$$

Regarding the relative end-effector orientation velocity $\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{R}}$, it is obtained by the difference between the two single end-effectors orientation velocities expressed with respect to A_e frame:

$$\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{R}} = \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}} \dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{B}} - \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{A}}. \tag{4.4}$$

Inserting the (4.2) and (4.4) in (4.1) and explicating the joint velocity vectors, $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a}}$ and $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{b}}$, it is possible to obtain the differential kinematics in the compact matrix formulation, as shown in [26]:

$$\begin{aligned}
\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k} &= \begin{bmatrix} \dot{\vec{\boldsymbol{P}}}_{\boldsymbol{R}} \\ \dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{R}} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \boldsymbol{J_{pA}} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a_k}} + \boldsymbol{S}(\vec{\boldsymbol{P}}_{\boldsymbol{R}}) \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \boldsymbol{J_{oA}} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a_k}} + \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}} \boldsymbol{J_{pB}} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{b_k}} \\ -\boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \boldsymbol{J_{oA}} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a_k}} + \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}} \boldsymbol{J_{oB}} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{b_k}} \end{bmatrix} \\
&= \begin{bmatrix} -\boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \boldsymbol{J_{pA}} + \boldsymbol{S}(\vec{\boldsymbol{P}}_{\boldsymbol{R}}) \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \boldsymbol{J_{oA}} + \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}} \boldsymbol{J_{pB}} \\ -\boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \boldsymbol{J_{oA}} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a_k}} + \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}} \boldsymbol{J_{oB}} \end{bmatrix} \begin{bmatrix} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a_k}} \\ \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{b_k}} \end{bmatrix} \\
&= \left[\begin{bmatrix} \boldsymbol{I} & -\boldsymbol{S}(\vec{\boldsymbol{P}}_{\boldsymbol{R}}) \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} -\boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{A_b}} \end{bmatrix} \begin{bmatrix} \boldsymbol{J_{pA}} \\ \boldsymbol{J_{oA}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^{\boldsymbol{A_e}}_{\boldsymbol{B_b}} \end{bmatrix} \begin{bmatrix} \boldsymbol{J_{pB}} \\ \boldsymbol{J_{oB}} \end{bmatrix} \right] \\
&\times \begin{bmatrix} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a_k}} \\ \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{b_k}} \end{bmatrix}
\end{aligned} \tag{4.5}$$

Therefore the final differential kinematic relation between the end-effector motion variables $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k}$ and the joints velocity indicated by the n_{ab} - dimensional column vector $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{ab}_k} = [\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{a}_k}, \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{b}_k}]^T$, can be defined as:

$$\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k} = \boldsymbol{J_R}^{\dagger}(\vec{\boldsymbol{q}}_{\boldsymbol{ab}_k}) \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{ab}_k} \tag{4.6}$$

where the $\boldsymbol{J_R}^{\dagger}(\vec{\boldsymbol{q}}_{\boldsymbol{ab}_k})$ matrix is the *relative Jacobian matrix*, which possesses a dimension equals to $(n_{ab} \times r_{ab})$ and presents the following structure:

$$\boldsymbol{J_R}(\vec{\boldsymbol{q}}_{\boldsymbol{ab}_k}) = \begin{bmatrix} -\boldsymbol{\psi}_{\boldsymbol{B_e}}^{\boldsymbol{A_e}} \boldsymbol{\Omega}_{\boldsymbol{A_b}}^{\boldsymbol{A_e}} \boldsymbol{J_A}(\vec{\boldsymbol{q}}_{\boldsymbol{a}_k}) & \boldsymbol{\Omega}_{\boldsymbol{B_b}}^{\boldsymbol{A_e}} \boldsymbol{J_B}(\vec{\boldsymbol{q}}_{\boldsymbol{b}_k}) \end{bmatrix} \tag{4.7}$$

where $\boldsymbol{\psi}_{\boldsymbol{B_e}}^{\boldsymbol{A_e}}$ is the wrench transformation matrix that takes into account the skew-symmetric matrix $\boldsymbol{S}(\vec{\boldsymbol{P}}_{\boldsymbol{R}})$, while the $\boldsymbol{\Omega}_{\boldsymbol{A_b}}^{\boldsymbol{A_e}}$ and $\boldsymbol{\Omega}_{\boldsymbol{B_b}}^{\boldsymbol{A_e}}$ are diagonal matrices, which include the rotation matrices $\boldsymbol{R}_{\boldsymbol{A_b}}^{\boldsymbol{A_e}}$ and $\boldsymbol{R}_{\boldsymbol{B_b}}^{\boldsymbol{A_e}}$ evaluated at time k:

$$\boldsymbol{\psi}_{\boldsymbol{B_e}}^{\boldsymbol{A_e}} = \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{S}(\vec{\boldsymbol{P}}_{\boldsymbol{R}}) \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \quad \boldsymbol{\Omega}_{j}^{i} = \begin{bmatrix} \boldsymbol{R}_{j}^{i} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_{j}^{i} \end{bmatrix}. \tag{4.8}$$

The novelty of this innovative relative Jacobian formulations concerns the presence of $\boldsymbol{\psi}_{\boldsymbol{B_e}}^{\boldsymbol{A_e}}$, which did not appear in the previous expressions of the relative Jacobian. This term is normally present in parallel mechanisms, (e.g., the dual-arm system). In particular, high values of $\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{A}}$ increase the relative end effectors position error, as demonstrated in [26]. When $\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{A}}$ is low, its contribution to the relative translational velocity is negligible, and the wrench transformation matrix is approximated with the identity matrix, as commonly assumed in the literature [59, 60].

The relative Jacobian matrix $\boldsymbol{J_R}$ expressed in (4.7) presents several advantages. First of all, a dual-arm system can be controlled by the same algorithms used for controlling single manipulators. Moreover it is simple to obtain from the Jacobians of the individual manipulators, so that, in case a manipulator is replaced with another one, it is sufficient to change the respective Jacobian without recalculating the entire relative Jacobian matrix. Finally this formulation shows that the relative Jacobian is not rank deficient even if one of the individual Jacobian loses ranks. Therefore, the dual-arm robot can work even if one robotic arm becomes singular.

4.2 Inverse kinematics solution for cooperative manipulators

Since the equivalent manipulator possesses a higher number of jointsn_{ab} than the end-effector variables r_{ab}, $(n_{ab} > r_{ab})$, it results to be kinematic redundant with respect to the relative motion task. Therefore, the relative Jacobian is a low rectangular matrix, and in accordance with the CLIK algorithm based on (3.12), the inverse differential kinematics is calculated by the Jacobian pseudo-inverse method, as shown in (4.9).

$$\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{ab}_k} = \boldsymbol{J_R}^{\dagger}(\vec{\boldsymbol{q}}_{\boldsymbol{ab}_k})(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k} + \boldsymbol{K}\vec{\boldsymbol{e}}_{\boldsymbol{R}_k}) \tag{4.9}$$

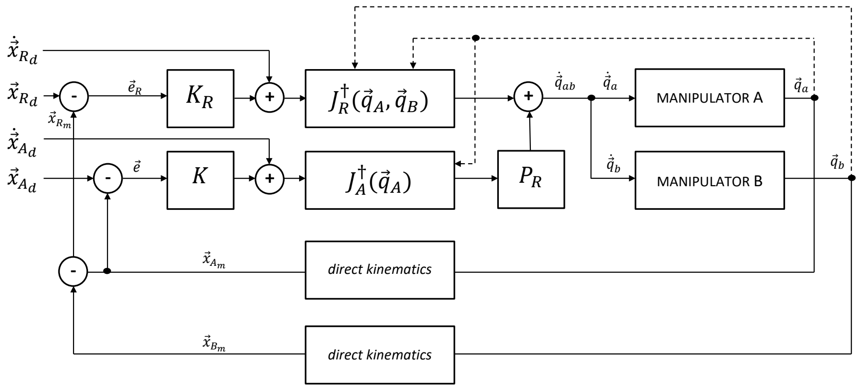


Figure 4.2: Block diagram of the relative Jacobian pseudo-inverse algorithm

where $\boldsymbol{J_A}^{\dagger}$ has been replaced with $\boldsymbol{J_R}^{\dagger}$, and $\boldsymbol{e_{R_k}}$ indicates the error between the desired relative end effectors position $\boldsymbol{x_{R_{d_k}}}$ and the relative end effectors position $\vec{\boldsymbol{x}}_{\boldsymbol{R}_k}$

$$\vec{\boldsymbol{e}}_{\boldsymbol{R}_k} = \vec{\boldsymbol{x}}_{\boldsymbol{R_{d_k}}} - \vec{\boldsymbol{x}}_{\boldsymbol{R}_k} \tag{4.10}$$

Since, the relative Jacobian possesses a null space with dimension equals to $n_{ab} - r_{ab}$, it is possible to projects a secondary task in this space, in accordance with (3.15):

$$\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{ab}_k} = \boldsymbol{J_R}^{\dagger}(\vec{\boldsymbol{q}}_{\boldsymbol{ab}_k})(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{R_{d_k}}} + \boldsymbol{K_R}\vec{\boldsymbol{e}}_{\boldsymbol{R}_k}) + \boldsymbol{P}_{\boldsymbol{R}_k}\dot{\vec{\boldsymbol{q}}}_0^{+} \tag{4.11}$$

where $\boldsymbol{P}_{\boldsymbol{R}_k} = \boldsymbol{I} - \boldsymbol{J_R}^{\dagger}\boldsymbol{J_R}$ is the $(n_{ab} \times n_{ab})$ -dimensional orthogonal projector matrix into the relative Jacobian null space, while $\boldsymbol{K_R}$ is the gain matrix that converges to zero the relative end effectors pose error $\vec{\boldsymbol{e}}_{\boldsymbol{R}}$.

Since, the (4.9) does not consider the end-effector motion of each manipulator in the operative space (i.e., $m \leq 6$ in a three dimensional space) but only their relative motions, it is possible to consider the translational and rotational motion of each end-effector in the space as a desired secondary task. In particular, it is sufficient to consider only the motion of the end-effector master A, $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}}$, because the motion of both end-effectors is constrained by $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{R_{d_k}}}$.

Therefore it is possible to rewrite the (4.11) with $\boldsymbol{J_A} = [\boldsymbol{J_A}(\vec{\boldsymbol{q}}_{\boldsymbol{a}_k}), \boldsymbol{J_A}(\vec{\boldsymbol{q}}_{\boldsymbol{b}_k})]$:

$$\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{ab}_k} = \boldsymbol{J_R}^{\dagger}(\vec{\boldsymbol{q}}_{\boldsymbol{ab}_k})(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{R_{d_k}}} + \boldsymbol{K_R}\vec{\boldsymbol{e}}_{\boldsymbol{R}_k}) + \boldsymbol{P}_{\boldsymbol{R}_k}(\boldsymbol{J_A}^{\dagger}(\vec{\boldsymbol{q}}_{\boldsymbol{a}_k}, \vec{\boldsymbol{q}}_{\boldsymbol{b}_k})\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k}) \tag{4.12}$$

Since the end-effector motion A is the secondary task, in accordance with Jacobian null space theory, the relative end-effector motion performance (primary task) is not affected from it. Finally, the block diagram representation about the CLIK algorithm based on (4.12) is shown in Figure (4.2).

Since the equivalent manipulator obtained by the relative Jacobian method can be considered as a generic redundant manipulator, its redundancy can be exploited to perform multiple tasks in accordance with the hierarchical priority execution task, as described in the previous chapter.

Therefore it is possible to reconsider the condition expressed in (3.25) and adapt it according to the n_{ab} joints possessed by the equivalent manipulator:

$$\sum_{i=1}^{l} rk_i = n_{ab} \tag{4.13}$$

As previously introduced, a task for a dual-arm is composed generally by two kinds of motions: relative end-effectors motion and absolute system motion (motion of whole system in the space). Therefore, possible objective functions can be satisfied only when there are yet available redundant motions for executing a third task. In order to calculate this number, the (4.13) can be rewritten as the following inequality:

$$n_{ab} - (rk_{ab} + rk_a) > 0 \tag{4.14}$$

where s_{ab} and s_a indicates the rank of the Jacobian associated to first task (relative end-effector motion) and second task (absolute system motion). Therefore, if the inequality is valid, it is possible to add other lower priority tasks having maximum rank value equal to $s^* = n_{ab} - (s_{ab} + s)$, in order to obtain a hierarchical structure of the tasks.

Generally, given several secondary tasks with different priority levels, it is possible to obtain the $\dot{\vec{q}}^{*}_{\boldsymbol{ab}_k}$ vector by projecting joints velocity vector $\dot{\vec{q}}^{+}_{i}$ of the i-th task into the null space relative to the higher priority task.

Defining $\dot{\vec{q}}_{\boldsymbol{2}}^{+} = \boldsymbol{J_A}^{\dagger}(\vec{q}_{\boldsymbol{a}_k}, \vec{q}_{\boldsymbol{b}_k})\dot{\vec{x}}_{\boldsymbol{d}_k}$ and a generic joints velocity task $\dot{\vec{q}}_{\boldsymbol{1}}^{+}$ (having dimensions equal to n_{ab}), the general expression for the execution of three tasks can be obtained in accordance with [61]:

$$\dot{\vec{q}}^{*}_{\boldsymbol{ab}_k} = \boldsymbol{J_R}^{\dagger}(\vec{q}_{\boldsymbol{ab}_k})\dot{\vec{x}}_{\boldsymbol{R}_{\boldsymbol{d_k}}} + \boldsymbol{P}_{\boldsymbol{R}_k}(\dot{\vec{q}}_{\boldsymbol{2}}^{+} + \boldsymbol{P}_{\boldsymbol{ab}_k}\dot{\vec{q}}_{\boldsymbol{1}}^{+}) \tag{4.15}$$

where $\boldsymbol{P}_{\boldsymbol{ab}_k}$ is a $n_{ab} \times n_{ab}$ dimensional matrix that contains the orthogonal projectors into the null spaces of the manipulators A and B.

$$\boldsymbol{P}_{\boldsymbol{ab}_k} = \begin{bmatrix} \boldsymbol{P}_{\boldsymbol{a}_k} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{P}_{\boldsymbol{b}_k} \end{bmatrix} \tag{4.16}$$

Several examples of hierarchical priority task applications will be discussed in the next chapter.

Chapter 5

Proposed kinematic control algorithms for obstacle and joint limits avoidance

This chapter presents the redundancy resolutions based on the relative Jacobians, which have been studied and implemented for controlling different dual-arm systems in several scenarios. The main contributions concerns the management of the redundant motions and the use of hierarchical priority task architecture in order to satisfy both primary task and the desired performance criteria. In detail the proposed performance criteria are: obstacle avoidance, joint limits avoidance and fault joint tolerance.

5.1 Control algorithm for obstacle avoidance

5.1.1 Case study

As illustrative example this section presents a case study consisting of two Jaco manipulators [62] mounted on both sides of a smart wheelchair, whose degrees of redundancy are employed to move an object along a predefined path, while avoiding an obstacle in the manipulator's workspace at the same time.

