Politecnico di Torino

Collegio di Ingegneria Informatica, del Cinema e Meccatronica
Master's Degree in Mechatronic Engineering

Master's Thesis

Accuracy Enhancement of 6DOF Articulated Robots through Geometric and Thermal Error Compensation

Supervisor:
Professor Marina Indri

Company Tutor:
Engineer Joseph Toma
(AXIST S.r.l.)

Candidate:
Vittorio Ascanio

October 2018

I am putting myself to the fullest possible use,
which is all I think that any conscious entity can ever hope to do.
HAL 9000, 2001: A Space Odyssey

Abstract

Industrial robots are able to interact with the external environment and perform given appointed tasks by moving specific working tools, therefore it is extremely important to estimate correctly the position and the orientation of the robot end effector in the 3D space, in order to operate properly a certain tool as desired. This implies the knowledge of the robot kinematic model, i.e. the mathematical expression describing the relation between the displacements of the robot joints and the pose of the robot end effector.

The kinematic model typically stems from the design developed by the manufacturer, hence it cannot suit perfectly any specific robot unit, because of countless unavoidable issues, e.g., deviations from nominal architecture (interface slacks, dimension tolerances, etc.) or variations of operating conditions (motion speed, payload weight, etc.). For this reason, the default accuracy of the vast majority of industrial robots is rather poor and definitely represents a crucial bottleneck especially in those applications involving either precise inspection and measurement tasks or CAD based off-line programmed tasks.

This research activity is primarily aimed at improving significantly the volumetric accuracy of 6DOF articulated robots, by investigating and compensating the model errors driven by geometry discrepancies and temperature changes in the robots structure. Geometric errors constantly affect kinematics and are studied by means of a thorough analysis of the model limitations and the numerical issues. Thermal errors variably affect kinematics and are studied indirectly, i.e. relating, through a simple heuristic model, the robot thermal gradient changes to the robot kinematic parameters variations.

An exhaustive experimental study is set up and carried out on a real articulated robot unit and an entirely genuine software implementation is developed in MATLAB® environment. The algorithmic procedure designed for parameters identification is first simulated numerically and then tested on actual robot specimens in order to prove its speed and reliability. The practical achievements corroborate the theoretical speculations, laying sound bases for further developments.

Contents

Chapter 1

Introduction

1.1 Research Traits

The position in the three dimensional space of the working tool installed on the flange of industrial robots is usually described and evaluated by means of their kinematic model, starting from the knowledge of the corresponding joint displacements. Being only an abstract representation, the kinematic model of any real industrial robot is unavoidably subject to imperfections and deficiencies.

In general, the tool positioning performances of a given robot manipulator are essentially determined by two crucial indices characterising that robot manipulator, namely the repetitive precision, that is a measure of the tool position spread, and the absolute accuracy, that is a measure of the tool position error: the vast majority of industrial robots generally feature quite decent repeatability but rather poor accuracy; on the other hand, repeatability is assessed and reported by the manufacturer but may not be adjusted by the user, whereas accuracy is not assessed nor reported by the manufacturer but may be adjusted by the user.

The enhancement of the default level of accuracy provided by industrial robots, better known as robot manipulator calibration, represents a need of utmost importance for all of those practitioners willing to implement, safely and reliably, a certain range of applications. Unlike the early stages of industrial automations, when the error in the tool position was almost irrelevant for the quality standard of the production process and the manipulator movements were mainly programmed on-line by means of teach pendants, nowadays, industrial robots are also entrusted with delicate or sophisticated tasks, where precision represents a very strict requirement (or even configured as proper measuring instruments) and their motion is often programmed off-line in order to integrate the information included in the CAD models of the objects within the work-cell and to optimise the overall efficiency of the industrial process.

The accuracy of an industrial robot is essentially determined by the discrepancy between the ideal kinematic behaviour expected by that robot model and the real kinematic behaviour exhibited by that robot unit. From a practical point of view, there exist countless elements capable of affecting significantly the accuracy, although only some of

them may be taken into account at the same time by from a practical point of view. In any case, these detrimental elements are routinely grouped according to their nature. In particular, the defects in the kinematic model induced by geometry deviations and temperature variations represent the most significant sources of inaccuracy in industrial robots equipped with light payload.

The kinematic model of a robot manipulator gets usually derived starting from the mechanical description of that robot manipulator provided by its manufacturer. On the other hand, the spatial characteristics of an actual unit necessarily differ from the spatial characteristics of the design model, because of dimension tolerances, interface slacks and lock plays originating as a result of machining and assembling processes, hence the kinematic model is intrinsically subject to geometric errors.

Moreover, industrial robots regularly deal with very large levels of mechanical and electrical power, which necessarily translate into massive dissipation as heat: the consequent variable temperature gradient across the body of a certain unit further modifies its spatial characteristics, hence the kinematic model is also subject to thermal errors.

The primary goal of this activity was thus first to weigh and then to damp the effect of geometric defects and thermal deformations on the accuracy performances of six degrees of freedom articulated robots. More specifically, such a task was carried out studying the kinematic characteristics of 6DOF articulated robots in general, investigating the kinematic and thermal behaviour of the robot manipulator unit under study, formulating a nominal kinematic model with its corresponding parameters from the design data and eventually devising a strategy to adjust that raw model.

The issue of thermal drift is normally addressed simply turning on the controller and standing by a certain amount of time in order for the robot manipulator to warm up sufficiently prior to operating it, which necessarily translates in a loss of time and resources, therefore a real time compensation scheme may give the possibility to keep robots working with stable performances, dramatically boosting the efficiency of the productive process. Considering that thermal error compensation was hinged on repeated trials of geometric error compensation at different temperature conditions, great care has been paid in order to generate a really fast and trustworthy algorithmic parameter identification software implementation by means of an extremely careful analysis of the relation between modelling features and numerical issues.

The applied work has been carried out within the facilities of Axist research and development department on a 6DOF articulated robot specimen there hosted and provided especially for calibration purposes. The core business of Axist consists in providing a wide range of products and services in the field of dimensional measurement and inspection, taking advantage of automated machinery, among which articulated robots often become protagonist thanks to their good level of usage flexibility and motion dexterity. The accuracy of a robot is a major concern for the company as it unavoidably affects the reliability and the quality of the overall measurement system that robot is part of, hence robot calibration becomes an issue of paramount importance in the economy of this enterprise engineering activities.

Owing to the large volume of work daily performed by the department staff, no

research effort had been diverted yet in the area of robot calibration, the management of which was normally entrusted to general purpose professional 3D graphical software solutions. For this reason, the set-up of a fresh scheme of robot calibration of its own was rather desirable by the company, for both incorporating the missing functionalities into some of the projects that have been already developed and replacing the existing functionalities into some of the projects that are still to be developed.

The full project of geometric and thermal calibration has been entirely developed and coded using MATLAB® software environment. Such a development framework proved especially useful also to check preliminarily the dependability of the devised algorithmic procedure through sheer numerical simulation, before actually testing it in the field on a real robot unit. The achieved experimental results are quite promising and represent thus the first step forward towards the development of a project with greater breadth, upon conversion into the software framework compliant with the company needs.

1.2 Contents Structure

The various thematic contents, ranging from pretty general topics to more specific issues, are organised and presented according to their different levels of insight and complexity. Specifically, the thesis is arranged as follows.

- Chapter 2 illustrates the fundamentals of manipulators kinematics, delineating both architectures and characteristics of robotic spatial linkages, laying the theoretical bases for kinematic modelling and reviewing the kinematic performances along with the alternative programming methods as a pretext to introduce the concept of calibration.

- Chapter 3 presents a comprehensive outline of manipulators calibration, specifying the basic framework, investigating the most significant possible target sources of error, itemising the crucial operative stages that make up the whole process and discussing the major issues that may be encountered when the problem is addressed from a practical point of view.

- Chapter 4 reports a complete applied case study of a robot manipulator calibration aimed at both geometric and thermal error compensation, stating first briefly means and goals and detailing then carefully all the devised operative steps of the experimental implemented strategy.

- Chapter 5 describes the target numerical optimisation techniques, portraying the characteristics of problems that originate from model identification along with the most relevant iterative methods aimed at their solution, framing the problem of kinematic parameters identification within the scope of non linear least squares optimisation and surveying its specific numerical properties.

- Chapter 6 develops a kinematic error model, studying infinitesimal perturbations of rigid transformations along with the asymptotic behaviour of the homogeneous

transformation matrices and applying the general results to the reference frames transformations included in the manipulator kinematic model.

- Chapter 7 reviews the overall research activity, recapitulating the attained results and bridging with potential future works.
- Appendix A retraces the basic notions of linear algebra, revising concepts and properties of vectors, matrices, bases and elements of Euclidean geometry.
- Appendix B provides the mathematical tools of rigid body kinematics, characterising rigid transformations, analysing the possibilities for attitude representation and introducing the formalism of homogeneous coordinates.

Chapter 2

Manipulator Kinematics

2.1 Robot Manipulators

2.1.1 Introduction

The Robot Institute of America (RIA) gives its definition of **industrial robot** as "a reprogrammable, multifunctional manipulator designed to move material, parts, tools or specialized devices through variable programmed motions for the performance of a variety of tasks". Such a definition perfectly reflects all the main aspects of machine automation applied to the industrial production context, such as equipment adaptability and reconfigurability, flexible task completion, supervised behaviour scheduling and motion planning, specific application tool handling and object manipulation.

Industrial robot manipulators normally feature very low **autonomy**, as they are required to operate in specific, familiar and established structured environments, but very high **versatility**, as they are able perform a *practically* infinite set of different operations simply by mounting the proper tools.

Some of the most famous sources in the robotics literature, e.g., [1], [2], [5], [6] and [7], are behind the essential notions regarding robot manipulators hereinafter presented.

2.1.2 Kinematic Chain

From a merely mechanical point of view, a manipulator is a system of rigid bodies, termed **links**, connected by movable articulations, termed **joints**: links erect the frame of the structure while joints provide the mobility to the structure. The constrained aggregation of fixed and movable elements constituting the mechanical structure of a manipulator is termed **kinematic chain**. One end of the chain is constrained to a fixed surface and is called **base**; the other end of the chain is attached to a working tool and is called **flange** or **end effector**.

According to the number of kinematic paths between base and flange, it is customary to distinguish between open and closed kinematic chains. The chain is said to be open if there is only one sequence of links and joints connecting base and flange; a manipulator with an open kinematic chain is termed serial manipulator, as it consists of a series of

links and joints in alternating fashion. The chain is said to be closed if there is more than one sequence of links and joints connecting base and flange; a manipulator with a closed kinematic chain is termed parallel manipulator, as it consists of parallel branches of links and joints. The chain is said to be hybrid if only a part of it is closed; a manipulator with a hybrid kinematic chain is termed hybrid manipulator.

Serial manipulators are way more wide-spread because they are endowed with quite good flexibility, maneuverability and dexterity. Parallel manipulators are instead employed to handle heavy payload at high speed as they provide fairly large stiffness, though at the price of limited range of motion and narrow work volume. Hybrid manipulators combine the advantages of both structures: for instance, manipulators designed to lift heavy loads over a wide volume are typically equipped with parallelogram linkages.

2.1.3 Joints Motion

A manipulator manages to move only thanks to the presence of joints. Each joint provides a simple motion, adding a degree of freedom to open kinematic chains. This is the reason why serial manipulators with N joints are improperly termed N-DOF manipulators. The correspondence between number of joints and degrees of freedom is lost for dependent joints and closed kinematic chains.

A joint is composed of two elements, one proximal, called base, and one distal, called follower, in *relative* motion between them: focusing on a single joint, the follower moves while the base remains still. Joints managing to withstand an external effort without moving are said to be active, while joints collapsing under the stress are said to be passive. Sometimes the same joint might behave in both ways.

Joints are also classified according to the kind of motion they yield; even though there exist several types of joints, only two are involved in industrial robotics, namely **prismatic joints**, i.e. sliding pairs providing a linear (translational) motion between the links connected to them and **revolute joints**, i.e. hinged pairs providing an angular (rotational) motion between the links connected to them.

A manipulator is *essentially* a means of operating tools or handling objects conveniently within a certain volume, therefore its basic purpose is to control the motion of its flange, in terms of position and/or orientation. Rotations are necessarily required to orient bodies, hence revolute joints are required when the attitude of the flange has to be adjusted. On the other hand, revolute joints are sometimes preferred to prismatic joints even to modify the position of the flange, owing to the better degree of dexterity they lend to the overall kinematic structure.

2.1.4 Joint Drives

The joints of a manipulator have to be properly driven by a dedicated control system, in order to operate the manipulator as desired. In this respect, active joints are always equipped with **actuators**, i.e. devices entrusted with maneuvering the joint motion, and **sensors**, i.e. devices entrusted with monitoring the joint motion. The combination

of actuators and sensors gives rise to so called servo-mechanisms, i.e. devices able to perform active position control.

Joint Actuators

Joints may be actuated in different ways, according to the desired kinematic and dynamic performances: joints actuators are typically hydraulic, pneumatic or electric. In the vast majority of industrial manipulators, joints are electrically driven, that is to say actuated by means of electric motors, such as brushless DC motors or stepper motors.

Gears are customarily inserted in the joint motion transmission system for a two-fold reason: motion reduction and motion conversion. Reduction is especially useful because high speed motors can be employed even when the desired joint motion is quite slow and, at the same time, the effective load applied to the motor shaft is weakened. Conversion is mandatory whenever the desired kind of joint motion is different from the motion provided by the actuator, in terms of type, e.g., linear and rotary, or alignment, e.g., axis offset or skew. Actuators able to provide *directly* the motion to the joints, that is with no geared transmission buffer whatsoever, are said to be direct drive.

Joint Sensors

Several diverse transducers may be employed for measuring linear or angular joint positions, namely potentiometers, linear encoders, linear variable differential transformers for prismatic joints and potentiometers, resolvers, rotary encoders, rotary variable differential transformers for revolute joints.

Some joint transducers, called absolute transducers, manage to measure the absolute position of a joint, whereas some others, called incremental transducers, manage to measure *only* the relative position of a joint, i.e. the displacement from a reference position. Joint position transducers are ordinarily installed inside the manipulator shell, on the motor shaft, upstream of the transmission gears. For this reason, *de facto*, they sense the displacement of the shaft which sometimes might be quite different from the actual displacement of the joint.

2.1.5 Manipulator Anatomies

Manipulators meant to work in a 3D volume are typically made up of two clearly distinct fundamental pieces: the **arm**, providing mobility to the manipulator and the **wrist**, providing dexterity to the manipulator. According to the sequence of joint types of the arm, different families of manipulators can be defined:

- PPP joint sequence, with mutually perpendicular translation axes, gives rise to cartesian manipulators, suited for tasks defined using rectangular variables, since each degree of freedom of the arm corresponds to a rectangular coordinate of the 3D space;

- RPP joint sequence, with one translation axis parallel and the other translation axis perpendicular to the rotation axis, gives rise to cylindrical manipulators, suited for tasks defined using cylindrical variables, since each degree of freedom of the arm corresponds to a cylindrical coordinate of the 3D space;

- RRP joint sequence, with the second rotation axis perpendicular to both the first rotation axis and the translation axis, gives rise to spherical manipulators, suited for tasks defined using spherical variables, since each degree of freedom of the arm corresponds to a spherical coordinate of the 3D space;

- RRR joint sequence, with rotation axes not all parallel to each other, gives rise to articulated manipulators, suited for tasks defined in different or variable ways, since there is no correspondence between degrees of freedom of the arm and any kind of coordinates of the 3D space.

The same RRP joint sequence of spherical manipulators, but with all rotation and translation axes parallel to each other, gives rise to a further architecture, called SCARA, acronym of Selective Compliance Assembly Robot Arm: the mechanical arrangement of such manipulators is compliant in the horizontal direction and stiff in the vertical direction, making them suitable for precise and fast vertical assembling processes.

The most common industrial robots are undoubtedly articulated manipulators. Their arm is characterised by three (or more) consecutive revolute joints; typically, the rotation axes of the second joint and the third joint are parallel to each other and perpendicular to the rotation axis of the first joint. Articulated manipulators with such a standard framework are sometimes referred also as anthropomorphic manipulators, owing to their resemblance with a human upper limb: along these lines, the first three joints are respectively called **waist**, **shoulder** and **elbow**, while the first three links are respectively called **trunk**, **upper arm** and **forearm**; the remaining three joints make up the **wrist** and the end interface is called **hand**.

The *peculiar* morphology of articulated manipulators provides them with a rather good level of **dexterity** and **mobility**, making them suitable for a full range of applications, though at the price of losing the correspondence between joints degrees of freedom and Cartesian space coordinates, with the result that the position and the orientation of the hand are quite complex functions of the joint displacements.

Serial robot manipulators may be schematically represented through very simple geometric elements. For instance, links are often depicted as beams or rods, regardless of their actual shape and volume, while prismatic joints and revolute joints are often depicted as prisms and cylinders, respectively. The simplified schematics of a standard articulated manipulator is shown in Figure 2.1.

Arm

The kinematic structure of the arm described above is the same for all articulated manipulators. Since the hand must be able to move with respect to the base, then one end of the arm is constrained to a fixed framework by means of a mounting surface.

The kinematic and dynamic performances of a manipulator *substantially* depend on mechanical characteristics of the arm: the quickness of its joints conditions the maximum speed, the size of the its links is responsible for the overall reach, the strength of both its joints and links determines the maximum payload. Articulated manipulators are supposed to span a wide area, bear a significant weight and move pretty fast, hence their arm is customarily rather large, massive and bulky.

Wrist

The peculiar kinematic structure of articulated manipulators proves especially convenient because it decouples the position and the orientation of the end effector: the three degrees of freedom of the arm serve to position the wrist while the three degrees of freedom of the wrist serve to orient the hand.

For the sake of honesty, such a statement is not completely true and is considered factually valid only when the three rotation axes of the wrist joints do intersect in a single point, giving rise to the so called spherical wrist. Manipulators with non spherical wrists are obviously easier to design and build, although they cannot provide the decoupling between position and orientation, that results in a much more complicated kinematic control of the hand.

Nevertheless, even though the six degrees of freedom of articulated manipulators cannot be precisely split into two separate groups, the arm is *mainly* entrusted with positioning the hand and the wrist is *mainly* entrusted with orienting the hand. This is the reason why the dimensions of the wrist of a manipulator are normally far more compact than those of the arm of the same manipulator.

Hand

The ultimate purpose of the whole manipulator kinematic structure is the motion of its hand in the 3D space; the hand indeed carries a **tool**, which is a characteristic device designed to perform a certain operation within the work-cell. The tool installed on the hand is always application specific, therefore a given tool is the *distinctive* element of a manipulator, in the sense that it identifies uniquely the function of that manipulator, i.e. the way in which it is expected to interact with the surrounding environment. Sometimes the tool itself is erroneously referred as **end effector** as well.

Most common tools include: handling grippers, welding torches, grinding wheels, milling cutters, drilling bits, milling spindles, painting spray guns, suction cups, grasping claws, screw drivers, laser and water jet cutters, glue dispensers, blow torches, flame throwers, touch-trigger probes, laser scanners, collision detectors and so on. When a manipulator is required to handle more than one tool, a suitable tool changer has to be employed: in order to be interchangeable, the different tools must share the *same* mechanical interface with the mounting plate on the flange.

Some tools need to be energised, e.g., by means of electric, hydraulic or pneumatic actuators, therefore the manipulator must be capable of transmitting the corresponding

power through its entire kinematic chain, from the base up to the hand, either internally, i.e. housing the cables within the cover when they are thin enough, or externally, i.e. fixing the cables to the outer shell through guides. When the tool can also move with respect to the hand, e.g., if it is extensible/retractable, the *overall* kinematic chain has more than six degrees of freedom.

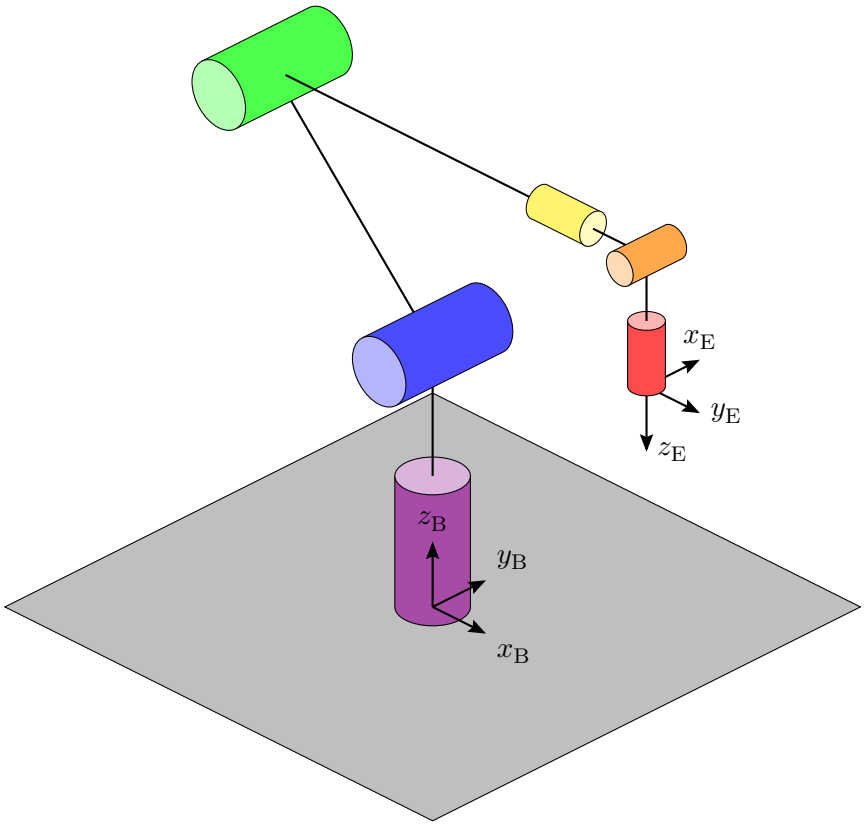

Figure 2.1: Stylised graphical representation of a standard articulated manipulator.

2.1.6 Work-space

A manipulator is capable of performing a certain task by moving its end effector, that holds the tool, in the three dimensional space surrounding it, therefore, in order to operate correctly the manipulator, it is essential to identify clearly the space in which it is *effectively* able to work.

The **work-space** of a manipulator can be defined as that region of the 3D space the manipulator is capable of placing its end effector within, or, alternatively, as that region of the 3D space spanned by the end effector as the manipulator joints perform every admissible motion. When Cartesian coordinates are used, since the displacements provided by the joints are neither unlimited nor discontinuous, then the work-space is a

bounded and connected subset of $\mathbb{R}^3$.

The characteristics of the work-space are determined by the geometry of all possible configurations of the manipulator: in particular, its volume mainly depends on the size of the links and the travel of the joints, whereas its shape is mainly comes from the sequence of joint types. To make some blatant examples, the work-space of a cartesian manipulator is a rectangular sector, the work-space of a cylindrical manipulator is a cylindrical sector, the work-space of a spherical manipulator is a spherical sector, and so on. The work-space of an articulated manipulator is far more complicated: it is normally a portion, with spherical symmetry, of the volume between two spheres with different radii; the radius of the outer sphere is ruled by the reach, i.e. the longest distance the manipulator end effector is capable of arrive at, approximately equal to the length of the fully extended arm, whereas the radius of the inner sphere is determined by the physical constraints between the links.