5.1.2 Task Description

The study case consists in the development of a kinematic control based on the relative Jacobian matrix of two cooperative Jaco manipulators, namely A and B, mounted on the left and right sides of a smart wheelchair, as shown in Figure 5.1. In this scenario, the user can reach autonomously any indoor point by using the smart wheelchair, thanks to a Simultaneous Localization And Map building algorithm (SLAM) [63, 64, 65]. The two cooperative manipulators add manipulation capability to the wheelchair in order to transport objects along a predefined path. Since the aim of this section is focused on the analysis of the

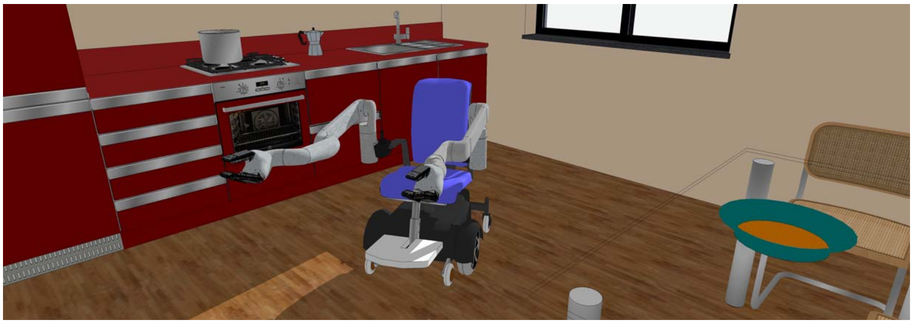


Figure 5.1: Concept of the wheelchair with two manipulators within an AAL scenario

kinematic performance of a dual-manipulation system, the wheelchair motion is not considered during the execution of the manipulation tasks. Moreover the case study is faced in the planar case. The reader can refer to the concluding section for an explanation of possible future works which can take into account for a cooperation between the wheelchair and the manipulators, and for the possible extension to the three-dimensional case.

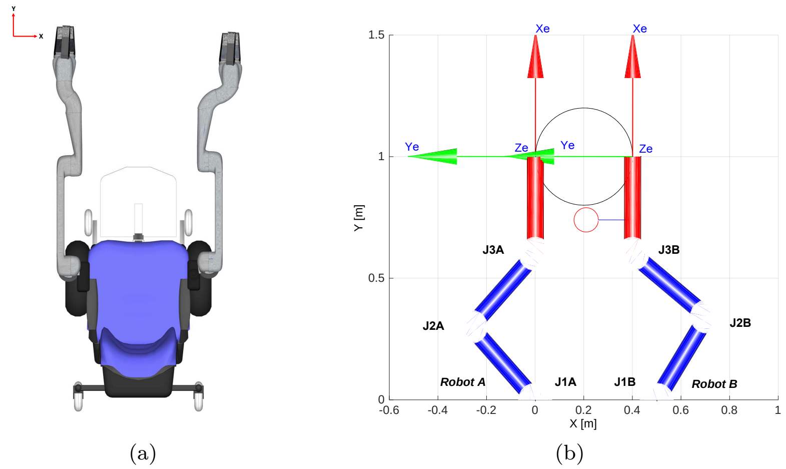


Figure 5.2: Top view of the system (a) and initial planar manipulators configuration (b)

The considered task is common in a daily scenario: during a meal the wheelchair user commands the two manipulators to move his/her dish from point $\vec{\boldsymbol{p}}_1$ to point $\vec{\boldsymbol{p}}_2$ of the table. Moreover, the manipulators must avoid a near object (e.g., a bottle placed between the start and end positions).

The task is modelled in the XY-plane:

- the motion of the dish is along a straight line in the XY-plane, which is a common path to choose in order to reduce both the time of the task execution and the motor energy consumption;
- the dish is represented by a circle of radius r_1;
- the bottle is represented by a circle of radius r_2;
- the two anthropomorphic manipulators (Figure 5.2(a)) are modelled as two planar manipulators having three joints each ($n_a = n_b = 3$), which grasp the dish so that the Cartesian distance between the two end effectors is equal to the diameter of the dish (Figure 5.2(b));
- the end-effector A has a low orientation velocity, thus the wrench transformation is approximated by the identity matrix.

According to the hierarchical priority task architecture, the proposed scenario could be decomposed in two tasks of different tasks: the higher priority task (primary task) and the lower priority task (secondary task). In particular, the primary task ensures the grasping of the dish throughout the motion time, i.e., the distance $\|\vec{\boldsymbol{p}}_{\boldsymbol{R}}\|$ has to be maintained constant and, consequently, its derivative with respect to time has to be equal to zero, $\dot{\vec{\boldsymbol{p}}}_{\boldsymbol{R}} = [0,0]^T$, so as the desired relative end effectors orientation velocity $\dot{\vec{\boldsymbol{\phi}}}_{\boldsymbol{R}} = 0$. Therefore, the relative end effectors velocity vector $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k}$ can be defined as:

$$\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{5.1}$$

where $dim[\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k}] = s_{ab} = 3$. Hence, it is possible to achieve an equivalent manipulator which possesses $n_{ab} = n_a + n_b = 6$ joints and $s_{ab} = 3$ end-effector motion variables, thus the degree of redundancy is $dr_{ab} = n_{ab} - s_{ab} = 3$.

The degree of redundancy is used to execute the secondary task, which defines the translation of the dish between two points. It is obtained by projecting a desired Cartesian velocity $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k}$ relative to end-effector A (or equivalently to end-effector B) in the relative Jacobian null space. In fact, since the relative motion between end effectors is established by $\dot{\vec{\boldsymbol{p}}}_{\boldsymbol{R}} = [0,0]^T$, the motion of only one end-effector implies the motion of the whole dual arms system. Since the rotation of both end effectors is not important for the correct execution of the this task, it is left unspecified, in order to have $dim[\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k}] = s_a = 2$. If the end-effector A is chosen, then the analytic Jacobian associated to it, is a lower rectangular matrix $\boldsymbol{J_A}$ ($dim[\boldsymbol{J_A}] = 2 \times 3$, with $rank[\boldsymbol{J_A}] = s_a = 2$).

Therefore, it is possible to obtain the 6-dimensional joints velocity vector $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{2}}^{+}$ relative to secondary task by using of CLICK pseudo-inverse expression as described in (4.9):

$$\dot{\vec{\boldsymbol{q}}}^{+}_{\boldsymbol{2}_k} = \boldsymbol{J_A}^{\dagger}(\vec{\boldsymbol{q}}_{\boldsymbol{a}_k}, \vec{\boldsymbol{q}}_{\boldsymbol{b}_k})(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k} + \boldsymbol{K}\vec{\boldsymbol{e}}_k) \tag{5.2}$$

where $\boldsymbol{J_A}^{\dagger}(\vec{\boldsymbol{q}}_{\boldsymbol{a}_k}, \vec{\boldsymbol{q}}_{\boldsymbol{b}_k}) = \begin{bmatrix} \boldsymbol{J_A}(\vec{\boldsymbol{q}}_{\boldsymbol{a}_k}) & \boldsymbol{0} \end{bmatrix}^{\dagger}$, due to the fact that the manipulator A has been chosen.

The final degree of redundancy of the dual arms system dr_{ab} results

$$dr_{ab} = n_{ab} - s_{ab} - s_a = 1 \tag{5.3}$$

From (5.3) it results that the dual arm system has still one degree of redundancy, which can be used to execute a third secondary task, such as obstacle avoidance. In particular, the dual arms system is able to avoid an obstacle of unknown position, which could be detected, for instance, by using a vision system.

The obstacle avoidance task is obtained by using a projecting gradient method [42]. This method permits to increase the distance between the obstacle and the nearest manipulator by projecting the joint space velocity relative to the third task in the null space of the secondary task. Since $s_a = 2$, the third task does not affect the translation of the end effectors, but only their orientation that is not important for the correct execution of the higher priority task. The third task can be performed by an algorithm that operates in two steps. In the first step, the algorithm detects the arm closest to the obstacle, by calculating iteratively the minimum distance vector $\vec{\boldsymbol{w}}$ between $\vec{\boldsymbol{a}}(\vec{\boldsymbol{q}}_{\boldsymbol{a}_k})$ and $\vec{\boldsymbol{a}}(\vec{\boldsymbol{q}}_{\boldsymbol{b}_k})$ which are the position vectors of two generic points along the both structure manipulators, and $\vec{\boldsymbol{b}}$ that is the position vector of a suitable point on the obstacle

$$\vec{w}_{i_k} = min\|\vec{\boldsymbol{a}}(\vec{\boldsymbol{q}}_{\boldsymbol{i}_k}) - \vec{\boldsymbol{b}}\|^2 \tag{5.4}$$

where the subscript $i = a, b$ indicates the two manipulators A and B. Suppose now that A is the arm closest to the obstacle (the same applies if B is the closest), the second step consists in maximizing the distance by calculating the gradient $\nabla|\vec{w}_a|$ in the joints space. Therefore, it is possible to obtain the 6-dimensional joints velocity vector $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{1}}^{+}$ $(s^* = 1)$ relative to manipulator A

$$\dot{\vec{\boldsymbol{q}}}^{+}_{\boldsymbol{1}_k} = k_a[\nabla_q|\vec{w}_{a_k}|\ \boldsymbol{0}]^T \tag{5.5}$$

where the obstacle avoidance gain k_a is defined as

$$\begin{cases} k_a = (1 - (|\vec{w}_a|/d_T))k_a^* & |\vec{w}_a| < d_T \\ k_a = 0 & |\vec{w}_a| \geq d_T \end{cases} \tag{5.6}$$

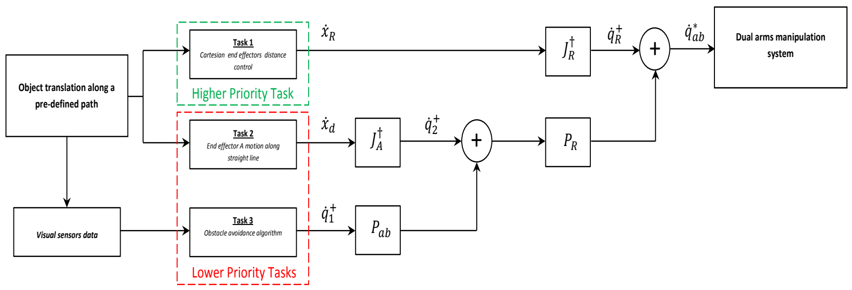


Figure 5.3: Block diagram of two cooperative manipulators

with k_a^* being the nominal gain value and d_T is the threshold distance where the the obstacle avoidance task is active. Finally, it is possible to calculate the final joints velocity vector of the equivalent manipulator by inserting (5.2) and (5.5) into (4.15)

$$\begin{aligned}\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{ab}_k} = \boldsymbol{J_R}^\dagger(\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}_k} + \boldsymbol{K_R}\vec{\boldsymbol{e}}_{\boldsymbol{R}_k}) + \boldsymbol{P}_{\boldsymbol{R}_k}(\begin{bmatrix}\boldsymbol{J_A}(\vec{\boldsymbol{q}}_{\boldsymbol{a}_k}) & \boldsymbol{0}\end{bmatrix}^\dagger \\ (\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{d}_k} + \boldsymbol{K}\vec{\boldsymbol{e}}_k) + \boldsymbol{P}_{\boldsymbol{ab}_k} k_a[\nabla_q|\vec{w}_{a_k}|\ \boldsymbol{0}]^T)\end{aligned} \tag{5.7}$$

Equation (5.7) presents a hierarchical task structure, which is summarized in the block digram shown in Figure 5.3. Thus, the final degree of redundancy of the system is

$$dr_{ab} = n_{ab} - s_{ab} - s_a - s^* = 0 \tag{5.8}$$

5.1.3 Simulation results

In this section we simulated the movement of a dish of radius $r_1 = 0.2m$ and centre coordinates $p_c = [0.2m, 1m]$ along a straight line between the points $\vec{\boldsymbol{p}}_{\boldsymbol{1}} = [0m, 1m]^T$ and $\vec{\boldsymbol{p}}_{\boldsymbol{2}} = [0.25m, 1m]^T$. Moreover, a bottle centred in $\vec{\boldsymbol{c}} = [0.21m, 0.74m]^T$ and of radius $r_2 = 0.05m$ is placed in the dual manipulation workspace [66, 67], so that an obstacle avoidance algorithm is implemented and assigned to the manipulator nearest to the obstacle. This simulation is composed by three different priority level tasks

1. maintaining a relative distance between the two end effectors $\|\vec{\boldsymbol{p}}_{\boldsymbol{R}}\| = 0.4m$;

2. moving the manipulator A along the defined path;

3. avoiding an obstacle (i.e., a bottle).

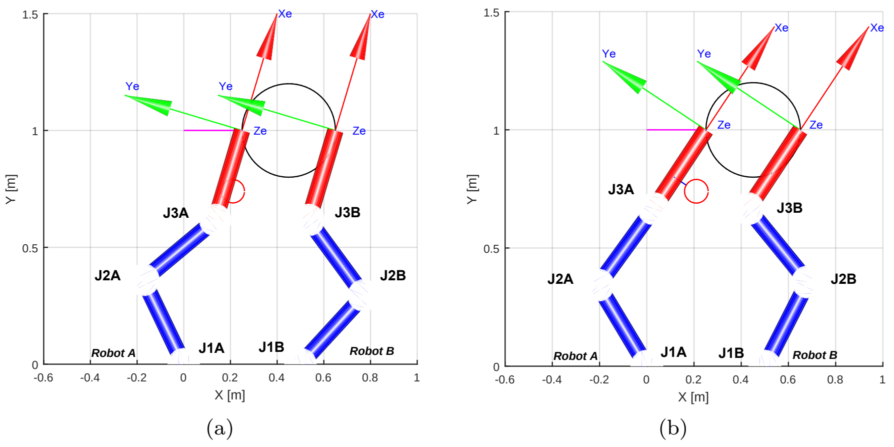


Figure 5.4: Final manipulators configuration without (a) and with (b) obstacle avoidance task.

In order to keep a constant end effectors distance, the relative Cartesian velocity vector $\dot{\vec{x}}_{Rd}$ is imposed in accordance with [53], while the desired Cartesian velocity is assigned to be $\dot{\vec{x}}_d = [0.05m, 0m]^T$. Finally, the obstacle avoiding velocity is obtained in the joints velocity space by $k_a \nabla_q |\vec{w}|$, when the distance between obstacle and manipulators is less than a threshold distance d_T which is imposed to $0.20m$.

The start pose of the dual-arm system is shown in Figure 5.2(b), where the starting A manipulator pose is $\vec{x}_{ea} = [0m, 1m, \pi/2rad]^T$, the starting B manipulator pose is $\vec{x}_{eb} = [0.4m, 1m, \pi/2rad]^T$, while the distance between the manipulator bases d_{AB} is equal to $0.50m$ in accordance with the width of the wheelchair. The black circle and the red circle represent the section of the dish and bottle in the XY-plane, respectively; while the blue line represents

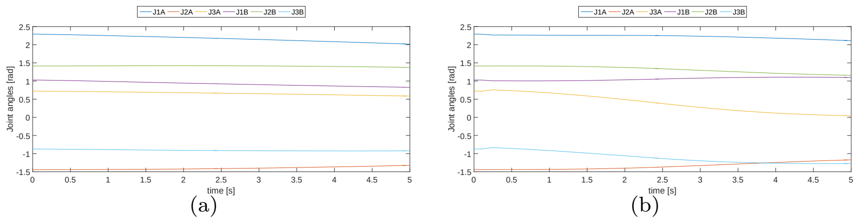


Figure 5.5: Joint angle position without (a) and with (b) obstacle avoidance task

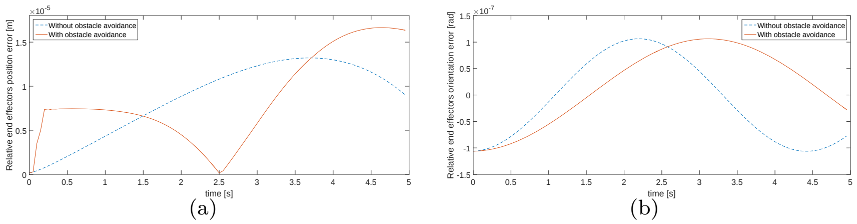


Figure 5.6: Relative end-effectors position error (a) and orientation error (b)

the minimal distance between the obstacle point $\vec{\boldsymbol{b}}$ and the nearest manipulator point $\vec{\boldsymbol{a}}(\vec{\boldsymbol{q}}_{\boldsymbol{i}_k})$ at time k. Figure 5.4(a) and (b) show the final pose of the dual-arm system, without and with the obstacle avoidance task, while the purple line indicates the path tracked by manipulator A. It is worth noting that, since the end effectors orientations are not specified by the task, they can arbitrary change during the motion when the obstacle avoidance task is not applied. However, the relative end effectors orientation value is kept constant by the primary task.