There are some particular portions of the working volume that are not nimbly reached by the manipulator, and, as a consequence, the orientation of the end effector may prove tough to control in those regions. For this reason, it is then customary to distinguish between different subsets of the manipulator surrounding volume, according to the attained level of dexterity. The locus of points in the 3D Cartesian space the end effector manages to reach with some orientation is referred manipulator primary or reachable work-space, whereas the locus of points in the 3D Cartesian space the end effector manages to reach with any orientation is referred as secondary or dexterous work-space. The dexterous work-space is clearly a subset of the reachable work-space, which is typically termed thus simply work-space.

2.2 Kinematic Spaces

Manipulator kinematics generally refers to the motion of the system of bodies making up the manipulator without explicitly discerning between the motion of the joints and the motion of the tool. In this regard, the same motion of the kinematic chain as a whole, required by a given task, may be analytically represented focusing the attention on either the coordinates of the joints or the pose of the end effector, thus two distinct but interconnected topological layers may be defined, namely joint space and task space.

2.2.1 Joint space

The motion of each joint can be analytically described by means of a *single* kinematic variable, therefore joints kinematics is usually studied assigning an univocal coordinate to each joint displacement: clearly, if the joint is prismatic, the joint coordinate is a linear position along a line, whereas, if the joint is revolute, the joint coordinate is an angular position about a line.

The set of all permissible joint coordinates is referred as manipulator **joint space**; its dimension is equal to the number of joints in the manipulator. The i-th joint coordinate, that is to say the real variable describing the motion of the i-th joint, is denoted by

$q_i \in \mathbb{R}$. Such a notation naturally suggests to define, for a a manipulator with n joints, the n-dimensional array

$$\boldsymbol{q} \coloneqq (q_1, \ldots, q_n) \in \mathbb{R}^n$$

having the n joints coordinates as its entries. Every real joint is capable of providing only a finite displacement, either linear or angular, within a given range, called joint stroke, therefore the joint space is a bounded subset of $\mathbb{R}^n$.

The set of all admissible links kinematic configurations[1] is referred as manipulator configuration space; its dimension is equal to the number of independent motions in the kinematic chain, which is also called degree of actuation, degree of motion, degree of mobility or more simply mobility.

The configuration variables can be transformed into a subset of the joint coordinates, therefore, in general, the degree of actuation is bounded by the number of joints and the two may not coincide. Nevertheless, mechanical linkages are usually endowed only with the indispensable number of mechanisms, because they are complex and expensive components, thus manipulators normally have *independent* joints only, otherwise there would be two or more joints acting as one, resulting in an unjustified waste: since different values of joint coordinates yield different link kinematic configurations and the degree of actuation is equal to the number of joints, within the framework of industrial robotics, configuration space and joint space are often used interchangeably as synonyms.

2.2.2 Task space

From the mechanical point of view, a manipulator is a more or less complex system of rigid bodies, each of which has its own *specific* motion, though only the motion of the end effector, which is indeed a rigid body itself, is normally subject of interest. A body in 3D space has, in principle, six degrees of freedom, resulting from the combination between the three position and the three orientation degrees of freedom; when some constraints are applied to the rigid body, though, the available degrees of freedom may decrease, thus, more in general, a body in 3D space has up to six degrees of freedom.

The set of all end effector poses permissible by a given task, is referred as manipulator **task space** or **operational space**; its dimension is equal to the task degrees of freedom, i.e. the degrees of freedom of the end effector demanded by that task: it depends on the constraints set by that task and, in any case, can be six at most.

From an abstract point of view, the pose of a rigid body is the collection of the position and the orientation of that body; it is always a well defined mathematical entity, regardless of the actual representation of the orientation. The end effector position $\boldsymbol{r} \in \mathbb{R}^3$ is a point in the 3D Cartesian space, while the end effector orientation $\boldsymbol{\phi} \in \mathrm{SO}(3)$ is an element of the Special Orthogonal Group of degree 3; the pose of the end effector, i.e. the juxtaposition of the position and the orientation of the end effector, belongs to

[1] In rigid body mechanics, the **configuration** of a rigid body or a system of rigid bodies, is any collection of variables with minimum cardinality, called generalised coordinates, able to describe completely the position of *any* point of the rigid body or the system of rigid bodies.

the to the Special Euclidean group of degree 3

$$\boldsymbol{p} \in \mathrm{SE}(3)$$

where $\mathrm{SE}(3) \coloneqq \mathbb{R}^3 \otimes \mathrm{SO}(3)$. Also the orientation is sometimes described by a triple of numbers, i.e. three Euler angles; for instance, when the roll-pitch-yaw angles are adopted, the position and the orientation may be respectively described by $\boldsymbol{r} = (x, y, z)$ and $\boldsymbol{\phi} = (\alpha, \beta, \gamma)$, even though only the position is a proper vector, whereas the orientation is just an array of numbers. That is the reason why, with a slight abuse of notation, the pose is sometimes considered a vector of $\mathbb{R}^6$. Since the task may limit the motion of the end effector, then, in general,

$$\boldsymbol{p} \in \mathbb{R}^m$$

where $m \leq 6$ is the number of task degrees of freedom.

Even though the concepts may sound quite similar, task space and work-space are not synonyms: the former is the unlimited mathematical space a certain task is formally represented in whereas the latter is a circumscribed physical region of the work-cell the end effector may move within. Nevertheless, they might also be described according to a common analytical framework: in this case, a given task may be performed by a manipulator only if the intersection between the (position) task space and the work-space is non empty and the dimension of the (pose) task space is not larger than the manipulator degrees of freedom.

2.2.3 Degrees of Freedom and Redundancy

Depending on the relationship between the number of joints and the number of degrees of freedom demanded by a given task, three situations lie ahead. When $n < m$, the manipulator is unsuitable for that task, therefore the task requires a minimum number of joints, i.e. $n \geq m$: when $n > m$, the manipulator is said to be **redundant** for that task, whereas, when $n = m$, the manipulator is said to be **non redundant** for that task. The difference $n - m$ measures the level of redundancy and is thus termed redundancy degree or degree of redundancy.

Redundancy means that not all the joints are *actually* required to achieve the desired degrees of freedom, and that the same end effector pose might be attained by means of several combinations of joint coordinates, dramatically improving the manipulator dexterity over the space. Redundant manipulators are extremely hard to control but they may prove especially useful for applications demanding high levels of agility.

Sometimes the manipulator is redundant in relation to a particular task, that is when that task does not need all manipulator degrees of freedom. This situation is not that unlikely, in fact it often occurs, for instance, when the end effector of a manipulator designed to operate *routinely* in the space, is constrained on portions of planes (e.g., a panel or a wall) or lines (e.g., a track or a groove). For this reason, the same manipulator may happen to be redundant for a given task, but non-redundant for another task. There exist indeed two different types of kinematic redundancy: when the degree of motion is larger than the task degree of freedom, the manipulator is said to be functionally

redundant, whereas, when the degree of motion is larger than the manipulator degree of freedom, the manipulator is said to be intrinsically redundant.

2.3 Kinematic Model

2.3.1 Definition and Motivation

Kinematics is the branch of mechanics that studies the motion of bodies without investigating the causes that generate it. The only physical quantities involved in kinematics are usually linear and angular positions and their derivatives, such as velocities, accelerations, jerks, and so on. Kinematics applied to manipulators is called manipulator kinematics: it studies the motion of the manipulator kinematic chain as a whole, focusing its attention mainly on the motion of the joints and the motion of the end effector, along with their relationship.

In order to fulfil a given task assigned in the framework of a certain application, the end effector is required to carry out a designated motion in the work-space. Considering that a manipulator is operated only by driving its joints servos, the control system must be able to relate the motion of the end effector to the motion of the joints, in order to command properly the manipulator movements, that is to say *consistently* with the desired kinematic behaviour. It immediately follows that the development of the manipulator **kinematic model**, i.e. the mathematical relationship between the end effector kinematic quantities and the joints kinematic quantities, is a problem of *paramount* importance in robot control.

2.3.2 Kinematic Notation

As just mentioned, manipulator kinematics essentially investigates the motion of the end effector in relation to the motion of the joints, therefore it is necessary to represent somehow the pose of the end effector with respect to a privileged reference frame, which customarily sticks to some fixtures, with easily identifiable geometry, in the space surrounding the manipulator, such as floors, roofs, walls, objects with regular shapes or the manipulator base itself.

A rigid body can be represented kinematically through a reference frame rigidly attached to that body. The kinematics of a manipulator may be thus investigated describing a reference frame tied to the manipulator end effector, with respect to a reference frame tied to the manipulator base. The *fixed* reference frame attached to the base is referred as **base frame** $\mathcal{R}_{\mathrm{B}}$ or **world frame** $\mathcal{R}_{\mathrm{W}}$ and is habitually selected according to the geometry of the environment around the manipulator. The *mobile* reference frame attached to the end effector is referred as **end frame** $\mathcal{R}_{\mathrm{E}}$ or **tool frame** $\mathcal{R}_{\mathrm{T}}$ and is habitually selected according to the geometry of the tool held by the manipulator. A popular choice of these reference frames for articulated manipulators is shown in Figure 2.1.

Exploiting homogeneous coordinates notation, the pose of the end effector with re-

spect to the base is described through the homogeneous transformation matrix

$$\boldsymbol{T}_{\mathrm{E}}^{\mathrm{B}} = \begin{pmatrix} \boldsymbol{R}_{\mathrm{E}}^{\mathrm{B}} & \boldsymbol{t}_{\mathrm{E}}^{\mathrm{B}} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} \tag{2.1}$$

where $\boldsymbol{R}_{\mathrm{E}}^{\mathrm{B}}$ is the rotation matrix giving the orientation of the end effector with respect to the base and $\boldsymbol{t}_{\mathrm{E}}^{\mathrm{B}}$ is the translation vector giving the position of the end effector with respect to the base.

2.3.3 Link Frames

When developing the kinematics of manipulators with many joints, such as articulated manipulators, a *direct* approach, that is the immediate computation, by means of plain geometric inspection, of the kinematic relationship between the world frame and the tool frame is practically infeasible, and an *indirect* approach, based on several relative transformations between reference frames deployed along the kinematic chain from the base up to the flange, is definitely preferable. In this light, the *overall* kinematic transformation between the world frame and the tool frame may be imagined as the cascade of more transformations between pairs of reference frames coming from a sequence that starts with the world frame and ends with the tool frame.

Since the links are proper rigid bodies with their distinctive motion, then it makes sense to assign a different reference frame to *each* link. This choice is especially appropriate because the relationship between reference frames attached to adjacent links may be described by means of fixed quantities except for a single joint variable. The number of interposing reference frames along the kinematic chain of the manipulator is thus not arbitrary, in fact it is imposed by the number of links. The reference frames the manipulator links have to be tied to are referred as **link frames**.

A n-DOF serial manipulator is composed of n joints and $n+1$ links. The links are labelled according to their number: conventionally, the first one is called link 0 while the last one is called link n, then the i-th joint connects the $(i-1)$-th link and the i-th link. The relation between two consecutive link frames $\mathcal{R}_i$ and $\mathcal{R}_{i-1}$ is thus affected by the i-th joint motion only and is described by the homogeneous transformation matrix

$$\boldsymbol{T}_{i}^{i-1} = \begin{pmatrix} \boldsymbol{R}_{i}^{i-1} & \boldsymbol{t}_{i}^{i-1} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix}$$

as $\boldsymbol{R}_{i}^{i-1}$ describes the orientation of $\mathcal{R}_i$ with respect to $\mathcal{R}_{i-1}$, whereas $\boldsymbol{t}_{i}^{i-1}$ describes the position of $\mathcal{R}_i$ with respect to $\mathcal{R}_{i-1}$.

The homogeneous transformation matrix describing the total relationship between the 0^{th} link frame and the n^{th} link frame is the product of the homogeneous transformation matrices relative to all partial relationships between subsequent link frames

$$\boldsymbol{T}_{n}^{0} = \boldsymbol{T}_{1}^{0}\boldsymbol{T}_{2}^{1}\cdots\boldsymbol{T}_{n-1}^{n-2}\boldsymbol{T}_{n}^{n-1} = \prod_{i=1}^{n}\boldsymbol{T}_{i}^{i-1}$$

and the homogeneous transformation matrix $\boldsymbol{T}_{\mathrm{E}}^{\mathrm{B}}$ describing the complete relationship between base frame and end frame is

$$\boldsymbol{T}_{\mathrm{E}}^{\mathrm{B}} = \boldsymbol{T}_{0}^{\mathrm{B}}\boldsymbol{T}_{n}^{0}\boldsymbol{T}_{\mathrm{E}}^{n} = \boldsymbol{T}_{0}^{\mathrm{B}}\boldsymbol{T}_{1}^{0}\cdots\boldsymbol{T}_{n}^{n-1}\boldsymbol{T}_{\mathrm{E}}^{n}$$

where $\boldsymbol{T}_{0}^{\mathrm{B}}$ describes the further relationship between the base frame and the first link frame, while $\boldsymbol{T}_{\mathrm{E}}^{n}$ describes the further relationship between the last link frame and the end frame, whenever these frames are not coincident.

The relationship between adjacent links come out once the reference frames are clearly identified over the manipulator kinematic chain. A systematic procedure is therefore needed to determine precisely how to define the link frames. There exist several guidelines and directives devised to address this matter, commonly gathered into different collections of rules, better known as **kinematic conventions**, among which the most widespread and renowned is undoubtedly the Denavit-Hartenberg convention.

2.3.4 Denavit-Hartenberg Convention

Coordinate Frames Selection

In principle, the frame representing the position and the orientation of a link might be *arbitrarily* set, that is to say choosing whatever origin and axes, provided that it moves along with the link. Up to six degrees of freedom are associated to a generic reference frame transformation, thus six parameters would be required to describe it. Nonetheless, it is possible to define some reasonable rules to constrain link frames in order to reduce the number of parameters required to describe the relative transformations.

Specifically, if one of the three coordinate axes of the link frame is directed along the spatial line defining the motion, either linear or rotary, of a joint, then a single elemental transformation with respect to that axis would suffice to represent the joint displacement: several conventions intended to model manipulator kinematics set the z axis of link frames following this argument indeed. According to the kinematic convention proposed by Denavit and Hartenberg (see [9]), the link frames are assigned following some precise criteria based on the geometric concepts of **motion axis** and **common normal**.

Geometric Tools

The 3D oriented line a revolute joint rotates about or the 3D oriented line a prismatic joint slides along is generically referred as **joint motion axis**. The notion of joint motion axis surely represents the milestone of Denavit-Hartenberg convention.

The **common normal** between the motion axes of two consecutive joints is defined as the segment with minimum distance connecting those axes, in a narrow sense, or the line containing such a segment, in a broad sense. By definition, the common normal between two motion axes is orthogonal to *both* of them, as shown in fig. 2.2. The notion of common normal is another key element of Denavit-Hartenberg convention.

This abstract concept gives a quite good insight of the spatial linkages geometry although it reveals its limits in two particular cases, namely axes intersection and alignment: when two motion axes converge in a single point, the common normal segment

degenerates into that point, therefore the common normal line is undefined, because there exist *infinite* lines containing that point; when two motion axes have the same orientation, the common normal segment is undefined, because there exist *infinite* parallel segments orthogonal to both axes.

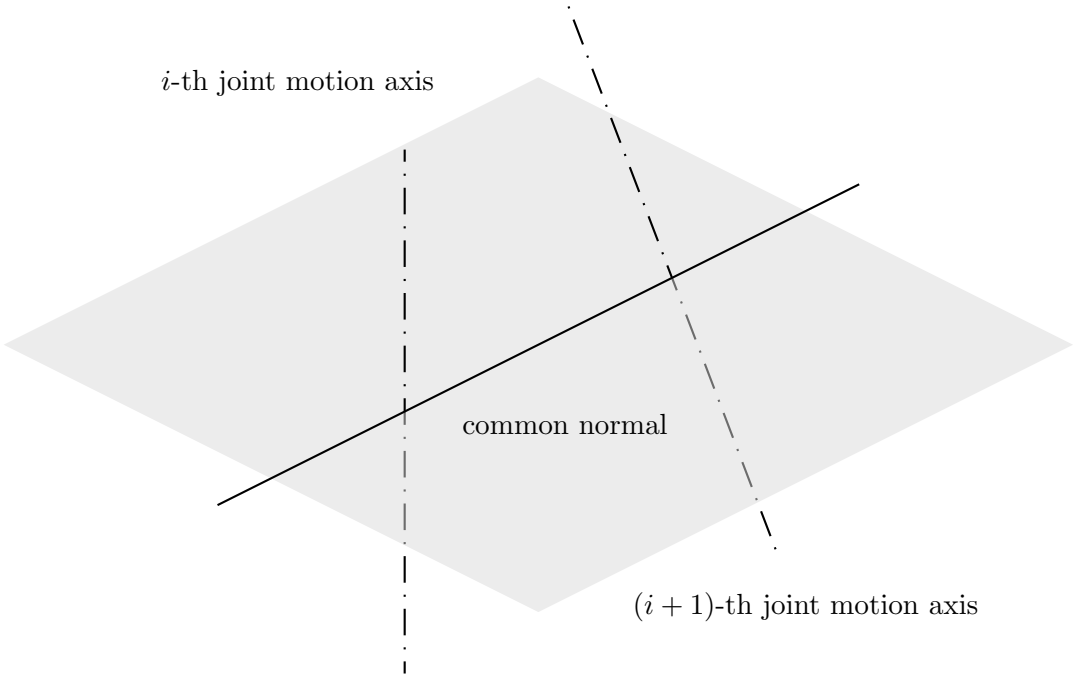

Figure 2.2: Common normal between consecutive joint motion axes.

Systematic Procedure

The Denavit-Hartenberg kinematic convention proposes a systematic procedure with the rules for defining the relation between two adjacent links frames. The origin position and the coordinate axes orientation of $\mathcal{R}_i$ with respect to $\mathcal{R}_{i-1}$ are set after detecting the couple of joint motion axes and identifying the corresponding common normal:

- the origin of $\mathcal{R}_i$ must lie on the intersection between the common normal and the $(i+1)$-th joint motion axis;
- the z axis of $\mathcal{R}_i$ must be parallel to the $(i+1)$-th joint motion axis, following the positive direction of motion;
- the x axis of $\mathcal{R}_i$ must be parallel to the common normal, following the direction from the i-th joint motion axis to the $(i+1)$-th joint motion axis;
- the y axis of $\mathcal{R}_i$ must be orthogonal to the common normal and the $(i+1)$-th joint motion axis, following the direction that yields a right-handed **frame**.

The $(i-1)$-th i link frame is therefore transformed into the i-th link frame, performing first a translation along the z axis in order to reach the intersection between the i-th joint motion axis and the common normal, and a rotation about the z axis in order to orient the x axis with the common normal, then a translation along the x axis in order the reach the intersection between the common normal and the $i+1$-th joint motion axis, and a rotation about the x axis in order to orient the z axis with the $(i+1)$-th joint motion axis.

Symbolic Notation

Four elemental transformations, two translations and two rotations, are needed to bring the link frame $\mathcal{R}_{i-1}$ to overlap with the link frame $\mathcal{R}_i$; four quantities, two lengths and two angles, are thus required to specify either symbolically or numerically the complete transformation: they are generally referred as Denavit-Hartenberg parameters, or, sometimes, simply as kinematic parameters. The i-th DH parameters, i.e. the DH parameters related to the i-th transformation, are called $d_i, \theta_i, a_i, \alpha_i$. In particular:

- d_i is the translation distance along the i-th joint motion axis;
- θ_i is the rotation angle about the i-th joint motion axis;
- a_i is the translation distance along the i-th common normal;
- α_i is the rotation angle about the i-th common normal.

The i-th kinematic parameters and the corresponding transformations that relate $\mathcal{R}_{i-1}$ and $\mathcal{R}_i$ are graphically portrayed in fig. 2.3.

The first two elemental transformations are defined with respect to the joint motion axis, therefore they might integrate directly the motion of the joint, that is to say one of the first two kinematic parameters d_i and θ_i might represent the i-th joint coordinate, up to a constant offset: more specifically, if the joint is prismatic, d_i accounts for the i-th joint linear displacement, whereas, if the joint is revolute, θ_i accounts for the i-th joint angular displacement. The other kinematic parameters not related to the joint displacement must be inferred from the geometry of the link connected to the pair of joints under consideration. In summary, among the Denavit-Hartenberg parameters, there are three constants that depend *only* on the geometry of the i-th link, and one variable that depends *also* on the motion of the i-th joint.

The homogeneous transformation matrix describing the relationship between $\mathcal{R}_{i-1}$ and $\mathcal{R}_i$ according to the Denavit-Hartenberg convention is constructed as

$$\boldsymbol{T}_i^{i-1} \coloneqq \mathbf{Trans}\left(\boldsymbol{k}, d_i\right) \mathbf{Rot}\left(\boldsymbol{k}, \theta_i\right) \mathbf{Trans}\left(\boldsymbol{i}, a_i\right) \mathbf{Rot}\left(\boldsymbol{i}, \alpha_i\right)$$

multiplying the homogeneous transformation matrices corresponding to the four elemental transformations parametrised by the kinematic parameters, observing the order

defined above. The complete homogeneous transformation matrix is

$$\boldsymbol{T}_i^{i-1} = \begin{pmatrix} \boldsymbol{R}_i^{i-1} & \boldsymbol{t}_i^{i-1} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} = \begin{pmatrix} \mathrm{c}_{\theta_i} & -\mathrm{s}_{\theta_i}\mathrm{c}_{\alpha_i} & \mathrm{s}_{\theta_i}\mathrm{s}_{\alpha_i} & a_i\mathrm{c}_{\theta_i} \\ \mathrm{s}_{\theta_i} & \mathrm{c}_{\theta_i}\mathrm{c}_{\alpha_i} & -\mathrm{c}_{\theta_i}\mathrm{s}_{\alpha_i} & a_i\mathrm{s}_{\theta_i} \\ 0 & \mathrm{s}_{\alpha_i} & \mathrm{c}_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

therefore the orientation of $\mathcal{R}_i$ in $\mathcal{R}_{i-1}$ depends only on the angles θ_i and α_i, while the position of $\mathcal{R}_i$ in $\mathcal{R}_{i-1}$ depends on both the lenghts d_i and a_i and the angle θ_i. depends on *both* lengths and angle parameters, but not on the angle of the last rotation.