The variations of joints angles without and with obstacle avoidance task are shown in Figure 5.5(a) and (b), respectively.

The performances of the first task are shown in Figure 5.6(a) and (b), where the relative end effectors pose error and the relative end effectors orientation error are reported. The obtained results indicate that the primary task is independent from the other secondary tasks.

The performances of the second task are shown in Figure 5.7(a) and (b), which indicate respectively the end-effector A position along the reference path and the relative error. Because no specification on the end effectors orientation is assigned, it can be calculated in order to maximize the distance between obstacle and manipulators in accordance with the obstacle avoidance task.

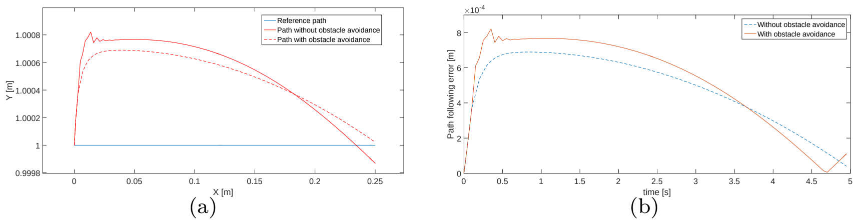


Figure 5.7: End-effector A position along the path (a) and the path following error (b)

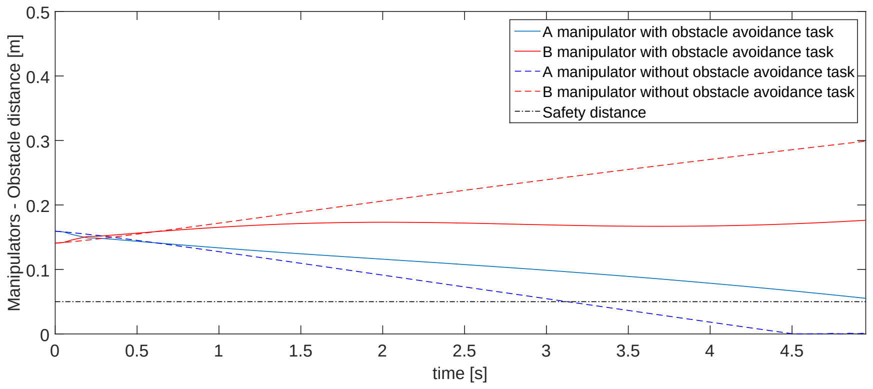


Figure 5.8: Minimum distance between manipulators and obstacle

In particular, Figure 5.8 shows the minimum distance between the two manipulators and obstacle, so that it is possible to note how the obstacle avoidance algorithm permits to keep the minimum distance value higher than safety distance. Since the primary task imposes the same relative end-effectors orientation value during the motion, the B manipulator remains near the obstacle when the obstacle avoidance task is executed.

Discussion

The obtained results show that the lower priority tasks do not affect the performance of the higher priority task. Note that, due to the presence of the Jacobian pseudo-inverse, the Jacobian null space technique used in this paragraph may present a relevant computing effort, and thus a less computationally demanding control scheme, should be considered for implementation on a real controller of two anthropomorphic manipulators (most suitable in the AAL scenarios).

5.2 Control algorithm for joint limits avoidance

This section proposes a kinematic controller for dual-arm cooperative manipulation that ensures safety by providing coordinated motion as highest priority task and joint limit avoidance and trajectory following at a lower priority. The coordination of motions is based on a relative Jacobian formulation. The approach is applicable to systems composed of redundant or not-redundant manipulators. Experiments in simulation demonstrate the behaviour of the approach under different redundancy configurations. Experiments on physical hardware with mixed redundancy demonstrate the applicability of the approach to cooperative manipulation under joint limit constraints. Finally, a supervisor controller is proposed, which is able to guarantee all joint position limits of two cooperative redundant manipulators, even when the only redundant motions are no longer sufficient to make ensure them.

5.2.1 Proposed joint position limits avoidance strategy

A classical approach to avoid joint limits is to define the gradient of a cost function as the lowest priority task [55]. This approach guides each joint towards the middle of its range, regardless of the joint position's closeness to the limit. In order to optimize the number of redundant motions that are employed in the joint limit avoidance task, a repulsive joint velocity is used in order to move only the critical joints away from their limits [68]. In particular, a joint is defined as a critical joint when its distance α from its nearest limit $q_L = [q_{L_{min}}, q_{L_{max}}]$ is less than distance β between q_L and the activation threshold $q_T = [q_{T_{min}}, q_{T_{max}}]$ of the joint limit avoidance task. For the dual arm system composed of two manipulators, it is possible to obtain a repulsive joint velocity vector for each manipulator, $\dot{\vec{q}}_{\boldsymbol{A}}^{+}$ and $\dot{\vec{q}}_{\boldsymbol{B}}^{+}$, respectively as

$$\dot{\vec{q}}_{\boldsymbol{A}}^{+} = \boldsymbol{H_A W_A}(\vec{q}_{\boldsymbol{TA}} - \vec{q}_{\boldsymbol{A}}) \tag{5.9}$$

$$\dot{\vec{q}}_{\boldsymbol{B}}^{+} = \boldsymbol{H_B W_B}(\vec{q}_{\boldsymbol{TB}} - \vec{q}_{\boldsymbol{B}}) \tag{5.10}$$

where $\vec{q}_{\boldsymbol{TA}}$ and $\vec{q}_{\boldsymbol{TB}}$ are two n_a and n_b dimensional column vectors, which contain the joint threshold positions closest to the current joint position. $\boldsymbol{H_A}$ and $\boldsymbol{H_B}$ are $(n_A \times n_A)$ and $(n_B \times n_B)$ dimensional diagonal matrices representing the gains of the control law of this task. However, these gains are weighted by the two smooth activation diagonal matrices $\boldsymbol{W_A}$ and $\boldsymbol{W_B}$, whose components w_i depend on α according to

$$w_i(\alpha) = \frac{1}{2}\left[1 - \tanh\left(\frac{1}{1-\frac{\alpha}{\beta}} - \frac{\beta}{\alpha}\right)\right], \forall\alpha \in [0, \beta] \tag{5.11}$$

giving $w_i \in [0, 1]$ as shown in Figure 5.9. The smooth transition allows to reduce the discontinuities in the joint velocity signals compared to a binary activation matrix [45].

Finally, setting $\dot{\vec{q}}_{\boldsymbol{k}}^{+} = [\dot{\vec{q}}_{\boldsymbol{A}}^{+}, \dot{\vec{q}}_{\boldsymbol{B}}^{+}]^T$ in (4.15), it is possible to obtain the following final compact matrix equation

$$\begin{aligned}\dot{\vec{q}}_{\boldsymbol{ab}} =& \boldsymbol{J}_R^{\dagger}(\dot{\vec{x}}_{\boldsymbol{Rd}} + \boldsymbol{K_R}\vec{e}_{\boldsymbol{R}}) + \boldsymbol{P_R}\Big([\boldsymbol{J_A}\ \boldsymbol{0}]^{\dagger}(\dot{\vec{x}}_{\boldsymbol{Ad}} + \boldsymbol{Ke}) + \\ & \boldsymbol{P_{AB}H_{AB}W_{AB}}(\vec{q}_{\boldsymbol{T_{AB}}} - \vec{q}_{\boldsymbol{AB}})\Big)\end{aligned} \tag{5.12}$$

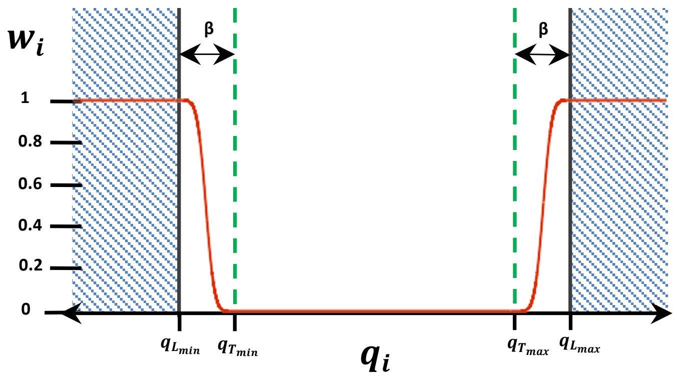


Figure 5.9: Activation function for one component w_i of W matrix.

where $\vec{\boldsymbol{q}}_{\boldsymbol{T_{AB}}} = [\vec{\boldsymbol{q}}_{\boldsymbol{T_A}}{}^+, \vec{\boldsymbol{q}}_{\boldsymbol{T_B}}{}^+]^T$, while $\boldsymbol{H_{AB}}$ and $\boldsymbol{W_{AB}}$ are defined as

$$\boldsymbol{H_{AB}} = \begin{bmatrix} \boldsymbol{H_A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H_B} \end{bmatrix} \qquad \boldsymbol{W_{AB}} = \begin{bmatrix} \boldsymbol{W_A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W_B} \end{bmatrix}. \tag{5.13}$$

Note that the joint limit avoidance task has lower priority than trajectory following. This can appear counter-intuitive. However, if the system is redundant, the lower priority of the task avoids losing trajectory tracking performance while still using the redundancy to avoid joint limits. In the case of a non-redundant system, the null-space projector $\mathbf{P_{AB}}$ needs to be modified so that the trajectory tracking and joint limit avoidance have the same priority.

Joint limit avoidance is not ensured when the manipulators possess more critical joints than redundant motions [68]. To ensure that the joint limits are satisfied, we let the trajectory performance related to $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Ad}}$ degrade gradually and temporarily by having same priority for trajectory tracking and joint limit avoidance. To keep this specific to the non-redundant manipulator, (5.12) is adapted by replacing the null-space projection matrix of the non-redundant manipulator with an identity matrix I having the same dimension. In other words, $\boldsymbol{P_{AB}}$ in (5.12) can be replaced with one of the orthogonal projectors

$$\boldsymbol{P_{IB}} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{P_B} \end{bmatrix} \qquad \boldsymbol{P_{IA}} = \begin{bmatrix} \boldsymbol{P_A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \tag{5.14}$$

where $\boldsymbol{P_{IX}}$ is the projector matrix when the manipulator X is non-redundant. Finally, if both manipulators are non-redundant, $\boldsymbol{P_{AB}}$ matrix is replaced by an identity matrix $\boldsymbol{I_{AB}}$.

Figure 5.10 shows an example of the joint limit avoidance. Given a generic critical joint q_1, it will converge to an equilibrium where two opposite velocity

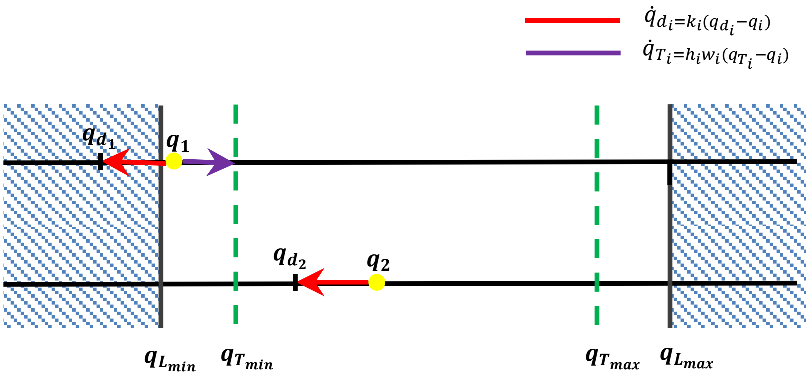


Figure 5.10: Operating principle of the joint limits avoidance strategy.

components are balanced: the first velocity (red arrow) depends on the gain matrix $\boldsymbol{K}$ and pushes q_1 towards its desired position q_{d_1}, while the second one (violet arrow), which depends on the product of h_1 and w_1, pushes q_1 towards the minimum threshold position $q_{T_{min}}$. On the other hand, a non-critical joint q_2 is not affected by the second velocity signal ($w_2 = 0$), therefore it will converge to its desired position q_{d_2}.

5.2.2 First case study

The following simulations show how to use the proposed methodology to ensure that joint limits are preserved (i.e., there is no violation of joint position constraints) in a dual-arm system performing a desired task. In detail, two identical planar manipulators with 3-DOF are involved in three different cases:

Case I both manipulators are non-redundant;

Case II both manipulators have one redundant motion;

Case III manipulator A has no redundant motions, while manipulator B has one redundant motion.

The manipulators considered in the simulation study are functionally redundant, as described in chapter II. Therefore, the number of redundant motions (or degree of redundancy) dr_A and dr_B can be calculated by applying the rank-nullity theorem [55] as $n - s = dr$.

$$n_A - s_A = dr_A \qquad\qquad n_B - s_B = dr_B \tag{5.15}$$

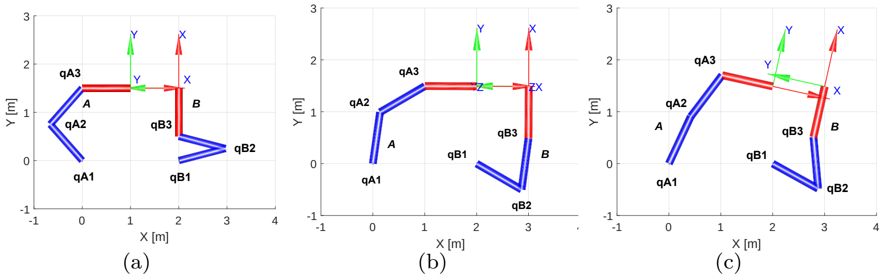


Figure 5.11: Manipulator configurations: (a) initial, (b) Case I and III final configuration, (c) Case II final configuration.)

Since both manipulators have the same number of joints ($n_A = n_B = 3$), the degree of redundancy of each manipulator can be changed by assigning different number of task variables to each end effector (functional redundancy definition, see 2.3). Hence, the tasks proposed in **Case I** are defined by three variables $s_A = s_B = 3$, so that the degree of redundancy is equal to zero for both manipulator $dr_A = dr_B = 0$. **Case II** presents two tasks described by only two motion variables $s_A = s_B = 2$ (translation motions along x and y axes), so that the degree of redundancy is equal to one, $dr_A = dr_B = 1$. Finally, **Case III** presents the cooperation of manipulators having different degree of redundancy. The task of **Case I** is assigned to manipulator A, while the task of **Case II** is assigned to manipulator B. Therefore, the degrees of redundancy are $dr_A = 0$ and $dr_B = 1$.

The proposed task consists of translating the dual arm system from position $P_1 = [1, 1.5]$ m to position $P_2 = [2, 1.5]$ m, while keeping the relative pose constant. Moreover, two joint limits q_{A2} and q_{B1} are enforced using (5.9) and (5.10) with the values reported in Table 5.1, which have been Empirically tuned.

Table 5.1: Joint limits.

Critical Joint	$q_{L_{[min,max]}}$	$q_{T_{[min,max]}}$	β	k_i	h_i
q_{A2}	$[-1.7, 3.14]$	$[-1.5, 3.04]$	0.1	10	20
q_{B1}	$[-0.7, 3.14]$	$[-0.5, 2.94]$	0.2	10	20

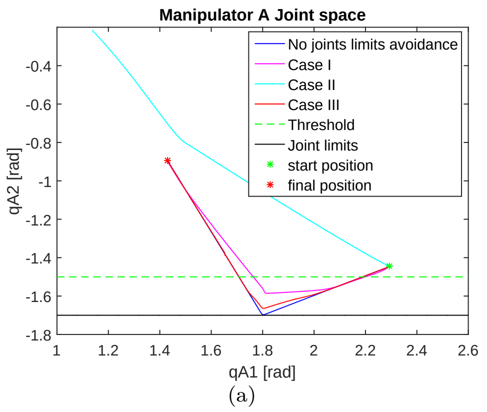
The proposed task is decomposed into prioritized sub-tasks according to (5.12). Since the execution of the joint limit avoidance depends on dr_A and dr_B, their execution priority changes for each cooperation case, as reported in Table5.2.

Table 5.2: Decomposition of the task into prioritized subtasks.

Case	Priority	Sub-task description	dimension
I	3	$\dot{\vec{x}}_{Rd} = [0\ m/s, 0\ m/s, 0\ rad/s]^T$	3
	2	$\dot{\vec{x}}_{Ad} = [0.01\ m/s, 0.01\ m/s, 0\ rad/s]^T$	3
	2	$\dot{\vec{x}}_{Bd} = [0.01\ m/s, 0.01\ m/s, 0\ rad/s]^T$	3
	2	$\dot{q}^+_{A2} = 20w_{A2}(-1.5\ rad - q_{A2})$	1
	2	$\dot{q}^+_{B1} = 20w_{B1}(-0.5\ rad - q_{B1})$	1
II	3	$\dot{\vec{x}}_{Rd} = [0\ m/s, 0\ m/s, 0\ rad/s]^T$	3
	2	$\dot{\vec{x}}_{Ad} = [0.01\ m/s, 0.01\ m/s]^T$	2
	2	$\dot{\vec{x}}_{Bd} = [0.01\ m/s, 0.01\ m/s]^T$	2
	1	$\dot{q}^+_{A2} = 20w_{A2}(-1.5\ rad - q_{A2})$	1
	1	$\dot{q}^+_{B1} = 20w_{B1}(-0.5\ rad - q_{B1})$	1
III	3	$\dot{\vec{x}}_{Rd} = [0\ m/s, 0\ m/s, 0\ rad/s]^T$	3
	2	$\dot{\vec{x}}_{Ad} = [0.01\ m/s, 0.01\ m/s, 0\ rad/s]^T$	3
	2	$\dot{\vec{x}}_{Bd} = [0.01\ m/s, 0.01\ m/s]^T$	2
	2	$\dot{q}^+_{A2} = 20w_{A2}(-1.5\ rad - q_{A2})$	1
	1	$\dot{q}^+_{B1} = 20w_{B1}(-0.5\ rad - q_{B1})$	1

Case I: in this case both manipulators have no degree of redundancy available for the execution of the joint limit avoidance, $dr_A = dr_B = 0$. Therefore, it is possible to assign to this task the priority of the higher priority task $\dot{\vec{x}}_{Ad}$, by replacing the $\boldsymbol{P_{AB}}$ in (5.12) with an identity matrix $\boldsymbol{I_{AB}}$. Figure 5.12 shows the joint position space relative to the first two joints of each manipulator. In particular, the difference between the joint trajectory without the proposed joint limit avoidance strategy (blue line) and with it (violet line). The figure shows that the joint limit avoidance works correctly, pushing both joints away from their limits when q_{A2} and q_{B1} exceed their threshold limits (green dotted line). Due to the joint limit avoidance, the trajectory tracking accuracy is degraded temporarily for both end-effectors, as shown in Figure 5.13. Moreover, Figure 5.11(b) shows that the final end effector orientations assumed by both manipulators are equal to the initial ones (see Figure 5.11(a)), enforced by the relative motion task $\dot{\vec{x}}_{Rd}$.

Case II: in this case each manipulator possesses one degree of redundancy, because both end effector orientation velocities are not specified. Therefore, (5.12) is applied directly. Since a redundant manipulator admits an infinite number of solutions for the inverse kinematic problem, it is possible to note in Figure 5.12 that the joint trajectories obtained (sky blue) are completely different from the trajectories tracked without joint limit avoidance (blue). However, the repulsive joint velocities generate self-motions in each manipulator not affecting their relative pose or their translation task . Therefore, the



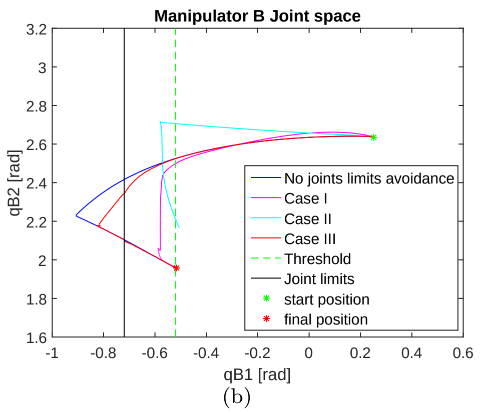


Figure 5.12: A (B) manipulator joint position space relative to q_{A1}-q_{A2} (a) (q_{B1}-q_{B2} (b)).

final configurations of the manipulators obtained (Figure 5.11(c)) are different from the starting ones (Figure 5.11(a)), and the path following error is negligible (green line in Figure 5.13).

Case III: in this last case is a combination of the previous ones. Since manipulator A is non-redundant ($\dot{\phi}_A$ is assigned), it is necessary to replace the $\boldsymbol{P_{AB}}$ in (5.12) with the matrix $\boldsymbol{P_{IB}}$ defined in (5.14). In this way, $\dot{q}_{A2}$ has the execution priority of $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Ad}}$, while $\dot{q}_{B1}$ has lower priority (see Table5.2). It is interesting to note that while the joint limit of q_{A2} is satisfied (see red line in Figure 5.12(a)), the joint limit of q_{B1} is violated (see red line in Figure 5.12(b)). In fact, although the $\dot{\phi}_B$ is not assigned, ϕ_B must be kept constant respect to ϕ_A as specified from highest priority task $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{Rd}}$. Therefore, $\dot{\phi}_B$ depends on $\dot{\phi}_A$, and consequently the B manipulator loses its degree of redundancy, so that the $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{B}}^{+}$

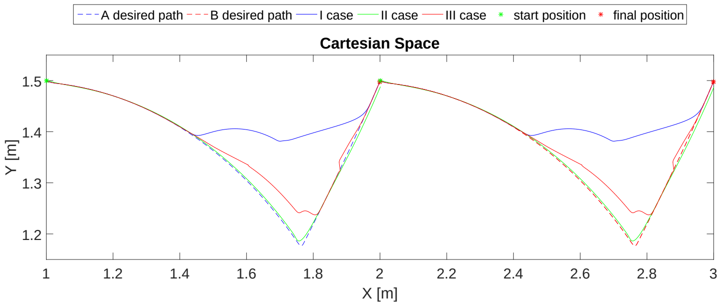


Figure 5.13: A and B manipulators Cartesian path for the three considered simulation cases.

can not be performed. In order to satisfy the joint limit of q_{B1}, it is necessary to assign to q_{B1}^{+} the same priority execution level of the higher priority task $\dot{\boldsymbol{x}}_{\boldsymbol{Bd}}$ as demonstrated in **Case I**. Therefore, the trajectory tracking performances of both manipulators are temporary degraded as described in **Case I**, and the final end effector orientations are equal to those shown in Figure 5.11(b).

Joint velocities during motion are illustrated in Figure 5.14. This shows that unlike the existing method proposed in [49], the proposed controller strategy and the smooth activation function eliminate discontinuities in joint velocity commands.

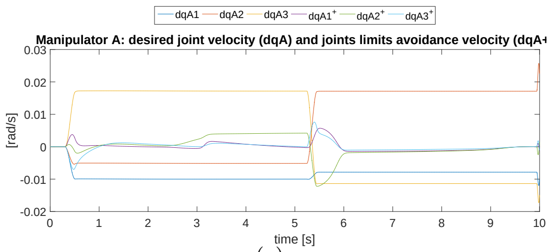
5.2.3 Second study case

The experimental setup is composed of two different manipulators holding a woodblock using their end effectors (Figure 5.15). The coordinate framed for the set-up are depicted in Figure 5.16, which are expressed in accordance with relative Jacobian formulation (see Figure 4.1).

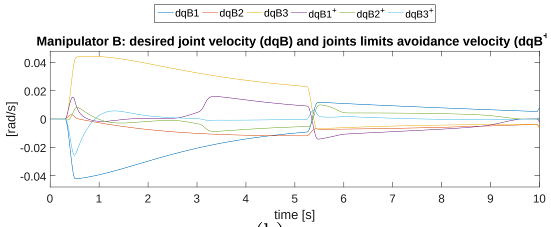
Using the set-up, these experimental test are based on four cases (Cases A–D) of cooperative manipulation, with joint position constraints shown in Table 5.3. The goal of the experiment is to analyse how the proposed controller handles the task and joint constraints in the practical setting with different sources of error and uncertainty.

Table 5.3: Experimental cases.

Case	**Kinova Jaco (non-redundant)**	**Kuka LWR (redundant)**	**Controller**
Case A	—	—	Predefined path
Case B	*Joint Limit*	—	Online path change
Case C	—	*Joint Limit*	Similar to Case A
Case D	*Joint Limit*	*Joint Limit*	Similar to Case B



(a)



(b)

Figure 5.14: Joint velocity space with repulsive velocity for A manipulator and B manipulator.

The manipulators adopted in the experimental set-up are a 6-DOF Kinova Jaco (non-redundant) [62] and a 7-DOF KUKA LWR4+ (intrinsically redundant) [69] placed opposite and parallel to each other with a distance of $1.48m$. Their task is to cooperatively manipulate a woodblock of $0.9kg$. The task priorities are similar to earlier:

1. Task 1 (highest priority): maintaining relative position and orientation of the end effectors such that $\vec{\boldsymbol{P}}_{\boldsymbol{R}} = [0, 0.5, 0]^T m$.

2. Task 2: moving the Jaco end effector A_e along the predefined path.

3. Task 3: joint limit avoidance for cooperative manipulators (see Table 5.3).

The target trajectory is a circle with radius 0.04 m, as illustrated in Figure 5.17(a). Jaco starts from initial position $A_e^o = [0.262, -0.258, 0.388]^T$ m. In this case, the Joint limits has been enforced by using the coefficient shown in Table 5.4), which has been tuned by experimental mode.

Figure 5.18 illustrates the Cartesian tracking errors and the relative pose error for Cases A-C. Case D is not shown as its behaviour is almost identical to Case B. Case A (without joint limits) serves as a reference for the achievable performance due to limitations of the hardware, in particular Kinova Jaco.

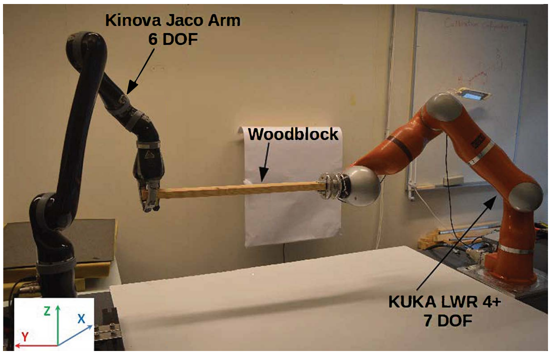


Figure 5.15: Two robot manipulators holding a woodblock to perform tightly coordinated cooperative operations.

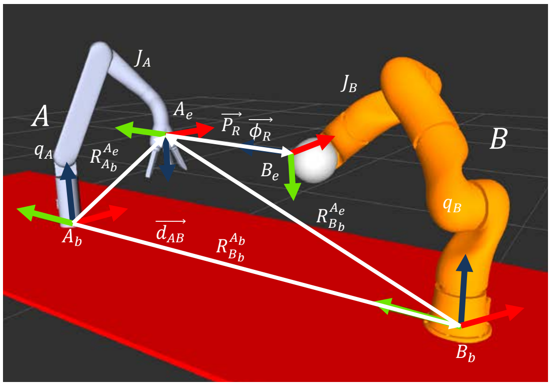


Figure 5.16: Heterogeneous two-robot system: coordinate frame transformation for the relative Jacobian formulation.

In Case B with joint limit in the non-redundant robot (Jaco), the joint limit is avoided as shown in Figure 5.19(a). During joint limit avoidance, path following accuracy is temporarily sacrificed to up to 15 mm position error (shown by the red line in Figure 5.18(a–b). The coordination of motion (Task 1) is kept enforced as the relative motion error is not increased over baseline (Figure 5.18c). The same behavior is illustrated in the Cartesian space in Figs. 5.17(a-b).

In Case C with joint limit in the redundant robot (KUKA LWR), the proposed controller is able to avoid the joint limit as illustrated in Figure 5.19b. Due to the redundancy, the accuracy of Cartesian trajectory or relative position are not deteriorated as shown in Figs. 5.17 and 5.18.

Even though KUKA can use its redundancy to avoid conflicts in joint and task space, this extra degree of freedom does not solve the problem of joint limit occurring on Jaco. Therefore the cooperative behaviour in Case D (joint

Table 5.4: Joint limit parameters.

Critical Joint	$q_{L_{[min,max]}}$	$q_{T_{[min,max]}}$	β	k_i	h_i
q_{A3}	$[0.1, 0.59]$	$[0.1, 0.54,]$	0.1	10	20
q_{B1}	$[0.1, 0.88]$	$[0.1, 0.83]$	0.2	10	20

limits for both) is similar to Case B (Jaco joint limit).

Looking more closely at the relative position errors in Figure 5.18(c), the errors remain small in all cases. In Case B the cooperative manipulators need to change Cartesian path to avoid joint limits, increasing the path following error temporarily but the limited relative motion error is maintained. This demonstrates the ability of the proposed approach to handle cooperative manipulation in a safe manner.

Finally, cooperation between intrinsically redundant and non-redundant coupled robots demonstrates the proposed controller ability to take advantage of redundancy, as well as to temporarily sacrifice the Cartesian trajectory accuracy to comply with joint limits.

Discussion

The experimental results indicate one interesting observation for heterogeneous systems: the lower performance robot sets the performance limit for the entire system. This has a significant consequence for the system design in that the lower performance robot should be used as the master, whose trajectory tracking error is used as feedback. That is, in the context of this experiment the lower performance robot should be manipulator A whose trajectory error, $\vec{\mathbf{e}}$, should be used as the trajectory feedback. This ensures that the performance of the highest priority coordinated motion task will not depend on the

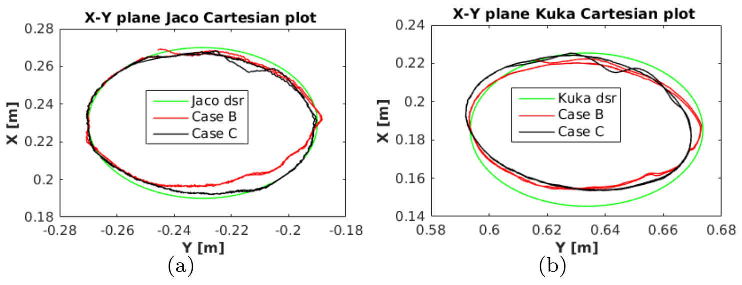


Figure 5.17: Coordinated manipulation results of tightly-coupled manipulators under joint constraints (Case B and C).

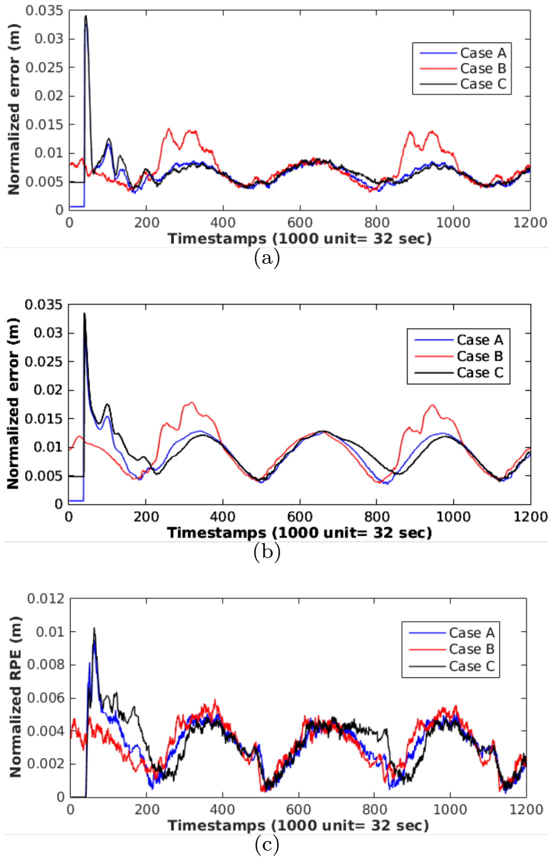


Figure 5.18: Cooperative performance of tightly-coupled manipulators for the Cases (Case A-C), refer to Table5.3.

performance of the lower performance robot.