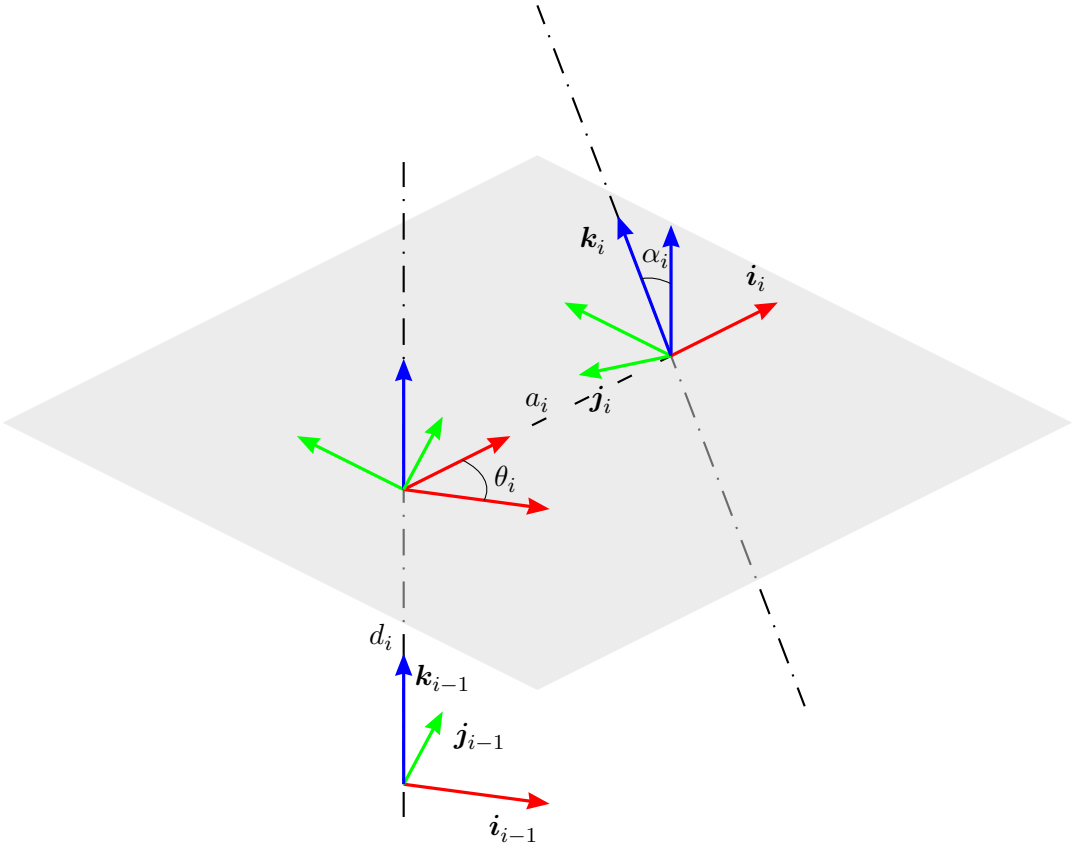

Figure 2.3: Denavit-Hartenberg transformation between the $(i-1)$-th and the i-th link frames.

Ambiguity

As already anticipated in the introductory discussion about the notion of common normal, there exist two particular geometric configurations that give rise to ambiguity when following the rules of Denavit-Hartenberg convention given before.

When the two consecutive joint motion axes are parallel, the common normal is not unique. The origin $\boldsymbol{o}_i$ of the i-th link frame might be thus *arbitrarily* placed along the $(i+1)$-th joint motion axis; for the sake of simplicity, it is customarily located at the point of intersection between the i-th joint motion axis and the common normal line that passes through the origin $\boldsymbol{o}_{i-1}$ of the $(i-1)$-th link frame too, in order to yield no offset along the joint motion axis, i.e. $d_i = 0$. On the contrary, the orientation of the x axis $\boldsymbol{i}_i$ of the i-th link frame is still *distinctly* defined, because every common normal line shares the same spatial direction. Such an eventuality usually occurs for instance in articulated manipulators between the motion axes of the 2^{nd} and the 3^{rd} joints.

When two consecutive joint motion axes are incident, the common normal does not exist. The x axis $\boldsymbol{i}_i$ of the i-th link frame cannot be thus directed by following the aforementioned criteria, and, in principle, it might be *arbitrarily* oriented on the plane orthogonal to the $(i+1)$-th joint motion axis; nevertheless, bearing in mind that the common normal line has to be perpendicular to both joint motion axes, it is conventionally oriented by setting the condition $\boldsymbol{i}_i = \boldsymbol{k}_{i-1} \times \boldsymbol{k}_i$. Such a constraint could be universally used in place of the classical one based on the common normal, as it would return the same result except for the sense, the cross product being not commutative, if not for parallel joint motion axes, in which case it fails of course. In contrast, the origin $\boldsymbol{o}_i$ of the i-th link frame is still *uniquely* defined, because it coincides with the point of intersection. Such a circumstance usually occurs in articulated manipulators between the motion axes of the 4^{th} and the 5^{th} joints or between the motion axes of the 5^{th} and the 6^{th} joints.

Arbitrariness

The Denavit-Hartenberg convention sets the rules for transforming reference frames attached to subsequent links basing on the respective joint motion axes. For this reason, it cannot provide any useful hint whatsoever about the specification of the base frame and the end frame.

The base frame $\mathcal{R}_{\text{B}}$ normally corresponds to the first link frame, i.e. the reference frame $\mathcal{R}_0$ attached to the 0^{th} link. The z axis is the only element of $\mathcal{R}_0$ that can be unequivocally assigned, for it must coincide with the 1^{st} joint motion axis. The origin and the coordinate axes x and y of $\mathcal{R}_0$ are not constrained in any way by the rules of Denavit-Hartenberg convention, because there is no joint upstream of the first link, which is anchored to a fixed surface. The base frame might be then conveniently defined in compliance with the geometry of the manipulator base: for instance, the origin $\boldsymbol{o}_0$ is usually placed at the centre of the mounting surface, while the coordinate axes $\boldsymbol{i}_0$, $\boldsymbol{j}_0$ and $\boldsymbol{k}_0$ are usually oriented respectively with the frontal direction, the lateral direction and the vertical direction.

The end frame $\mathcal{R}_{\text{E}}$ normally corresponds to the last link frame, i.e. the reference frame $\mathcal{R}_n$ attached to the n^{th} link. No element of $\mathcal{R}_n$ can be unequivocally assigned, because there is no joint downstream of the n^{th} link, which is firmly attached to the tool. The end frame $\mathcal{R}_n$ might be then conveniently defined in compliance with the geometry of the manipulator end effector: for instance, the origin $\boldsymbol{o}_n$ is usually placed

at the centre of the equipped tool, while the coordinate axes, $\boldsymbol{i}_n$, $\boldsymbol{j}_n$ and $\boldsymbol{k}_n$ are usually oriented respectively with the normal direction, the sliding direction and the approach direction.[2] In any case, $\boldsymbol{i}_{n-1}$ and $\boldsymbol{k}_n$ must be orthogonal, owing to the specific sequence of transformations of the Denavit-Hartenberg convention.

2.3.5 Direct Kinematics

Manipulator **direct kinematics** or **forward kinematics** studies the *direct* relationship between the motion of the joints and the motion of the tool. Specifically, given certain joint coordinates $\boldsymbol{q} \in \mathbb{R}^n$ in the joint space, direct kinematics seeks to find the value of the end effector pose $\boldsymbol{p} \in \mathbb{R}^m$ in the task space that are given by that coordinates. The function $\boldsymbol{f}_k \colon \mathbb{R}^n \to \mathbb{R}^m$, defined by the relation

$$\boldsymbol{p} = \boldsymbol{f}_k(\boldsymbol{q}) \tag{2.2}$$

is termed **direct kinematic function**.

The practical task of finding a mathematical relationship relating the joint coordinates to the corresponding tool poses is usually quite easy; when the problem is feasible, a solution always exists and is unique: as long as the joint coordinates lie within the joint space, that is to say all the joints do not exceed their admissible strokes, there exists necessarily only one end effector pose generated by that joint coordinates. For this reason, direct kinematics is always a well-posed[3] problem, provided that the joint coordinates are picked inside the manipulator joint space.

The direct kinematic function of a manipulator can be derived analytically by exploiting all the information already available about the kinematic model of that manipulator. For instance, if the Denavit Hartenberg convention is adopted to model the kinematic relationships between pairs of subsequent links frames, as already mentioned, the ordered product of all the corresponding homogeneous transformation matrices may already give a compact description of the end effector position and orientation with respect to the base, as a function of the joint coordinates $\boldsymbol{q}$, which can be written as

$$\boldsymbol{T}_n^0(\boldsymbol{q}) = \prod_{i=1}^{n} \boldsymbol{T}_i^{i-1}(q_i) = \boldsymbol{T}_1^0(q_1)\, \boldsymbol{T}_2^1(q_2) \cdots \boldsymbol{T}_{n-1}^{n-2}(q_{n-1})\, \boldsymbol{T}_n^{n-1}(q_n) \tag{2.3}$$

as the matrix $\boldsymbol{T}_i^{i-1}$ is a function of the i-th joint coordinate q_i only.

The position and the orientation of the end effector can be extracted separately, writing the homogeneous transformation matrix $\boldsymbol{T}_n^0$ as in eq. (2.1): specifically, the orientation of the end effector is represented by orientation matrix $\boldsymbol{R}_n^0$, computed as

$$\boldsymbol{R}_n^0 = \boldsymbol{R}_1^0 \cdots \boldsymbol{R}_n^{n-1} \tag{2.4}$$

[2] The normal direction is the direction orthogonal to the hand working plane, the sliding or orientation direction is the direction along which the hand can slide, the approach direction is the direction along which the hand can approach an object.

[3] According to the definition given by Jacques Hadamard, a problem is said to be well posed, whenever a solution to the problem does exist, is unique and changes with continuity with the conditions.

because the rotation block of the product of a sequence of homogeneous transformation matrices is the product of the rotation blocks of the sequence of homogeneous transformation matrices; whereas the position of the end effector is represented by the position vector $\boldsymbol{t}_n^0$, computed recursively as

$$\boldsymbol{t}_n^i = \boldsymbol{t}_{i+1}^i + \boldsymbol{R}_{i+1}^i \boldsymbol{t}_n^{i+1} \tag{2.5}$$

with the straightforward starting condition $\boldsymbol{t}_n^n = \boldsymbol{0}$. The direct kinematic function immediately comes out simply by making explicit the dependence on the joint coordinates in the orientation matrix and the position vector just computed, bearing in mind that the coordinates of prismatic joints may affect *only* the position, while the coordinates of revolute joints may affect *both* the orientation and the orientation.

If the i-th joint is revolute, then the joint coordinate q_i is an angle that modifies the rotation around the i-th joint motion axis, therefore both $\boldsymbol{R}_i^{i-1}$ and $\boldsymbol{t}_i^{i-1}$ are functions of that angle, whereas, if the i-th joint is prismatic, then the joint coordinate q_i is a distance that modifies the translation along the i-th joint motion axis, hence only $\boldsymbol{t}_i^{i-1}$ is a function of that distance while $\boldsymbol{R}_i^{i-1}$ is a constant matrix. Since 6-DOF articulated manipulators have six revolute joints, then the orientation matrix of end effector, written out in full as

$$\boldsymbol{R}_6^0 = \boldsymbol{R}_1^0 \boldsymbol{R}_2^1 \boldsymbol{R}_3^2 \boldsymbol{R}_4^3 \boldsymbol{R}_5^4 \boldsymbol{R}_6^5$$

is a function of all six joint coordinates, whereas the position vector of the end effector, written out in full as

$$\boldsymbol{t}_6^0 = \boldsymbol{t}_1^0 + \boldsymbol{R}_1^0 \left(\boldsymbol{t}_2^1 + \boldsymbol{R}_2^1 \left(\boldsymbol{t}_3^2 + \boldsymbol{R}_3^2 \left(\boldsymbol{t}_4^3 + \boldsymbol{R}_4^3 \left(\boldsymbol{t}_5^4 + \boldsymbol{R}_5^4 \boldsymbol{t}_6^5 \right) \right) \right) \right)$$

is a function of the first five joint coordinates.

Relying on the simplified geometric elements of the nominal manipulator design in order to reduce the complexity the direct kinematic function, albeit rather tempting, frequently proves to be of very little practical help, because the ideal geometry of the manipulator design is normally pretty far from the actual geometry of a manipulator unit. The manipulator user might indeed desire to give a more accurate mathematical description of the manipulator kinematics, by identifying and trimming the values of the kinematic parameters and then updating the kinematic model with those values, in such a way the predicted behaviour could match as closely as possible the actual behaviour. However, such a correction may be carried out only whether those parameters, the value of which is subject to modification, are *explicitly* included into the analytical expression of the kinematic function, which is not the case with the nominal model, where some parameters are completely wiped out for the sake of simplification.