Experimental implementation of a controller for heterogeneous robots is challenging because the access to hardware is usually not uniform. To implement the controller, the hardware plug in compatible with ROS (Robotic Operating System)-control framework has been adopted in order to abstract the two systems as a single robot. The implemented controller communicated with robot-specific native low-level controllers at 100 Hz. The low-level controller of Kinova Jaco was fixed and did not allow tuning. The performance of the low-level controllers differed significantly. To address this, the gains of the proposed controller were tuned manually. Nevertheless, the performance of the low-level controller for Jaco was limited which can be seen in Figure 5.18(a–b) as tracking errors even for the baseline Case A without joint limits. Despite this limitation, the experiments indicated no increase in tracking error unless

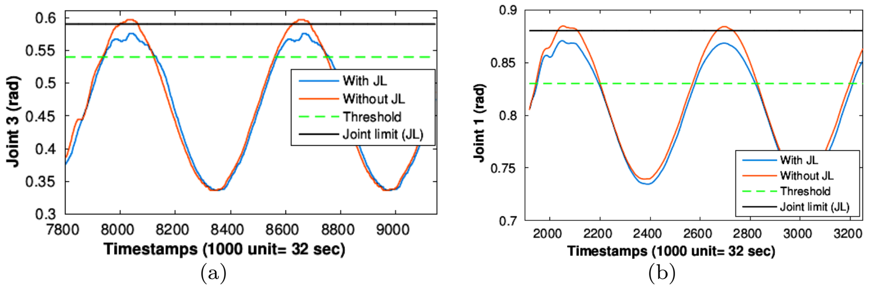


Figure 5.19: Controller treats joint limits enforced in manipulators.

forced by the joint limit avoidance.

The relative position errors were below 5 mm in all configurations. Thus the proposed approach is applicable in practice to collaborative manipulation. The remaining position errors were due to limitations in the performance of native low-level controllers. Applications requiring higher performance would likely need an integrated custom low-level controller controlling both manipulators.

5.2.4 Supervisor controller for redundancy management

As previously introduced in the hierarchical priority task paragraph (see 4.2.1) and demonstrated in the last experiments, the joint limit avoidance task has lower priority than trajectory following task. This can appear counter-intuitive. However, if the system is redundant, the lower priority of the task does not affect the trajectory tracking performance, and at the same time, it ensures the joint limits. However, when the manipulators possess more critical joints than redundant motions, joint limit avoidance is not ensured ([68]).

In order to ensure the joint limits that cannot be satisfied by the redundant motions, it is possible to assign the second execution priority level to joint limit avoidance task, which has the same priority level of the trajectory following task. The proposed solution is based on a supervisory control system, which decides if the *i-th* critical joint velocity has to be projected into the Jacobian null space associated to the specific robotic arm or if it has to be summed algebraically to the joint velocity contribution due to the trajectory tracking. The chosen criteria depends on the number of critical joints n that have been detected in a specific time instant, and in particular, when it is higher than the degree of redundancy dr (which can be obtained by calculating the rank of $\boldsymbol{P_{AB}}$ matrix), the supervisory controller assigns to the last detected critical joint the same priority execution level of the trajectory tracking (level equals to two), as shown in Figure 5.20. The disadvantage of this method concerns the lost of performance about the pre-defined trajectory tracking accuracy, because

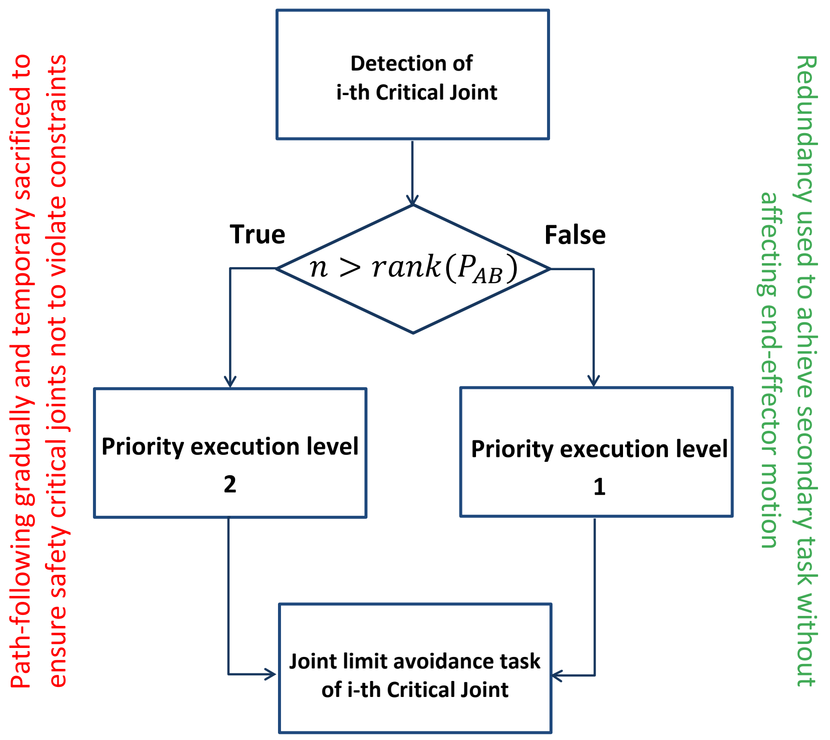

Detection of i-th Critical Joint
$n > rank(P_{AB})$
True
False
Priority execution level 2
Priority execution level 1
Joint limit avoidance task of i-th Critical Joint
Path-following gradually and temporary sacrificed to ensure safety critical joints not to violate constraints
Redundancy used to achieve secondary task without affecting end-effector motion

Figure 5.20: Operating principle of the supervisor controller.

the path following error is locally higher, but it is gradual and temporary.

In order to introduce the supervisory controller system, (5.12) can be modified as reported in the following:

$$\begin{aligned}\dot{\boldsymbol{q}}_{ab} = \boldsymbol{J}_R^{\dagger}(\dot{\boldsymbol{x}}_{R_d} + \boldsymbol{K}_R\boldsymbol{e}_R) + \boldsymbol{P}_R([\boldsymbol{J}_A \;\; \boldsymbol{0}]^{\dagger}(\dot{\boldsymbol{x}}_{A_d} + \boldsymbol{K}\boldsymbol{e}) + \\ +(\boldsymbol{C}_{AB} + \boldsymbol{B}_{AB}\boldsymbol{P}_{AB})\boldsymbol{H}_{AB}\boldsymbol{W}_{AB}(\boldsymbol{q}_{T_{AB}} - \boldsymbol{q}_{AB}))\end{aligned} \tag{5.16}$$

where the $\boldsymbol{C}_{AB}$ and $\boldsymbol{B}_{AB}$ matrices having a dimension equals to $(n_{ab} \times n_{ab})$, modify the column of $\boldsymbol{P}_{AB}$ matrix relative to the critical joints q_i^+ when the corresponding manipulator does not have available degrees of redundancy. Therefore, these matrices can be defined as

$$\boldsymbol{B}_{AB} = \begin{bmatrix} \boldsymbol{B}_A & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B}_B \end{bmatrix} \boldsymbol{C}_{AB} = \begin{bmatrix} \boldsymbol{C}_A & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_B \end{bmatrix} \tag{5.17}$$

where the matrices $\boldsymbol{B}_i$ and $\boldsymbol{C}_i$ (with $i = A, B$) are diagonal matrices, whose components can assume the following values:

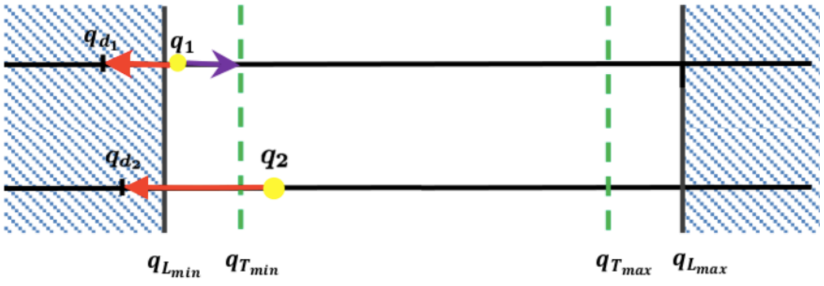


Figure 5.21: Operating principle of the joints avoidance strategy.

$$b_{A_{ii}} = \begin{cases} 1, & n_a \leq \text{rank}(\boldsymbol{P}_A) \\ 0, & n_a > \text{rank}(\boldsymbol{P}_A) \end{cases} \qquad b_{B_{ii}} = \begin{cases} 1, & n_b \leq \text{rank}(\boldsymbol{P}_B) \\ 0, & n_b > \text{rank}(\boldsymbol{P}_B) \end{cases}$$

$$c_{A_{ii}} = \begin{cases} 0, & n_a \leq \text{rank}(\boldsymbol{P}_A) \\ 1, & n_a > \text{rank}(\boldsymbol{P}_A) \end{cases} \qquad c_{B_{ii}} = \begin{cases} 0, & n_b \leq \text{rank}(\boldsymbol{P}_B) \\ 1, & n_b > \text{rank}(\boldsymbol{P}_B) \end{cases}$$

Therefore, the component values assigned by the supervisory controller depend on the n_a and n_b, which indicate the number of critical joint possessed by each manipulator in a specific time instant.

In detail, this chosen criteria can be implemented by using the example shown in Figure 5.21, which is focused on two joints: the first one is initially a critical joint q_1^+, while the second one starts as a not critical joint position q_2, but then becomes a critical joint when it overcomes the $q_{T_{min}}$ position in order to reach its desired final position q_{d_2} (which is out from the joint position interval).

Therefore, if a generic cooperative manipulator (i.e., A manipulator) has a degree of redundancy equals to one, the q_1^+ is managed by using redundancy, because $n = rank(P_A) = 1$, while the $\boldsymbol{B}_{AB}$ matrix is equal to the identity matrix and $\boldsymbol{C}_{AB}$ is a null matrix. Contrary, when the second joint becomes a critical joint, q_2^+, no degrees of redundancy are available and q_2^+ is managed by increasing the priority of the relative joint limit avoidance task. Therefore the $b_{A_{22}}$ and $c_{A_{22}}$ coefficients of $\boldsymbol{B}_A$ and $\boldsymbol{C}_A$ are switched to zero and one respectively, in order to properly modify the $\boldsymbol{P}_{AB}$ matrix.

5.2.5 Third study case

This section proposes an example of application of the joint limit strategy previously described. Since the Baxter robot possesses two homogeneous arms, the critical joints that have been considered in the study belong to the arm A

(left arm). However, the experiment can be repeated considering the joints of the arm B.

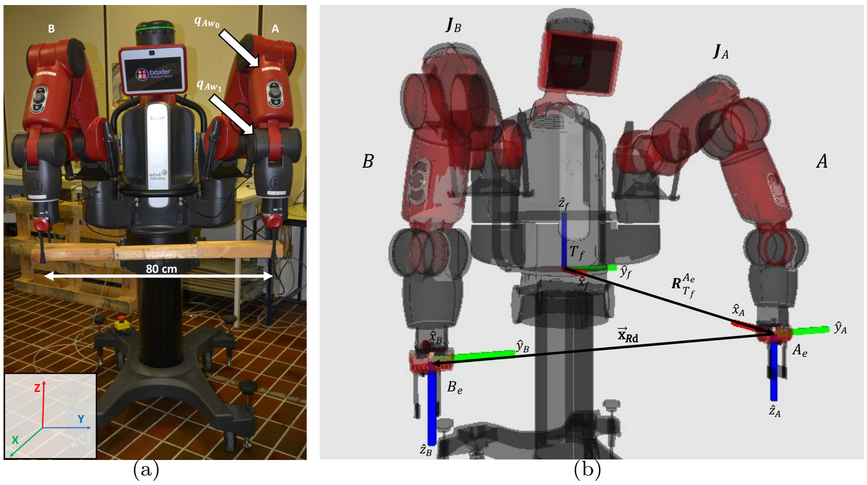


Figure 5.22: Study case setup (a) and coordinate transformation of the Baxter arms (b)

The proposed task can be considered as a typical manufacturing application, in which a dual-arm robot has to pick and move a large object by using both arms as shown in Figure 5.22(a). Therefore, this task requires a synchronized arms motion and respect of all joint position constraints. In order to apply the relative Jacobian method, the coordinate transformation of the Baxter arms is shown in Figure 5.22(b).

The motion assigned to the Baxter arms consists in a circle path with radius $r = 0.13m$ on the $X - Y$ plane, which has to be completed with an execution time $t_{ex} = 35s$. In order to ensure the gripping of the object during the whole path, the relative end-effectors distance must remain equal to its initial value $\vec{\boldsymbol{P}}_{\boldsymbol{R}} = 0.8m$. During the execution of the task, two joints, respectively $q_{A_{w0}}$ and $q_{A_{w1}}$, violate their maximum joint limits, which are respectively placed at $0.16rad$ and $0.95rad$. In order to show the efficacy of the proposed strategy, the task has been repeated three times in accordance with the three cases described in Table 5.5, while the controller parameters that have been used for all three cases, are reported in Table 5.6. In particular, the gain matrix coefficients k_i and h_i have been empirically tuned, and their values depend on β values and the desired joint positions.

Case I shows the motion of the end effector A has been performed without the imposition of any limit on its joints. The position values obtained for the considered joints, are shown in Figure 5.23. In particular, $q_{A_{w0}}$ is a critical joint

Case	**Controller**	**Supervisor**
I	*Relative Jacobian without joint limit avoidance (see (4.12))*	*Off*
II	*Relative Jacobian with joint limit avoidance (see (5.12))*	*Off*
III	*Relative Jacobian with joint limit avoidance (see (5.12))*	*On*

Table 5.5: Description of the experimental considered cases.

Critic. joint	$q_{L_{[min,max]}}$	$q_{T_{[min,max]}}$	β	k_i	h_i
$q_{A_{w0}}$	$[-3, 0.16]$	$[-2.96, 0.12]$	0.04	100	10
$q_{A_{w1}}$	$[-1.57, 0.95]$	$[-1.47, 0.85]$	0.1	100	50

Table 5.6: Parameters used for the experimental cases.

for the whole duration task as shown in Figure 5.23(a), while $q_{A_{w1}}$ becomes a critical joint only inside the time interval $[13s, 24s]$ (see Figure 5.23(b)). Since no joint position limit avoidance is considered in this case, both joints violate their maximum position limits.

Case II is based on the use of redundant motion possessed by the manipulator A. Since it has only a degree of redundancy, $q_{A_{w0}}$ is able to avoid its limit and it becomes a not critical joint after $t = 14s$. Moreover, the degree of redundancy permits to track correctly the desired path, as shown in Figure 5.24(a). However, when $q_{A_{w1}}$ becomes a critical joint, it cannot use any redundant motion because the unique redundant degree is yet employed by $q_{A_{w0}}$. For this reason, $q_{A_{w1}}$ is not able to satisfy its limit, as shown Figure 5.23(b).

Case III allows to overcome the problem described in Case II by turning on the supervisor. In fact, when the supervisory controller recognizes $q_{A_{w1}}$ like a critical joint, the controller assigns a higher priority execution level to the joint limit avoidance task of $q_{A_{w1}}$, such that, this possesses the same priority level of the path following task. Therefore, in this case both $q_{A_{w0}}$ and $q_{A_{w1}}$ are able to avoid their limits. The disadvantage of this strategy concerns the partial and temporary path following error due to the limitation of $q_{A_{w1}}$, as shown in Figure 5.24(a) and Figure 5.25(a). However, the path following error does not affect the desired relative end-effectors motion as demonstrated in Figure 5.25(b), because it has the highest priority execution level.