2.3.6 Inverse Kinematics

Manipulator **inverse kinematics** or **backward kinematics** studies the inverse relationship between the motion of the joints and the motion of the tool. Specifically, given a certain end effector pose $\boldsymbol{p} \in \mathbb{R}^m$ in the task space, inverse kinematics seeks to find

the values of the joint coordinates $\boldsymbol{q} \in \mathbb{R}^n$ in the joint space that give that pose. The function $\boldsymbol{f}_k{}^{-1} : \mathbb{R}^m \to \mathbb{R}^n$, defined by the relation

$$\boldsymbol{q} = \boldsymbol{f}_k{}^{-1}(\boldsymbol{p}) \tag{2.6}$$

is termed **inverse kinematic function**.

The practical task of finding a mathematical function relating the tool pose to the corresponding joint coordinates is usually quite tough and may be addressed in several ways, e.g., by means of geometric inspection, numeric optimisation, iterative solution, symbolic engines, etc., according to the available resources, the desired precision and the specific degree of complexity.

The desired end effector position completely determines the feasibility of the inverse kinematics problem: as a matter of fact, if the position lies within the work-space, there exists necessarily at least one combination of joint coordinates that brings the end effector to that position, with a certain orientation. Moreover, if the position lies within the dexterous work-space, there exists even a combination of joint coordinates that brings the end effector to the desired position with the desired orientation. In the end, inverse kinematics is a well posed problem, provided that the input position is picked inside the manipulator work-space.

As regards articulated manipulators, and, in general, all other types of manipulators endowed with revolute joints, since the expression of the end effector position and orientation involves some trigonometric functions of the joints coordinates , then direct kinematics features neither linearity, that is to say joints displacements do not yield linearly proportional tool pose variations, nor injectivity, that is to say different joint postures might yield the same tool pose.

These issues give rise to some annoying consequences in the practical implementation of inverse kinematics: firstly, the solution depends on the local properties of the problem, hence the solution corresponding to some conditions, cannot be extended or updated when those conditions change; secondly, even though certain conditions can guarantee the existence of the solution, they cannot ensure the uniqueness of the solution, even for non-redundant manipulators, hence it is necessary to devise a suitable criterion to single out a specific solution among all the admissible ones. For all these reasons, an analytical approach to inverse kinematics is *virtually* infeasible.

Simple Manipulators

Nevertheless, most articulated manipulators, although fairly different in shapes and sizes, are designed according to a common framework based on simplified geometries involving exact perpendicularity and parallelism between subsequent joint motion axes. Such peculiar spatial features simplify the requested analytical effort, thus making the kinematics of articulated manipulators *relatively* easy to investigate.

Manipulators with such a standard simplified geometry have been labelled kinematically simple manipulators, or directly simple manipulators. The inverse kinematic function of a simple manipulator may be computed analytically and has a closed-form

expression, as discussed in [12]. A systematic procedure to solve inverse kinematics of simple articulated manipulators, is outlined hereinafter.

All quantities are represented in the base frame $\mathcal{R}_0$, therefore, in order to ease the notation, the superscript 0 is dropped for all vectors and matrices, without ambiguity; moreover, when only base frame is involved, the subscript 0 is avoided as well. The joint angles $\theta_1, \ldots, \theta_6$ that yield a desired hand pose $\boldsymbol{p}_\mathrm{H}$ are computed; the array with the six joint coordinates $\boldsymbol{q}$ immediately comes out from the joint angles just found, after subtracting the respective offset angles. The tool pose $\boldsymbol{p}_\mathrm{H}$ is split into its fundamental components: the hand position $\boldsymbol{r}_\mathrm{H}$ and the hand orientation $\boldsymbol{\phi}_\mathrm{H}$; the angles in the orientation vector can be used to build the orientation matrix $\boldsymbol{R}_\mathrm{H}$, according to the selected attitude representation.

Arm Coordinates

Simple manipulators have **spherical wrists**, that is to say the last three joints motion axes all intersect in a single point, thus $a_4, a_5, a_6 = 0$. The intersection point is located at the centre of the wrist therefore it is called wrist position and is denoted by $\boldsymbol{r}_\mathrm{W}$. The hand position could be obtained shifting the wrist position along the 6^{th} motion axis by the distance offset, i.e. $\boldsymbol{r}_\mathrm{H} = \boldsymbol{r}_\mathrm{W} + d_6\boldsymbol{k}_5$, hence the wrist position can be computed as

$$\boldsymbol{r}_\mathrm{W} = \boldsymbol{r}_\mathrm{H} - d_6\boldsymbol{R}_\mathrm{H}\boldsymbol{k}$$

where $\boldsymbol{r}_\mathrm{H}$ is the desired hand position vector, and $\boldsymbol{R}_\mathrm{H}$ is the desired hand orientation matrix. The rotated z axis $\boldsymbol{R}_\mathrm{H}\boldsymbol{k}$ represents the hand approach direction.

On the other hand, the wrist is constrained by the base through the kinematic chain consisting of the trunk, the upper arm and the forearm, so the position of the wrist can be also deduced from the sequence of DH translations along z and x axes of link frames 0, 1, 2 and 3

$$\begin{aligned}
\boldsymbol{r}_\mathrm{W} &= d_1\boldsymbol{k}_0 + a_1\boldsymbol{i}_1 + d_2\boldsymbol{k}_1 + a_2\boldsymbol{i}_2 + d_3\boldsymbol{k}_2 + a_3\boldsymbol{i}_3 + d_4\boldsymbol{k}_3 = \\
&= d_1\boldsymbol{k}_0 + a_1\boldsymbol{R}_z(\theta_1)\,\boldsymbol{i}_0 + d_2\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x(\alpha_1)\,\boldsymbol{k}_0 + a_2\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x(\alpha_1)\,\boldsymbol{R}_z(\theta_2)\,\boldsymbol{i}_0 + \\
&+ d_3\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x(\alpha_1)\,\boldsymbol{R}_z(\theta_2)\,\boldsymbol{R}_x(\alpha_2)\,\boldsymbol{k}_0 + a_3\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x(\alpha_1)\,\boldsymbol{R}_z(\theta_2)\,\boldsymbol{R}_x(\alpha_2)\,\boldsymbol{R}_z(\theta_3)\,\boldsymbol{i}_0 + \\
&+ d_4\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x(\alpha_1)\,\boldsymbol{R}_z(\theta_2)\,\boldsymbol{R}_x(\alpha_2)\,\boldsymbol{R}_z(\theta_3)\,\boldsymbol{R}_x(\alpha_3)\,\boldsymbol{k}_0
\end{aligned}$$

brought back in link frame 0. The above expression might be significantly simplified by considering the *peculiar* geometric features of simple manipulators: the motion axis of the shoulder and the motion axis of the waist are perpendicular, thus $\alpha_1 = \pm\frac{\pi}{2}$; the motion axis of the elbow and the motion axis of the shoulder are parallel, thus $d_2 = 0$ and $\alpha_2 = 0 \wedge \pi$; the first motion axis of the wrist and the motion axis of the elbow are perpendicular, thus $\alpha_3 = \pm\frac{\pi}{2}$.

The angles $\alpha_1 = -\frac{\pi}{2}$, $\alpha_2 = 0$, $\alpha_3 = +\frac{\pi}{2}$ are here assumed, because they reflect the geometric layout of the majority of articulated manipulators, though the reasoning remains still valid even if the other angles are used instead, for they would yield equivalent

results, only with different signs. Substituting the simplified DH parameters into the expression of the wrist position gives

$$
\begin{aligned}
\boldsymbol{r}_{\mathrm{W}} &= x_{\mathrm{W}}\boldsymbol{i} + y_{\mathrm{W}}\boldsymbol{j} + z_{\mathrm{W}}\boldsymbol{k} = \\
&= d_1\boldsymbol{k} + a_1\boldsymbol{R}_z(\theta_1)\,\boldsymbol{i} + a_2\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x\!\left(-\tfrac{\pi}{2}\right)\boldsymbol{R}_z(\theta_2)\,\boldsymbol{i} + d_3\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x\!\left(-\tfrac{\pi}{2}\right)\boldsymbol{R}_z(\theta_2)\,\boldsymbol{k} + \\
&+ a_3\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x\!\left(-\tfrac{\pi}{2}\right)\boldsymbol{R}_z(\theta_2+\theta_3)\,\boldsymbol{i} + d_4\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x\!\left(-\tfrac{\pi}{2}\right)\boldsymbol{R}_z(\theta_2+\theta_3)\,\boldsymbol{R}_x\!\left(+\tfrac{\pi}{2}\right)\boldsymbol{k} = \\
&= d_1\boldsymbol{k} + a_1\boldsymbol{R}_z(\theta_1)\,\boldsymbol{i} + a_2\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_y(\theta_2)\,\boldsymbol{R}_x\!\left(-\tfrac{\pi}{2}\right)\boldsymbol{i} + d_3\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_x\!\left(-\tfrac{\pi}{2}\right)\boldsymbol{k} + \\
&+ a_3\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_y(\theta_2+\theta_3)\,\boldsymbol{R}_x\!\left(-\tfrac{\pi}{2}\right)\boldsymbol{i} + d_4\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_y(\theta_2+\theta_3)\,\boldsymbol{R}_x\!\left(-\tfrac{\pi}{2}\right)\boldsymbol{R}_x\!\left(+\tfrac{\pi}{2}\right)\boldsymbol{k} = \\
&= d_1\boldsymbol{k} + a_1\boldsymbol{R}_z(\theta_1)\,\boldsymbol{i} + a_2\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_y(\theta_2)\,\boldsymbol{i} + d_3\boldsymbol{R}_z(\theta_1)\,\boldsymbol{j} + \\
&+ a_3\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_y(\theta_2+\theta_3)\,\boldsymbol{i} + d_4\boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_y(\theta_2+\theta_3)\,\boldsymbol{k} = \\
&= d_1\boldsymbol{k} + a_1(\mathrm{c}_{\theta_1}\boldsymbol{i} + \mathrm{s}_{\theta_1}\boldsymbol{j}) + a_2\boldsymbol{R}_z(\theta_1)\,(\mathrm{c}_{\theta_2}\boldsymbol{i} - \mathrm{s}_{\theta_2}\boldsymbol{k}) + d_3(\mathrm{c}_{\theta_1}\boldsymbol{j} - \mathrm{s}_{\theta_1}\boldsymbol{i}) + \\
&+ a_3\boldsymbol{R}_z(\theta_1)\,(\mathrm{c}_{\theta_2+\theta_3}\boldsymbol{i} - \mathrm{s}_{\theta_2+\theta_3}\boldsymbol{k}) + d_4\boldsymbol{R}_z(\theta_1)\,(\mathrm{c}_{\theta_2+\theta_3}\boldsymbol{k} + \mathrm{s}_{\theta_2+\theta_3}\boldsymbol{i}) = \\
&= (a_1\mathrm{c}_{\theta_1} + a_2\mathrm{c}_{\theta_1}\mathrm{c}_{\theta_2} - d_3\mathrm{s}_{\theta_1} + a_3\mathrm{c}_{\theta_1}\mathrm{c}_{\theta_2+\theta_3} + d_4\mathrm{c}_{\theta_1}\mathrm{s}_{\theta_2+\theta_3})\boldsymbol{i} + \\
&+ (a_1\mathrm{s}_{\theta_1} + a_2\mathrm{s}_{\theta_1}\mathrm{c}_{\theta_2} + d_3\mathrm{c}_{\theta_1} + a_3\mathrm{s}_{\theta_1}\mathrm{c}_{\theta_2+\theta_3} + d_4\mathrm{s}_{\theta_1}\mathrm{s}_{\theta_2+\theta_3})\boldsymbol{j} + \\
&+ (d_1 - a_2\mathrm{s}_{\theta_2} - a_3\mathrm{s}_{\theta_2+\theta_3} + d_4\mathrm{c}_{\theta_2+\theta_3})\boldsymbol{k}
\end{aligned}
$$

where the basic properties of elemental rotations sequences have been applied.