Discussion

The proposed supervisory kinematic controller is able to ensure the joint position constraints in cooperative manipulation even when the number of redun-

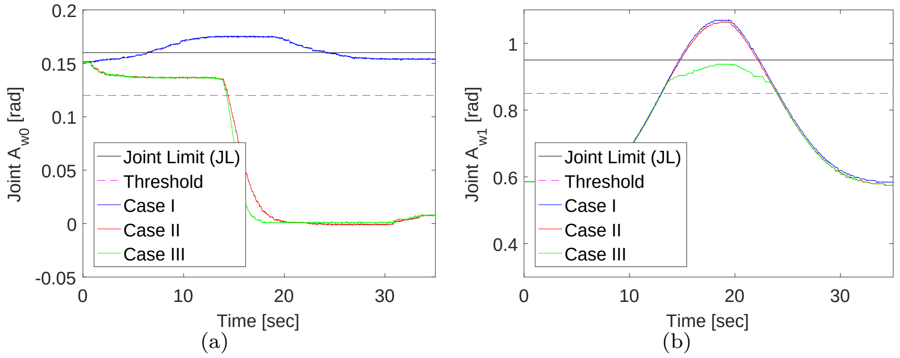


Figure 5.23: Controller treats joint limits enforced in manipulators.

dant system motions are not sufficient to guarantee them. This is achieved by a hierarchical execution priority which is dynamically assigned to each joint the first time that it becomes critical; when the number of redundant motion is no longer sufficient, then the joint limit avoidance task receives the same priority level of the path following task. This causes a gradual and temporary reduction of the path following performances, but ensures that safety critical joint constraints are not violated. The experimental results show that the approach avoids correctly the limits of the critical joints, while the path following error, which depends on the motion of those joints that cannot be managed via redundancy, remains small. The study has been performed on a Baxter dual-arm system, but the proposed strategy can be easily extended to heterogeneous cooperative manipulators, as long as their application does not require high path tracking accuracy. At the present state, the supervisory controller assigns the priority dynamically when a joint becomes critical. However the

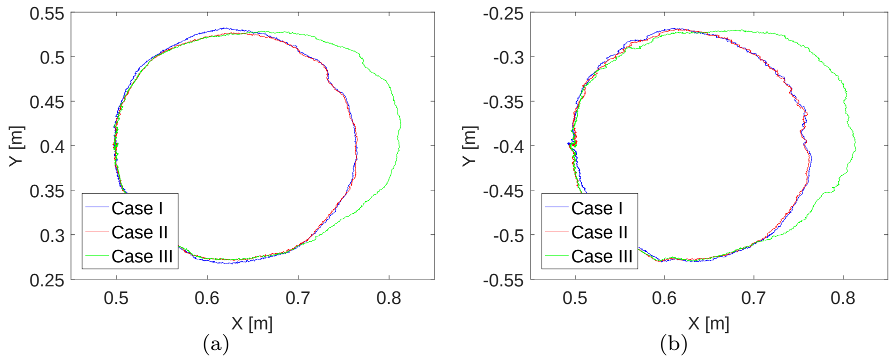


Figure 5.24: Coordinated manipulation results.

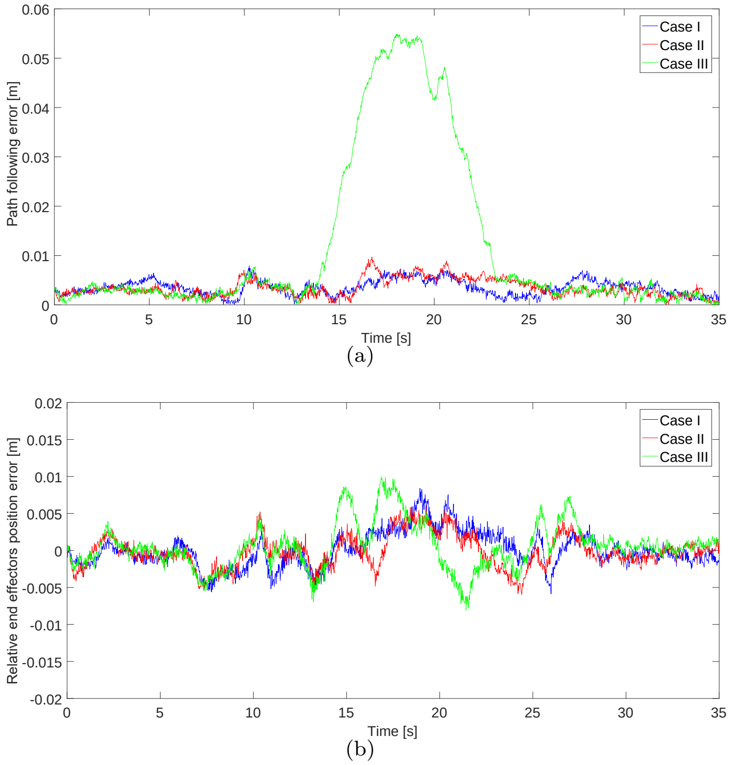


Figure 5.25: Cooperative performance of tightly-coupled manipulators.

assigned priority level does not change any-more with time (e.g., when a joint is no more critical, or a new redundancy degree becomes available). Therefore an appealing topic for further research concerns in time-variant assignment of priority execution levels.

Chapter 6

Proposed fault tolerant algorithms

6.1 Relative manipulability index

As it has been introduced in the Chapter I, a desired propriety of a dual arm system consists of being fault tolerant with respect to one or more faults, therefore this last Chapter is focused on the analysis of dual cooperative arms when a Partial Loss of Joint Torque Faults (PLJTF) [29] is occured.

The basic idea of this study consists of implementing a kinematic controller, which uses the degree of redundancy both to maximize the local optimum fault tolerance configuration and to overcome the loss of the end effector velocity due to fault occurrence by using of the SNS method as shown in [54].

The optimal fault tolerant configuration is the one that maximizes the residual manipulability of the manipulator affected by a joint fault [52]. As it has been shown in section 3.2.3, the manipulability $w(\vec{\boldsymbol{q}})$ of a robotic arm is a kinematic index that quantifies the performance of the manipulator [70]. It depends on the joint positions $\vec{\boldsymbol{q}}$ which describes the current configuration of the manipulator. The manipulability value is calculated through the objective function

$$w(\vec{\boldsymbol{q}}) = \sqrt{det(\boldsymbol{J}(\vec{\boldsymbol{q}})\boldsymbol{J}(\vec{\boldsymbol{q}})^T)} \tag{6.1}$$

where $J(\vec{\boldsymbol{q}})$ represents the Jacobian (geometrical or analytical) [55].

The manipulability index is a non-negative value that is equal to zero at the singular configurations of the manipulator. Therefore, the manipulator performance is directly dependent on the manipulability index, where higher $w(\vec{\boldsymbol{q}})$ values correspond to higher performance levels of the manipulator and vice-versa.

The loss of performance level due to a faulty joint can be measured by calculating the relative manipulability index $r_i(\vec{\boldsymbol{q}})$ [52, 71], which is the ratio between the manipulability index after the fault $w_i(\vec{\boldsymbol{q}})$ and the manipulability index before the fault $w(\vec{\boldsymbol{q}})$

$$r_i(\vec{\boldsymbol{q}}) = \frac{w_i(\vec{\boldsymbol{q}})}{w(\vec{\boldsymbol{q}})} \tag{6.2}$$

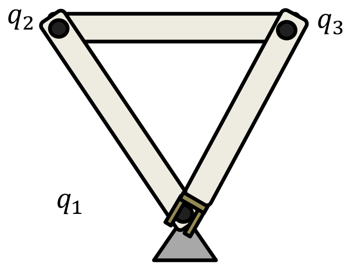


Figure 6.1: 3 DoF planar manipulator with fault tolerant configuration with respect to joint one (q_1) (revised version of Figure 2 from [52])

where the subscript i denotes the i-th faulty joint. The relative manipulability index, as well as the manipulability index, depends on the current configuration of the robot $\vec{\boldsymbol{q}}$. As proven in [52], if all joint faults are equally likely, the following property holds for the relative manipulability index

$$\sum_{i=1}^{N} r_i^2 = n - m \tag{6.3}$$

where n is the number of joints and m is the degree of motion required by the task, and their difference represents the degree of the redundancy. If the manipulator has three joints and only one degree of redundancy, then (6.3) can be rewritten as

$$r_1^2 + r_2^2 + r_3^2 = 1 \tag{6.4}$$

which shows the level of fault tolerance for each joint in a given configuration assumed by the manipulator.

For example, if $r_1 = 1$, for a specific configuration the end effector motion is fault tolerant respect to the fault on the first joint. A practical explanation is that, in the considered configuration, the first joint gives no contribution to the end effector motion, as shown in Fig6.1.

On the other hand, Fig6.1 shows that the manipulator's configuration is not fault tolerant with respect to any of the other joints ($r_2 = r_3 = 0$). It is therefore desirable to find those configurations which ensure a uniform distribution of the relative manipulability indices, i.e., the configurations for which the relative indices have the same value with respect to each joint fault. Without *a priori* knowledge of the fault distribution, these configurations provide the best fault tolerant capabilities.

6.2 Optimization of fault tolerant configuration

The relative manipulability index is strictly related to the null space of the Jacobian. For a manipulator with one degree of redundancy, relative manipulability index can be obtained by calculating the null vector $\vec{\boldsymbol{n}}_{\boldsymbol{J}}$ of the Jacobian $\boldsymbol{J}$. In detail, it holds that

$$n_i = (-1)^{i+1} det(\boldsymbol{J}_i) \qquad i = 1 \dots n \tag{6.5}$$

where n_i is the i-th component of $\vec{\boldsymbol{n}}_J$. (6.5) has been obtained by using of the Laplace expansion of the determinant [72], which for a rectangular Jacobian matrix of dimensions 2×3, it results to be the cross product of the two rows of $\boldsymbol{J}$. By considering only the absolute value of both sides of (6.5), we obtain

$$|n_i| = |det(\boldsymbol{J}_i)| = w_i(\vec{\boldsymbol{q}}) \tag{6.6}$$

from which follows

$$w(\vec{\boldsymbol{q}}) = |\vec{\boldsymbol{n}}_J| \tag{6.7}$$

For each manipulator considered in this chapter (i.e., see the manipulator A in Figure 6.2), the Jacobian null vector $\vec{\boldsymbol{n}}_J$ has the form shown in 6.8, as demonstrated in [52].

$$\vec{\boldsymbol{n}}_J = \begin{bmatrix} sin(q_3) \\ -sin(q_3) - sin(q_2 + q_3) \\ sin(q_2) + sin(q_2 + q_3) \end{bmatrix} \tag{6.8}$$

By considering the unit length null vector of the Jacobian $\hat{\vec{\boldsymbol{n}}}_J$ as

$$\hat{\vec{\boldsymbol{n}}}_J = \frac{\vec{\boldsymbol{n}}_J}{|\vec{\boldsymbol{n}}_J|} \tag{6.9}$$

it is possible to obtain

$$w_i(\vec{\boldsymbol{q}}) = |n_i| = |\hat{n}_i| w(\vec{\boldsymbol{q}}) \tag{6.10}$$

where $\hat{n}_i$ is the i-th component of $\hat{\vec{\boldsymbol{n}}}_J$. From Eqs.(6.2) and (6.10), the relative manipulability index results

$$r_i = |\hat{n}_i| \tag{6.11}$$

This expression is very important to find the optimal fault tolerance configuration. In fact, it is possible to obtain the same value of relative manipulability indexes respect to several joint faults, by imposing an equal magnitude to the Jacobian null space components $|\hat{n}_i|$ in (6.8) (when $q_2 + q_3 = k\pi$), as demonstrated in [52].

Therefore, it is possible to derive a scalar objective function H, which defines the distance between the configuration assumed by the manipulator ($\vec{\boldsymbol{q}}$) and the optimal fault tolerant configuration ($\vec{\boldsymbol{q}}_{\mathbf{0}}$):

$$H(\vec{\boldsymbol{q}}) = -\frac{1}{2}(\vec{\boldsymbol{q}} - \vec{\boldsymbol{q}}_{\mathbf{0}})^T(\vec{\boldsymbol{q}} - \vec{\boldsymbol{q}}_{\mathbf{0}}) \tag{6.12}$$

It is possible to minimize such distance in the joint space by using of the gradient, which is projected into the Jacobian null space of the manipulator affected by the fault [42].

6.3 Proposed fault tolerant configuration algorithm

The joints velocity vector $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{ab}_k}$ can be calculated by the right pseudo-inverse of Jacobian matrix $\boldsymbol{J_A}^\dagger$

$$\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{ab}} = \boldsymbol{J_R}^\dagger \dot{\vec{\boldsymbol{x}}}_{\boldsymbol{R}} + \boldsymbol{P_R}([\boldsymbol{J_A} \;\; \mathbf{0}]^\dagger \dot{\vec{\boldsymbol{x}}}_{\boldsymbol{A}}) \tag{6.13}$$

where $\boldsymbol{P_R} = (\boldsymbol{I} - \boldsymbol{J_R}^\dagger \boldsymbol{J_R})$ is the orthogonal projector matrix which projects the absolute end-effector velocity $\boldsymbol{A}$ related to manipulator A into the J_R null space. Note that in order to simplify the notation, in (6.13) and from now on, the dependencies of the Jacobian matrices on the joint position vectors and on sample time, are removed unless otherwise specified.

Since, the study is focused on two planar manipulators having a degree of redundancy, such that it is possible to solve their redundancy by optimizing the fault tolerant configuration function (6.12) through the Gradient Projector method

$$\dot{\vec{\boldsymbol{q}}}_H = k_q \begin{bmatrix} \nabla_q |H(\vec{\boldsymbol{q}}_a)| \\ \nabla_q |H(\vec{\boldsymbol{q}}_b)| \end{bmatrix} \tag{6.14}$$

where $\dot{\vec{\boldsymbol{q}}}_H$ is a n_{ab} dimensional vector and k_q is the gain of the gradient. The objective function H is projected into the manipulators Jacobian null space, so that, (6.13) can be rewritten in the following form

$$\dot{\vec{\boldsymbol{q}}}^*_{\boldsymbol{ab}} = \boldsymbol{J_R}^\dagger \dot{\vec{\boldsymbol{x}}}_{\boldsymbol{R}} + \boldsymbol{P_R}([\boldsymbol{J_A} \;\; \mathbf{0}]^\dagger \dot{\vec{\boldsymbol{x}}}_{\boldsymbol{A}} + \boldsymbol{P_{ab}} \dot{\vec{\boldsymbol{q}}}_{\boldsymbol{H}}) \tag{6.15}$$

where $\boldsymbol{P_{ab}} = [\boldsymbol{P_a} \;\; \boldsymbol{P_b}]$ is a $n_{ab} \times n_{ab}$ dimensional matrix that contains the orthogonal projectors into the null spaces of the manipulators A and B which dimension is different from zero. The (6.15) presents a hierarchical task structure [61], where $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{R}}$ is the task having the higher execution priority, while $\dot{\vec{\boldsymbol{q}}}_{\boldsymbol{H}}$ is the lower task priority. In particular, this last one operates on the current configuration of the individual manipulators, without influencing the end effectors motions defined by $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{R}}$ and $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{A}}$. In turn, $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{A}}$ does not influence the relative effectors motion, which possesses the higher execution priority. Therefore, if the three tasks conflict with each other, the performance of $\dot{\vec{x}}_{\boldsymbol{R}}$ motion is protected at the expense of the other two tasks.

6.4 Joint fault management with the SNS method

When a PLJTF occurs, it is treated like a joint having a reduction of its velocity bound. In this scenario, it is possible to compensate the degradation of the system performance by using the SNS method [48]. As described in Chapter 3 (see 3.2.4), this method allows to find an efficient solution to the pseudo-inverse problem for each kind of joint constraint, such as joint position bound $\vec{q}_{\boldsymbol{f}}$, joint velocity bound $\dot{\vec{q}}_{\boldsymbol{f}}$ and joint acceleration bound $\ddot{\vec{q}}_{\boldsymbol{f}}$. Therefore, it is possible to introduce the SNS expression relative to $\dot{\vec{q}}_{\boldsymbol{f}}$ described in 3.21, in order to obtain a new joints velocity vector $\dot{\vec{q}}^{*}_{SNS}$:

$$\dot{\vec{q}}^{*}_{SNS} = (\boldsymbol{J_R W_R})^{\dagger} s\dot{\vec{x}}_{\boldsymbol{R}} + \boldsymbol{P}^{*}_{\boldsymbol{R}}([\boldsymbol{J_A W_A} \;\; \boldsymbol{0}]^{\dagger}\dot{\vec{x}}_{\boldsymbol{A}} + \boldsymbol{P}^{*}_{\boldsymbol{ab}}(\dot{\vec{q}}_{\boldsymbol{H}} + \dot{\vec{q}}_{\boldsymbol{f}})) \tag{6.16}$$

where s is the task scaling factor ($s \in [0,1]$), while $\boldsymbol{W_R}$, $\boldsymbol{P_R}^{*}$ and $\boldsymbol{P}^{*}_{\boldsymbol{ab}}$ are defined as

$$\boldsymbol{W_R} = \begin{bmatrix} \boldsymbol{W_A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W_B} \end{bmatrix} \tag{6.17}$$

$$\boldsymbol{P_R}^{*} = (\boldsymbol{I} - (\boldsymbol{J_R W_R})^{\dagger}\boldsymbol{J_R}) \tag{6.18}$$

$$\boldsymbol{P}^{*}_{\boldsymbol{ab}} = \begin{bmatrix} \boldsymbol{I} - (\boldsymbol{J_A W_A})^{\dagger}\boldsymbol{J_A} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} - (\boldsymbol{J_B W_B})^{\dagger}\boldsymbol{J_B} \end{bmatrix} \tag{6.19}$$

and $\boldsymbol{W_B}$, $\boldsymbol{W_A}$ are diagonal matrices used for zeroing the columns of the two Jacobian manipulators $\boldsymbol{J_A}$ and $\boldsymbol{J_B}$, respectively.

Since each manipulator possesses only one degree of redundancy, it is not sufficient to project both $\dot{\vec{q}}_{\boldsymbol{H}}$ and $\dot{\vec{q}}_{\boldsymbol{f}}$ into the Jacobian null space in the same time. Hence if the manipulators are affected by the fault, the controller sets the gradient gain k_q to 0 in order to switch from current fault tolerant configuration to a new configuration due to SNS method. Moreover, the controller ensures that the column of $\boldsymbol{W}_R$ relative to the position of the faulty joint becomes zero, in order to obtain the reduced form of the relative Jacobian. In this way, the i-th faulty joint keeps the maximum permitted velocity $\dot{q}_{f_i}$, while the unperformed velocity is mapped into healthy joints by projecting into the reduced relative Jacobian null space. The new obtained manipulator pose allows to complete the task without affecting the relative end effectors motion.

6.5 Case study

Description of the task

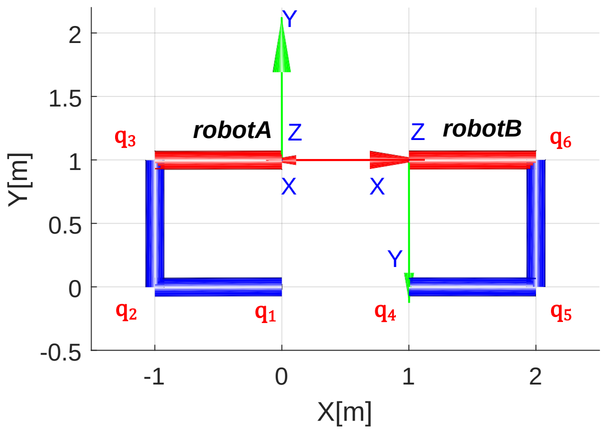


Figure 6.2: Starting configuration of the dual arm system

The case study is focused on the cooperation of two planar manipulators, namely A and B, having three degrees of motion ($n_a = n_b = 3$) and the same geometry. The joint velocity bounds in normal working is defined by $\dot{\vec{\boldsymbol{q}}}_M = [\pm 0.25, \pm 0.1, \pm 0.1, \pm 0.1, \pm 0.15, \pm 0.1]^T$ rad/s.

The specific task consists in the translation of the end effectors along a predefined straight line between $\vec{\boldsymbol{p}}_{\boldsymbol{1}} = [0, 1]^T$ m, the starting end effector position, and $\vec{\boldsymbol{p}}_{\boldsymbol{2}} = [1, 1]^T$ m, the ending end effector position, while the relative end effectors distance and orientation remain constant during the whole motion.

In order to execute this task, two reference velocity paths are fixed, namely the desired relative end effectors motion $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{R}} = [0, 0, 0]^T$ m/s, and the absolute end effector trajectory $\dot{\vec{\boldsymbol{x}}}_{\boldsymbol{A}} = [0.01, 0]^T$ m/s. In order to implement the condition of the optimal fault tolerant configuration shown in (6.12), the two manipulators are made kinematically redundant by excluding the absolute end effector orientation velocity from the task specifications. Figure 6.2 shows the starting pose of the two manipulators. This is an optimal fault tolerance configuration because the two $\vec{\boldsymbol{n}}_J$ vectors relative to both manipulators, posses the components $\hat{n}_i$ with magnitude equal to 0.577.

Moreover, since the starting manipulator pose is the same for both manipulators and the relative end effectors orientation is kept constant during the motion, it is possible to obtain the fault tolerant configurations of both manipulators by limiting the minimization of the objective function $\dot{\vec{\boldsymbol{q}}}_H$ for the manipulator A, and thus (6.14) can be rewritten as

$$\dot{\vec{q}}_H = k_q \begin{bmatrix} \nabla_q |H(\vec{q}_a)| \\ \mathbf{0} \end{bmatrix} \tag{6.20}$$

Several tests have been performed, imposing different fault magnitudes to one joint at a time. Two levels of maximum rotation velocity reduction have been verified, namely 50% and 80%. Moreover, the performance of the system has been simulated in four different fault instant times, i.e., $5s$, $6s$, $7s$ and $8s$.

6.5.1 Results

Figures (6.3)-(6.4) show the relative manipulability indices obtained for the two different fault magnitudes, such as 50% and 80% reduction of maximum joint velocity, at the mentioned fault instant times.

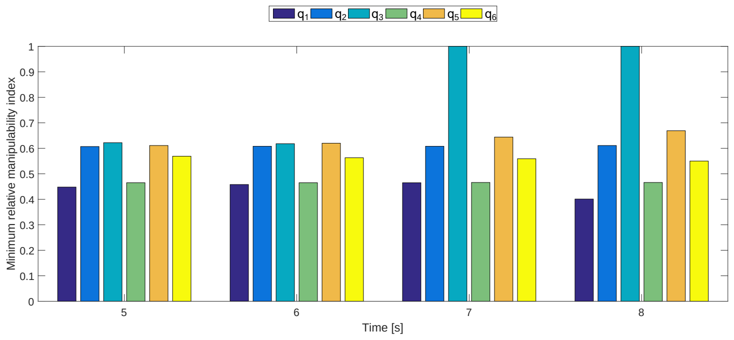


Figure 6.3: Relative manipulability indices for 50% reduction of maximum joint velocity

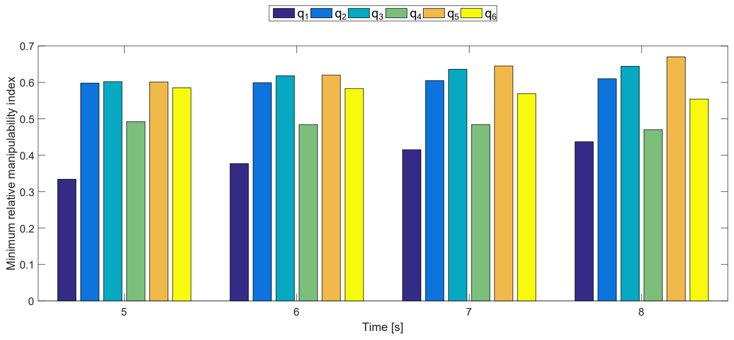


Figure 6.4: Relative manipulability indices for 80% reduction of maximum joint velocity

It is possible to note how the six relative manipulability values are very similar for both considered faults. The results show how r_i indices are more dependent on the time of the fault occurrence rather than the fault magnitude. Moreover, the r_i indices relative to the manipulator A (namely q_1, q_2 and q_3) increase when the fault occurs towards the end of the task, contrary to the r_i indices relative to the manipulator B. This depends on the specific task motion, where the whole system translate towards the manipulator B position in order to affect its manipulability index. Moreover, it is possible to note that q_1 possesses the lower r_i index, and thus provides the main motion contribute to the task. On the other hand, the r_i index of q_3 is equal to one after $t = 7$ s (see Figure 6.3), because the fault does not affect the desired velocity required by q_3 to complete the task.

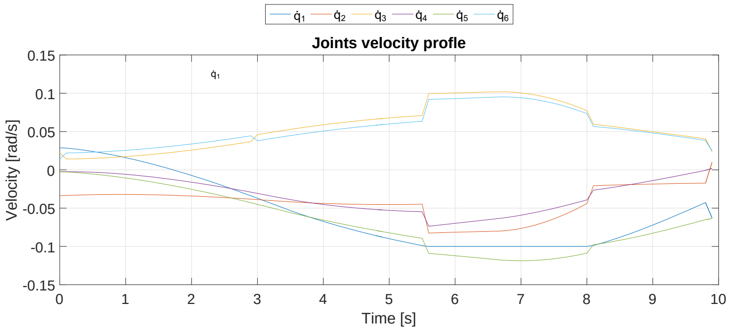


Figure 6.5: Joint velocity when the maximum $\dot{q}_1$ velocity is reduced of 50% at t=5s

Figures 6.5 and 6.6 show the joints velocity when the reduction of velocity of $\dot{q}_1$ and $\dot{q}_3$ is equal to 50% and occurs at $t = 5$ s. The unperformed velocity

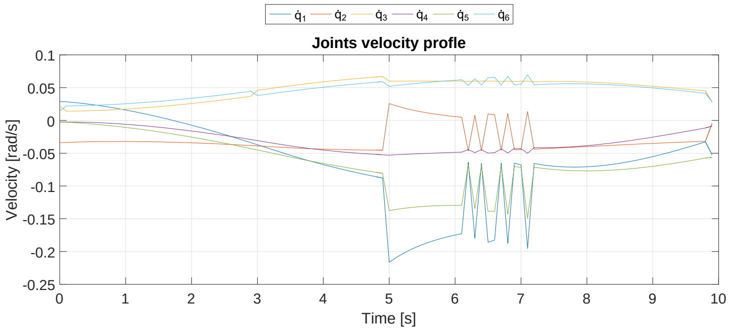
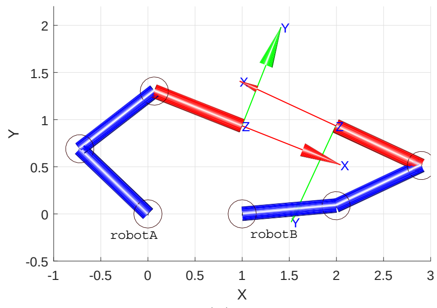
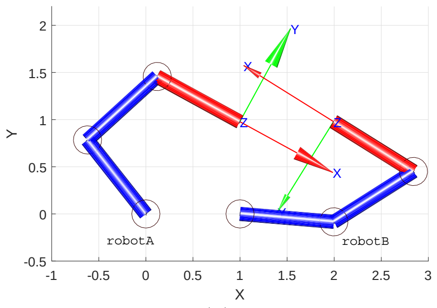


Figure 6.6: Joint velocity when the maximum $\dot{q}_3$ velocity is reduced of 50% at t=5s

is projected into the two reduced Jacobian null space. The last one increases the velocity of the healthy joints, which change the current configuration of the manipulators without influencing the task motion. However, when a fault occurs on $\dot{q}_3$, the velocity of $\dot{q}_1$ tends to its maximum value (0.25 rad/s). In order to avoid the saturation of the healthy joints velocity, it is possible to reduce the task velocity by choosing an appropriate value of s in (6.16). Moreover some joint velocities, such as $\dot{q}_1$, $\dot{q}_2$ and $\dot{q}_5$ show high chattering behaviours due to the switching controller, when $\dot{q}_3$ is close to its fault bound. These could involve several discontinuities into the acceleration levels, which can be eliminated by implementing a second order kinematic algorithm. Finally, Figure 6.7 shows the final configurations of the dual arm system, when a fault of 80% occurs on $\dot{q}_1$ or $\dot{q}_3$ at $t = 5$ s.



(a)



(b)

Figure 6.7: Final manipulators configuration, when a 80% maximum velocity loss on (a) $\dot{q}_1$ and (b) $\dot{q}_3$ occurs at t=5s

6.5.2 Discussion

This method changes the current configuration of both manipulators without affecting the desired relative end effectors distance. The results have shown that, in case of joint faults, the proposed method permits to correctly complete the assigned task. In this study, the faulty joint should be isolated and the fault magnitude is required to be known; this is typically achieved by implementing a fault diagnosis module. When a fault occurs, several discontinuities appear in the acceleration level. In order to solve this problem, the proposed controller can be replaced by a second order kinematic controller.

Chapter 7

Conclusions and future works

7.1 Conclusions

This thesis tackles the problem of kinematic control for cooperative dual-arm systems via redundancy resolution. The main contribution is the development of a kinematic controller for cooperative manipulation based on the relative Jacobian method, in order to treat two cooperative manipulators as a unique redundant manipulator, while achieving a secondary task as well. In detail, in this thesis the redundancy resolution algorithm based on Jacobian null space method with hierarchical prioritized task architecture has been developed, in order to satisfy some important secondary tasks: obstacle avoidance, joints position limits avoidance, and the relative manipulability index.

The obtained results have demonstrated the efficiency of the proposed redundancy resolution algorithms, because their performance were compatible with those of the single manipulator case. However, the low execution priority of the secondary task did not permit to guarantee the execution of secondary tasks, when the system did not have a sufficient number of redundant motions. The proposed solution is based on increasing the priority level relative to the secondary tasks, which degrades partially and temporary the performance of primary tasks. This technique has been studied during the cooperation between a non redundant manipulator with a redundant manipulator and successively implemented in a supervisor controller, in order to manage the temporary unavailability of redundant motions in the cooperation of two homogeneous redundant manipulators. Therefore this technique results to be flexible, because it is easily implementable on several types of manipulators, whose tasks do not require high path tracking accuracy.

An additional contribution of this thesis is the application of the proposed redundancy resolution algorithm for tolerating joint faults. The fault has been modelled like a reduction of the maximum joint velocity due to partial torque loss of a servomotor. The system has been made locally fault tolerant by exploiting its redundancy degrees via two different methods. The first method has permitted to reduce the manipulators' manipulability loss in case of fault occurrence, by imposing an optimal fault tolerant configuration to both manipulators; this was directly obtained from optimization of relative manipulability index. The second method has permitted to compensate the loss of the end effectors motion by using the saturation null space approach. The feasibility of the proposed joint fault tolerant control has been shown in a case study, whose results have demonstrated that the efficacy depends on two factors: fault intensity and the fault time instants.