The waist angle θ_1 can be easily extracted from the equation

$$-\sin\theta_1 x_{\mathrm{W}} + \cos\theta_1 y_{\mathrm{W}} = d_3$$

substituting the polar coordinates $x_{\mathrm{W}} = \rho_{\mathrm{W}}\cos\phi_{\mathrm{W}}$ and $y_{\mathrm{W}} = \rho_{\mathrm{W}}\sin\phi_{\mathrm{W}}$ of the wrist position on the xy-plane of the waist frame $\mathcal{R}_0$. The resulting equation

$$-\rho_{\mathrm{W}}\sin\theta_1\cos\phi_{\mathrm{W}} + \rho_{\mathrm{W}}\cos\theta_1\sin\phi_{\mathrm{W}} = d_3$$

can be transformed in an equation in the angle difference $\phi_{\mathrm{W}} - \theta_1$ only

$$\rho_{\mathrm{W}}\sin(\phi_{\mathrm{W}} - \theta_1) = d_3$$

by making use of the sine subtraction identity. This equation is solvable only when $\rho_{\mathrm{W}}^2 = x_{\mathrm{W}}^2 + y_{\mathrm{W}}^2 \leq d_3^2$, in which case it yields two[4] *distinct* values for $\phi_{\mathrm{W}} - \theta_1$ and thus for θ_1, corresponding to the so called **shoulder left** and **shoulder right** configurations.

Once the waist angle θ_1 has been found from the projection of the wrist position on the xy-plane, the elbow angle θ_3 may be computed. The sum of the squares of

$$
\begin{aligned}
\cos\theta_1 x_{\mathrm{W}} + \sin\theta_1 y_{\mathrm{W}} - a_1 &= +a_2\cos\theta_2 + a_3\cos(\theta_2+\theta_3) + d_4\sin(\theta_2+\theta_3) \\
z_{\mathrm{W}} - d_1 &= -a_2\sin\theta_2 - a_3\sin(\theta_2+\theta_3) + d_4\cos(\theta_2+\theta_3)
\end{aligned}
$$

gives the following equation

$$(\cos\theta_1 x_{\mathrm{W}} + \sin\theta_1 y_{\mathrm{W}} - a_1)^2 + (z_{\mathrm{W}} - d_1)^2 = a_2^2 + a_3^2 + d_4^2 + 2a_2a_3\cos\theta_3 + 2a_2d_4\sin\theta_3$$

[4] The equation $a\sin x = b$ has solutions $x = \begin{cases} \arcsin\frac{b}{a} + 2k\pi \\ \pi - \arcsin\frac{b}{a} + 2k\pi \end{cases}$, $k \in \mathbb{Z}$, when $a \neq 0$.

after simplifying many terms by means of the sine and cosine subtraction identities $\mathrm{c}_{\theta_2+\theta_3}\mathrm{c}_{\theta_2} + \mathrm{s}_{\theta_2+\theta_3}\mathrm{s}_{\theta_2} = \mathrm{c}_{\theta_2+\theta_3-\theta_2} = \mathrm{c}_{\theta_2}$ and $\mathrm{s}_{\theta_2+\theta_3}\mathrm{c}_{\theta_2} + \mathrm{c}_{\theta_2+\theta_3}\mathrm{s}_{\theta_2} = \mathrm{s}_{\theta_2+\theta_3-\theta_2} = \mathrm{s}_{\theta_2}$. As before, it is better to study the above equation in the elbow angle θ_3 through the change of variable to polar coordinates $a_3 = \rho_\mathrm{E} \sin\phi_\mathrm{E}$ and $d_4 = \rho_\mathrm{E} \cos\phi_\mathrm{E}$, justified by the fact that d_4 and a_3 are the rectangular coordinates of the wrist on the zx-plane of the elbow frame $\mathcal{R}_2$. The resulting equation

$$2a_2\rho_\mathrm{E}(\sin\phi_\mathrm{E}\cos\theta_3+\cos\phi_\mathrm{E}\sin\theta_3) = (\cos\theta_1 x_\mathrm{W}+\sin\theta_1 y_\mathrm{W}-a_1)^2+(z_\mathrm{W}-d_1)^2-a_2^2-a_3^2-d_4^2$$

can be transformed in an equation in the angle difference $\phi_\mathrm{E} - \theta_3$

$$2a_2\rho_\mathrm{E}\sin(\phi_\mathrm{E}-\theta_3) = (\cos\theta_1 x_\mathrm{W}+\sin\theta_1 y_\mathrm{W}-a_1)^2+(z_\mathrm{W}-d_1)^2-a_2^2-a_3^2-d_4^2$$

by making use of the sine subtraction identity. This equation admits a solution only when $\left((\cos\theta_1 x_\mathrm{W}+\sin\theta_1 y_\mathrm{W}-a_1)^2+(z_\mathrm{W}-d_1)^2-a_2^2-a_3^2-d_4^2\right)^2 \leq 4a_2^2\rho_\mathrm{E}^2$, in which case it yields two *distinct* values for $\phi_\mathrm{E} - \theta_3$ and thus for θ_3, corresponding to the so called **elbow up** and **elbow down** configurations.

Once the elbow angle θ_3 has been found, the shoulder angle θ_2 may be inferred studying the following equations

$$\begin{aligned}\cos(\theta_2+\theta_3)(\cos\theta_1 x_\mathrm{W}+\sin\theta_1 y_\mathrm{W}-a_1)-\sin(\theta_2+\theta_3)(z_\mathrm{W}-d_1) &= a_2\cos\theta_3+a_3\\ \sin(\theta_2+\theta_3)(\cos\theta_1 x_\mathrm{W}+\sin\theta_1 y_\mathrm{W}-a_1)+\cos(\theta_2+\theta_3)(z_\mathrm{W}-d_1) &= a_2\sin\theta_3+d_4\end{aligned}$$

through the change of variable to polar coordinates $\cos\theta_1 x_\mathrm{W}+\sin\theta_1 y_\mathrm{W}-a_1 = \rho_\mathrm{S}\sin\phi_\mathrm{S}$ and $z_\mathrm{W}-d_1 = \rho_\mathrm{S}\cos\phi_\mathrm{S}$, justified by the fact that $z_\mathrm{W}-d_1$ and $\cos\theta_1 x_\mathrm{W}+\sin\theta_1 y_\mathrm{W}-a_1$ are the rectangular coordinates of the wrist on the zx-plane of the shoulder frame $\mathcal{R}_1$. The resulting equations

$$\begin{aligned}\rho_\mathrm{S}\sin\phi_\mathrm{S}\cos(\theta_2+\theta_3)-\rho_\mathrm{S}\cos\phi_\mathrm{S}\sin(\theta_2+\theta_3) &= a_2\cos\theta_3+a_3\\ \rho_\mathrm{S}\sin\phi_\mathrm{S}\sin(\theta_2+\theta_3)+\rho_\mathrm{S}\cos\phi_\mathrm{S}\cos(\theta_2+\theta_3) &= a_2\sin\theta_3+d_4\end{aligned}$$

can be transformed in two equations in the angle difference $\phi_\mathrm{S} - \theta_2 - \theta_3$

$$\begin{aligned}\rho_\mathrm{S}\sin(\phi_\mathrm{S}-\theta_2-\theta_3) &= a_2\cos\theta_3+a_3\\ \rho_\mathrm{S}\cos(\phi_\mathrm{S}-\theta_2-\theta_3) &= a_2\sin\theta_3+d_4\end{aligned}$$

by making use of the sine and cosine subtraction identity. These equations admit a solution only when $\max\left\{(a_2\cos\theta_3+a_3)^2,(a_2\sin\theta_3+d_4)^2\right\} \leq \rho_\mathrm{S}^2$, and have to be solved *simultaneously* in order to give a unique value for $\phi_\mathrm{S} - \theta_2 - \theta_3$ and thus for $\theta_2 + \theta_3$. Finally, the shoulder angle θ_2 is immediately found subtracting the value of θ_3 from the value of the sum $\theta_2 + \theta_3$ just computed.

Wrist Coordinates

Once the waist angle, the shoulder angle and the elbow angle have been found, the wrist angles can be inferred from the orientation component of the desired hand pose. The

hand orientation may be obtained adjusting the orientation of the hand with respect to the wrist through the wrist orientation, i.e. $\boldsymbol{R}_{\mathrm{H}} = \boldsymbol{R}_{\mathrm{W}}\boldsymbol{R}_{\mathrm{H}}^{\mathrm{W}}$, hence the hand orientation relative to the wrist can be computed as

$$\boldsymbol{R}_{\mathrm{H}}^{\mathrm{W}} = \boldsymbol{R}_{\mathrm{W}}{}^{\mathsf{T}}\boldsymbol{R}_{\mathrm{H}}$$

where $\boldsymbol{R}_{\mathrm{H}}$ is the desired hand orientation matrix and $\boldsymbol{R}_{\mathrm{W}}$ is the wrist orientation matrix, that can be computed as

$$\boldsymbol{R}_{\mathrm{W}} = \boldsymbol{R}_1^0\boldsymbol{R}_2^1\boldsymbol{R}_3^2 = \boldsymbol{R}_z(\theta_1)\,\boldsymbol{R}_y(\theta_2+\theta_3)$$

from the values of the joint angles θ_1, θ_2, θ_3 of the arm.

On the other hand, the hand is constrained by the forearm through the kinematic chain of the wrist, so the orientation of the hand with respect to the wrist can be also deduced from the sequence of rotations about z and x axes of links frames 4, 5 and 6

$$\boldsymbol{R}_{\mathrm{H}}^{\mathrm{W}} = \boldsymbol{R}_4^3\boldsymbol{R}_5^4\boldsymbol{R}_6^5 = \boldsymbol{R}_z(\theta_4)\,\boldsymbol{R}_x(\alpha_4)\,\boldsymbol{R}_z(\theta_5)\,\boldsymbol{R}_x(\alpha_5)\,\boldsymbol{R}_z(\theta_6)\,\boldsymbol{R}_x(\alpha_6)$$

starting from link frame 3. As before, the above expression may be significantly simplified by considering the specific geometric features of simple manipulators: the motion axes of the wrist are mutually orthogonal, thus $\alpha_4 = \pm\frac{\pi}{2}$, $\alpha_5 = \pm\frac{\pi}{2}$ and the approach direction is aligned to the last motion axis, thus $\alpha_6 = 0 \wedge \pi$.

The angles $\alpha_4 = -\frac{\pi}{2}$, $\alpha_5 = +\frac{\pi}{2}$ and $\alpha_6 = 0$ are here assumed, because they reflect the geometric layout of the majority of articulated manipulators, while the other angles give equivalent results, only with different signs. Substituting the simplified DH parameters into the above expression of the hand orientation relative to the wrist gives

$$\boldsymbol{R}_{\mathrm{H}}^{\mathrm{W}} = \boldsymbol{R}_z(\theta_4)\,\boldsymbol{R}_x\big(-\tfrac{\pi}{2}\big)\,\boldsymbol{R}_z(\theta_5)\,\boldsymbol{R}_x\big(+\tfrac{\pi}{2}\big)\,\boldsymbol{R}_z(\theta_6)\,\boldsymbol{R}_x(0) = \boldsymbol{R}_z(\theta_4)\,\boldsymbol{R}_y(\theta_5)\,\boldsymbol{R}_z(\theta_6)$$

therefore θ_4, θ_5 and θ_6 can be easily extracted because they are the z-y-z Euler angles of this attitude matrix.