7.2 Future works

Future works should focuses on improving the proposed kinematic controllers in the cooperative manipulation field. First of all, the relative Jacobian method can be extended on cooperative manipulators mounted on different mobile bases, in order to increase the number of redundant motions possessed by each system. Secondary, the redundancy resolution can be extended to grasping motion control, when a robotic hand with multi-fingers (e.g., anthropomorphic hand model) is used to grasp a large and heavy object.

Regarding to the proposed fault tolerance algorithm, the stability of the switching controller should be proved, and experimental tests should be performed as well. Since in the considered study case the position and intensity of joint fault is known a priory, an interesting improvement of this algorithm concerns the integration of a fault detection and diagnosis algorithm, which allows to detect the presence of faults on multiple joints and evaluate the type of faults.

Finally, others redundancy resolution technique different from Jacobian null space (i.e., General-weighted least-norm method) can be implemented, in order to compare the different cooperation performances obtained by using different techniques.

Bibliography

[1] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic, "Dual arm manipulation—a survey," *Robot. Auton. syst.*, vol. 60, no. 10, pp. 1340 – 1353, 2012.

[2] J. F. Liu and K. Abdel-Malek, "Robust control of planar dual-arm cooperative manipulators," *Robotics and Computer-Integrated Manufacturing*, vol. 16, no. 2, pp. 109–119, 2000.

[3] P. Tsarouchi, S. Makris, G. Michalos, M. Stefos, K. Fourtakas, K. Kaltsoukalas, D. Kontovrakis, and C. Chryssolouris, "Robotized assembly process using dual arm robot," *Procedia CIRP*, vol. 23, pp. 47 – 52, 2014.

[4] J. Krüger, G. Schreck, and D. Surdilovic, "Dual arm robot for flexible and cooperative assembly," *CIRP Annals-Manufacturing Technology*, vol. 60, no. 1, pp. 5–8, 2011.

[5] V. Lippiello, L. Villani, and B. Siciliano, "An open architecture for sensory feedback control of a dualarm industrial robotic cell," *Industrial Robot: An International Journal*, vol. 34, no. 1, pp. 46–53, 2007.

[6] D. J. Cox, "Mock-up of hazardous material handling tasks using a dual-arm robotic system," in *Automation Congress, 2002 Proceedings of the 5th Biannual World*, 2002, vol. 14, pp. 527–532.

[7] D. W. Seward and M .J Bakari, "The use of robotics and automation in nuclear decommissioning," in *22nd International Symposium on Automation and Robotics in Construction ISARC*, 2005, pp. 11–14.

[8] M. J. Bakari, K. M. Zied, and D. W. Seward, "Development of a multi-arm mobile robot for nuclear decommissioning tasks," *International Journal of Advanced Robotic Systems*, vol. 4, no. 4, pp. 502–524, 2007.

[9] E. Simetti, G. Casalino, N. Manerikar, A. Sperindé, S. Torelli, and F. Wanderlingh, "Cooperation between autonomous underwater vehicle manipulations systems with minimal information exchange," in *OCEANS 2015 - Genova*, May 2015, pp. 1–6.

[10] H. Farivarnejad and S Ali A Moosavian, "Multiple impedance control for object manipulation by a dual arm underwater vehicle–manipulator system," *Ocean Engineering*, vol. 89, pp. 82–98, 2014.

[11] A. Stroupe, A. Okon, M Robinson, T. Huntsberger, H. Aghazarian, and E. Baumgartner, "Sustainable cooperative robotic technologies for human and robotic outpost infrastructure construction and maintenance," *Auton. Robots*, vol. 20, no. 2, pp. 113–123, Mar. 2006.

[12] E. Zereik, A. Sorbara, G. Casalino, and F. Didot, "Autonomous dual-arm mobile manipulator crew assistant for surface operations: Force/vision-guided grasping," in *Int. Conf. Recent Advances in Space Tech.*, June 2009, pp. 710–715.

[13] R. Simmons, S. Singh, D. Hershberger, J. Ramos, and T. Smith, "First results in the coordination of heterogeneous robots for large-scale assembly," in *Proc. Int. Symp. Experimental Robotics*, 2000.

[14] R.C. Goert, "Fundamentals of general-purpose remote manipulators," *Nucleonics (U.S.) Ceased publication*, vol. Vol: 10, No. 11, Nov 1952.

[15] R. O. Ambrose, H. Aldridge, R. S. Askew, R. R. Burridge, W. Bluethmann, M. Diftler, C. Lovchik, D. Magruder, and F. Rehnmark, "Robonaut: Nasa's space humanoid," *IEEE Intelligent Systems and their Applications*, vol. 15, no. 4, pp. 57–63, Jul 2000.

[16] P. Hynes, G. I. Dodds, and A.J.Wilkinson, "Uncalibrated visual-servoing of a dual-arm robot for mis suturing," in *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.* IEEE, 2006, pp. 420–425.

[17] F. Benetazzo, F. Ferraacuti, A. Freddi, A. Giantomassi, S. Iarlori, S. Longhi, A. Monteriù, and D. Ortenzi, *AAL Technologies for Independent Life of Elderly People*, pp. 329–343, Springer International Publishing, 2015.

[18] J. Xu, G. G. Grindle, B. Salatin, J. J. Vazquez, H. Wang, D. Ding, and R. A. Cooper, "Enhanced bimanual manipulation assistance with the personal mobility and manipulation appliance (permma)," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on.* IEEE, 2010, pp. 5042–5047.

[19] C. S. Chung, H. Wang, and R. A. Cooper, "Autonomous function of wheelchair-mounted robotic manipulators to perform daily activities," in *2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR)*, June 2013, pp. 1–6.

[20] A. Cunningham, W. Keddy-Hector, U. Sinha, D. Whalen, D. Kruse, J. Braasch, and J. T. Wen, "Jamster: A mobile dual-arm assistive robot with jamboxx control," in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, Aug 2014, pp. 509–514.

[21] F. Aghili and K. Parsa, "A reconfigurable robot with lockable cylindrical joints," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 785–797, 2009.

[22] L.Zlajpah and B. Nemec, "Kinematic control algorithms for on-line obstacle avoidance for redundant manipulators," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*. IEEE, 2002, vol. 2, pp. 1898–1903.

[23] M. B. Hong, "On the robot singularity: A novel geometric approach," *International Journal of Advanced Robotic Systems*, vol. 9, 2012.

[24] I. Iossifidis and G. Schoner, "Dynamical systems approach for the autonomous avoidance of obstacles and joint-limits for an redundant robot arm," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 580–585.

[25] J. F. O'Brien and J. T. Wen, "Redundant actuation for improving kinematic manipulability," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*. IEEE, 1999, vol. 2, pp. 1520–1525.

[26] R. S. Jamisola Jr and R. G. Roberts, "A more compact expression of relative jacobian based on individual manipulator jacobians," *Robot. Auton. syst.*, vol. 63, pp. 158 – 164, 2015.

[27] I. Koren and C. M. Krishna, *Fault-tolerant systems*, Morgan Kaufmann, 2010.

[28] J. D. English and A. A. Maciejewski, "Fault tolerance for kinematically redundant manipulators: Anticipating free-swinging joint failures," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 4, pp. 566–575, 1998.

[29] C. L. Lewis and A. A. Maciejewski, "Fault tolerant operation of kinematically redundant manipulators for locked joint failures," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 4, pp. 622–629, 1997.

[30] Y. Zeng, Y. R. Xing, H.J. Ma, and G. H. Yang, "Adaptive fault diagnosis for robot manipulators with multiple actuator and sensor faults," in *The 27th Chinese Control and Decision Conference (2015 CCDC)*. IEEE, 2015, pp. 6569–6574.

[31] A. A. Siqueira, M. H. Terra, and M. Bergerman, *Robust control of robots: fault tolerant approaches*, Springer Science & Business Media, 2011.

[32] C. Paredis, J. J . Christiaan, and P. K. Khosla, "Designing fault-tolerant manipulators: How many degrees of freedom?," *The international journal of robotics research*, vol. 15, no. 6, pp. 611–628, 1996.

[33] R. Isermann, *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*, Springer Science & Business Media, 2006.

[34] Z. Shiller and S. Dubowsky, "On computing the global time-optimal motions of robotic manipulators in the presence of obstacles," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 6, pp. 785–797, 1991.

[35] Z. Shiller, S. Sharma, I. Stern, and A. Stern, "Online obstacle avoidance at high speeds," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1030–1047, 2013.

[36] A. Ben-Israel and T. N. Greville, *Generalized inverses: theory and applications*, vol. 15, Springer Science & Business Media, 2003.

[37] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numerische mathematik*, vol. 14, no. 5, pp. 403–420, 1970.

[38] J. Baillieul, "Kinematic programming alternatives for redundant manipulators," in *Robotics and Automation. Proceedings. 1985 IEEE International Conference on.* IEEE, 1985, vol. 2, pp. 722–728.

[39] P. H. Chang, "A closed-form solution for inverse kinematics of robot manipulators with redundancy," 1987.

[40] L. Sciavicco and B. Siciliano, "A solution algorithm to the inverse kinematic problem for redundant manipulators," *IEEE Journal on Robotics and Automation*, vol. 4, no. 4, pp. 403–410, 1988.

[41] B. Siciliano and O. Khatib, *Springer handbook of robotics*, Springer Science & Business Media, 2008.

[42] R. V. Dubey, J. A. Euler, and S. M. Babcock, "An efficient gradient projection optimization scheme for a seven-degree-of-freedom redundant robot with spherical wrist," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1988, vol. 1, pp. 28–36.

[43] G. Antonelli, F. Arrichiello, and S. Chiaverini, "The null-space-based behavioral control for autonomous robotic systems," *Intell. Serv. Robot.*, vol. 1, no. 1, pp. 27–39, 2008.

[44] S. Chiaverini, “Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators,” *IEEE Trans. Robot. Autom.*, vol. 13, no. 3, pp. 398–410, 1997.

[45] N. Mansard, O. Khatib, and A. Kheddar, “A unified approach to integrate unilateral constraints in the stack of tasks,” *IEEE Trans. Robot.*, vol. 25, no. 3, pp. 670–685, 2009.

[46] A. Atawnih, D. Papageorgiou, and Z. Doulgeri, “Kinematic control of redundant robots with guaranteed joint limit avoidance,” *Robot. Auton. syst.*, vol. 79, pp. 122 – 131, 2016.

[47] N. Mansard and F. Chaumete, “Task sequencing for high-level sensor-based control,” *IEEE Tran. on Robotics*, vol. 23, no. 1, pp. 60–72, 2007.

[48] F. Flacco, A. De Luca, and O. Khatib, “Motion control of redundant robots under joint constraints: Saturation in the null space,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 285–292.

[49] Y. Hu, B. Huang, and G. Z. Yang, “Task-priority redundancy resolution for co-operative control under task conflicts and joint constraints,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2015, pp. 2398–2405.

[50] H.Abdi, A. A. Maciejewski, and S. Nahavandi, “A probabilistic approach for measuring the fault tolerance of robotic manipulators,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1995–2001.

[51] Y. She, W. Xu, H. Su, B. Liang, and H. Shi, “Fault-tolerant analysis and control of ssrms-type manipulators with single-joint failure,” *Acta Astronautica*, vol. 120, pp. 270–286, 2016.

[52] R. G. Roberts and A. A. Maciejewski, “A local measure of fault tolerance for kinematically redundant manipulators,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 543–552, 1996.

[53] A. Freddi, S. Longhi, A. Monteriù, and D. Ortenzi, “Redundancy analysis of cooperative dual-arm manipulators,” *International Journal of Advanced Robotic Systems*, vol. 13, no. 5, 2016.

[54] A. Freddi, S. Longhi, A. Monteriù, and D. Ortenzi, “A kinematic joint fault tolerant control based on relative jacobian method for dual arm manipulation systems,” in *IEEE Conference on Control and Fault-Tolerant Systems (SysTol)*, Barcelona, Spain, 2016.

[55] S. Bruno, S. Lorenzo, V. Luigi, and O. Giuseppe, *Robotics: Modelling, Planning and Control*, Springer Publishing Company, 1st edition, 2008.

[56] A. Colomé and C. Torras, "Closed-loop inverse kinematics for redundant robots: Comparative assessment and two enhancements," *IEEE/ASME Trans. Mechat.*, vol. 20, no. 2, pp. 944–955, 2015.

[57] D.G. Luenberger and Y. YeL, *Linear and nonlinear programming*, Springer Science & Business Media, 2008.

[58] A. De Luca and G. Oriolo, "Reduced gradient method for solving redundancy in robot arms," *Robotersysteme*, vol. 7, no. 2, pp. 117–122, 1991.

[59] J. Lee, P. H. Chang, and R. S. Jamisola, "Relative impedance control for dual-arm robots performing asymmetric bimanual tasks," *IEEE transactions on industrial electronics*, vol. 61, no. 7, pp. 3786–3796, 2014.

[60] B. Cao, G. I. Dodds, and G. W. Irwin, "Redundancy resolution and obstacle avoidance for cooperative industrial robots," *Journal of Robotic Systems*, vol. 16, no. 7, pp. 405–417, 1999.

[61] A. Freddi, S. Longhi, A. Monteriù, and D. Ortenzi, "Redundancy analysis of cooperative dual-arm manipulators," *International Journal of Advanced Robotic Systems*, vol. 13, no. 5, pp. 1729881416657754, 2016.

[62] V. Maheu, P. S. Archambault, J. Frappier, and F. Routhier, "Evaluation of the jaco robotic arm: Clinico-economic study for powered wheelchair users with upper-extremity disabilities," in *IEEE Int. Conf. Rehab. Robot.*, 2011, pp. 1–5.

[63] L. Cavanini, F. Benetazzo, A. Freddi, S. Longhi, and A. Monteriù, "Slam-based autonomous wheelchair navigation system for aal scenarios," in *2014 IEEE/ASME 10th International Conference on Mechatronic and Embedded Systems and Applications (MESA)*, Sept 2014, pp. 1–5.

[64] C. Gionatai, F. Francesco, A. Freddi, S. Iarlori, and A. Monteriù, *An Inertial and QR Code Landmarks-Based Navigation System for Impaired Wheelchair Users*, Springer International Publishing, 2014.

[65] G. Ippoliti, A. Monteriu, L. Jetto, and S. Longhi, *Comparative Analysis of Mobile Robot Localization Methods Based On Proprioceptive and Exteroceptive Sensors*, INTECH Open Access Publisher, 2007.

[66] M. Z. Huang and J. L. Thebert, "A study of workspace and singularity characteristics for design of 3-dof planar parallel robots," *The International Journal of Advanced Manufacturing Technology*, vol. 51, no. 5-8, pp. 789–797, 2010.

[67] M. Z. Huang, "Design of a planar parallel robot for optimal workspace and dexterity," *International Journal of Advanced Robotic Systems*, vol. 8, no. 4, pp. 176–183, 2011.

[68] F. Chaumette and E. Marchand, "A redundancy-based iterative approach for avoiding joint limits: application to visual servoing," *IEEE Trans. Robot. Autom.*, vol. 17, no. 5, pp. 719–730, 2001.

[69] R. Bischoff, J. Kurth, G. Schreiber, R. Koeppe, A. Albu-Schaeffer, A. Beyer, O. Eiberger, S. Haddadin, A. Stemmer, G. Grunwald, and G. Hirzinger, "The kuka-dlr lightweight robot arm - a new reference platform for robotics research and manufacturing," in *Int. Sym. Robot. (ROBOTIK)*, 2010, pp. 1–8.

[70] C. S. Ukidve, J. E. McInroy, and F. Jafari, "Using redundancy to optimize manipulability of stewart platforms," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 4, pp. 475–479, 2008.

[71] R. g. Roberts, H. G. Yu, and A. A. Maciejewski, "Fundamental limitations on designing optimally fault-tolerant redundant manipulators," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1224–1237, 2008.

[72] P. Lancaster and M. Tismenetsky, *The theory of matrices: with applications*, Elsevier, 1985.