2.4 Manipulator Performances

In addition to sheer manufacturing operations, such as machining, processing, finishing, assembling, etc.also testing becomes of paramount importance in the economy of a reliable productive process, because it is required to assess the quality and the conformity of the single product, in particular, or the entire manufacture, in general. Customary testing operations include inspection of mechanical parts, detection of defects and identification of contours. This further step clearly requires the manufacturers to perform several and diverse measurements on the samples. Commercial manipulators equipping specific tools tailored for measuring purposes, such as touch probes and laser scanners, perfectly fit for that purpose; since the end effector manages to span the 3D space surrounding the manipulator, the measuring tool carried by the flange can be moved and operated by that manipulator, giving rise to a flexible and automated measuring instrument, thanks to its good level of dexterity and mobility.

The kinematic performances of manipulators may be assessed by means of a reliable external instrumentation, adapting some concepts borrowed from metrology, i.e. the science of measurement. The following discussion makes indeed full use of the technical jargon of metrology, entirely gathered and extensively explained in [29]. Manipulator kinematics is now investigated as if the manipulator were a proper measuring instrument; as a matter of fact, any manipulator can be thought as a three-dimensional position transducer: the measurand[5] is the 3D spatial coordinate of the centre of the tool installed on the manipulator end effector, also referred as tool centre point (TCP), and the indication[6] is the position value computed processing the joints readings though the manipulator kinematic function, which is the measurement model[7]. Most of the properties involved in the description of the kinematic performances of manipulators can be quantified through some numerical indices; such quantities must be carefully interpreted, because lower values mean better performances and vice versa.

2.4.1 Precision

In metrology, **precision** is one of the main properties of a measuring instrument and is defined as the degree of concordance among indications given in repeated measurements of the same measurand under certain conditions. Basing on such conditions, measurement **precision** gives rise to measurement **repeatability**, when the successive measurements are performed in *same* location within a *brief* time period, and measurement **reproducibility**, when the successive measurements are performed in *different* locations within a *long* time period. In any case, the precision is normally quantified by means of statistical measures of dispersion, e.g., variance or standard deviation.

In the scope of manipulator kinematics, precision represents the capability of a manipulator to move its end effector to the same position in the work-space with the same spatial orientation, when subject to the same command. From a technical point of view, the precision of a manipulator can be characterised driving the end effector to the same target pose several times, surveying the position *actually* attained at each trial, and computing a numerical estimate of the amount of spread in the collected readings.

Since the manipulator precision is normally assessed by processing the results coming from a set of measurements taken in the same place and in a limited time, the term precision can be informally confused with the term repeatability, which is far more widespread in the industrial robotics community. Manipulator manufacturers generally report the position repeatability in the manipulator technical specifications; nevertheless, as the measurement protocol adopted to characterise the repeatability might differ from one manufacturer to another, e.g., owing to very diverse load and speed conditions or to the work-space target volume, it is *virtually* impossible to compare the values of this specification among different manipulator models. To overcome this problem,

[5]The measurand is the target quantity under measurement.

[6]The indication is the reading value yield by the measuring instrument.

[7]The measurement model is the mathematical relationship relating the directly measured quantities to the actually provided quantities.

all manipulator manufacturers should gauge and declare the repeatability by means of procedures compliant with current international standards (see [30] and [31]).

The manipulator repeatability is essentially due to the quality of the components constituting the manipulator, in general. More specifically, it is heavily determined by backlash effect in the geared transmission system, mechanical clearances in the supporting frame, finite numeric precision of the manipulator controller and limited resolution of joints position transducers. Repeatability of standard industrial robots is commonly quite satisfactory as it typically ranges between some hundredths and few tenths of mms although it cannot be improved by the manipulator user at all.

2.4.2 Accuracy

In metrology, **accuracy** is one of the main properties of a measuring instrument and is defined as the degree of proximity between the indication and the true value of the measurand. It is thus intimately connected to the concept of measurement **error**, i.e. the numerical deviation between the estimated value and the real value of the measurand. **Accuracy** must not be confused with **trueness**, which is the degree of proximity between the average of the indications given by many reiterated measurements and the true value of the measurand.

In the scope of manipulator kinematics, accuracy represents the capability of a manipulator to move its end effector to a desired position in the work-space with the desired spatial orientation, when subject to the corresponding command. From a technical point of view, the accuracy of a manipulator can be characterised driving the end effector to the same target pose several times, or to several different target poses, sampling the position *actually* reached at each trial, and computing the error between the achieved positions and the aimed positions.

The average position error gives an estimate of the average accuracy, earlier defined as trueness, whereas the maximum position error gives an estimate of the maximum accuracy, earlier defined as accuracy. The strict definition of accuracy cannot distinguish between systematic errors, exclusively taken into account by trueness, and random errors, exclusively taken into account by precision: for this reason, low accuracy may be due to either low precision or low trueness (or even both).

The manipulator accuracy is due to the unavoidably incomplete knowledge of the manipulator mathematical model. Imperfections and limitations in *practically* any constitutive element of a manipulator, e.g., interface slacks, lock plays and dimensions tolerances due to machining and assembling processes, materials flexibility, structural compliances, thermal drift, motion transmission system hysteresis, transducers resolution, controller round-off errors, may contribute to restrict the overall accuracy.

Manipulator manufacturers frequently omit the position accuracy in the manipulator technical specifications, even though there exist international standards prescribing the modality of accuracy assessment and statement (see [30] and [31]). The accuracy of standard industrial robots is normally rather poor as it usually ranges from some tenths up to tens of mms and, in general, it is *far* worse than the repeatability, although, unlike the latter, it can be partly improved by the manipulator user by means of rigorous

ad hoc procedures. The enhancement of the manipulator accuracy is the subject of **manipulator calibration**, widely discussed in chapter 3.

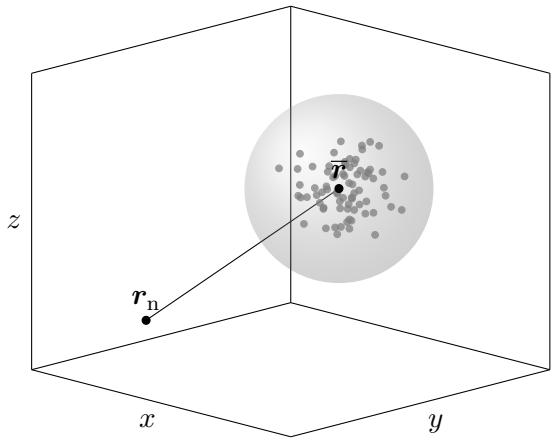

Figure 2.4: 3D tool position in repeated trials for assessing precision and accuracy.

2.4.3 Resolution

In metrology, **resolution** is one of the main properties of a measuring instrument and is defined as the *smallest* change of the measurand that is capable of determining an appreciable change in the indication.

In the scope of manipulator kinematics, resolution represents the smallest displacement of the end effector the manipulator is able to yield and mainly depends on the resolution of joints transducers, i.e. the position sensors measuring the displacement of the joints. For instance, the resolution of a rotary incremental encoder installed on a revolute joint is the minimum angular deviation of the joint the encoder manages to detect and depends on the number of segments on the entire round periphery of the disk, corresponding to the number of different angular measurement per revolution.

Finite resolution deeply affects the kinematic performances of manipulators for it automatically limits both precision and accuracy, in fact it gives rise to a rough estimate of the lower bound for those properties: if the misplacement of the manipulator end effector from a certain reference position induces a deviation in a joint smaller than the joint resolution, the transducer is not able to recognise the occurred change and continues to provide the same reading.

On the other hand, the resolution of transducers located in different joints may condition the *overall* resolution in different way; for instance the resolution of the waist, the shoulder or the elbow transducers is far more critical than the resolution of the wrist transducers because it gets amplified by a longer kinematic structure. Manipulator manufacturers customarily report the resolution of the joint position transducers in the manipulator technical specifications.

2.5 Manipulator Programming

Industrial robots embody the bridge between rigid automation, that is the world of fixed automated processes, and flexible automation, that is the ensemble of programmed automated processes. Manipulators have indeed spread considerably in industry, because they provide an adequate degree of adaptability, i.e. they manage to deal with very *diverse* applications; they are thus flexible machines that have to be suitably programmed to perform specific tasks and to accommodate the needs of the user.

Programming a manipulator means developing a software code meant to be executed by the controller to drive appropriately the servo actuators, in such a way that the manipulator end effector could perform the expected motion in the work-space. The programmer is thus required to describe, somehow, the spatial motion of the manipulator under programming. Manipulator programs are also the basic intelligent interface between the mechanical machine and its human user.

In the vast majority of standard industrial applications, manipulators are usually required to perform repetitive tasks, following recurring paths patterned on fixed sequences of points within the work-space. Manipulator programs emerging from this standard operating procedure are thus compiled *once* by the manipulator user and run *continually* by the manipulator controller.

2.5.1 On-line Programming

The aforementioned points, needed to trace a desired geometric **path** in the work-space, may be directly defined by performing *materially* the motion of the manipulator end effector along that path, giving rise to the so called **on-line programming**.

After defining the task the manipulator is supposed to execute, the operator selects a suitable set of points in the work-space pivotal for that task, sequentially brings the manipulator TCP into such points one at a time and records the values of the joints attained at each point. The complete motion is eventually programmed through a suitable interpolation of the stored tool poses.

The joint coordinates corresponding to the target tool positions are instructed, or 'taught', by the manipulator user through a dedicated panel, called teach pendant, which is capable of controlling either individual joints motion or separate Cartesian degrees of freedom independently. This is the reason why this sort of programming method is also also known as learn mode or teach mode.

This kind of programming strategy has been universally adopted in the early stage of industrial automation, because it is intrinsically straightforward, low cost and it provides a deep level of visual insight to the user; in particular, it proves especially useful when the manipulator is intended to perform uncomplicated tasks managing objects with elementary geometries. Manually moving the manipulator body step by step makes it easier and safer to avoid obstacles in the work-space environment and to handle constraints set by the robot structure.

Despite its intrinsic simplicity, intuitiveness and cheapness, teach mode comes in with many handicaps. First, physically engaging the manipulator to program may generate

a severe production downtime: teach method programming makes the manipulator unavailable for as long as it takes to reach all the target points and record the corresponding data in the controller memory, or even to test the entire completed program for validation and safety reasons, forcing the production process that manipulator takes part in to be halted. The application program developed by means of taught poses is closely related to the manipulator specimen under programming, therefore it is intimately tailored to a *specific* unit and it is not easily reproduced on different units of the same model. Last but not least, since the teaching procedure requires the close intervention of the manipulator operator, on-line programming is very prone to human error and the overall resulting performance may be heavily affected also by the skill of the operator.

In any case, such a programming philosophy relies on the assumption that repeated commands get translated into equal movements of the manipulator. Unfortunately, as discussed in subsection 2.4.1, the end effector position is always changeable and this stability is numerically measured by the manipulator precision. The absolute error in positioning the tool is not an issue, as long as it keeps relatively stable among different repeated runs: the performance provided by the program are still good, even if the the achieved poses are rather different from the computed poses, therefore the manipulator accuracy does not affect the result *whatsoever*. Since manipulators are typically designed, constructed and assembled in order to minimise their overall repeatability, they are *already* inherently suitable enough for tasks programmed by teach mode.

2.5.2 Off-line Programming

The aforementioned points, needed to trace a desired geometric path in the work-space, may be indirectly defined by reproducing *virtually* the motion of the manipulator end effector along that path, giving rise to the so called **off-line programming**.

The movements involved by the task are commonly emulated by means of a dedicated simulation environment. In order to recreate a virtual representation of the manipulator work-cell, this software obviously expects all the geometrical data not only about the manipulator unit, but also about all tools, equipment, machinery and objects involved in the definition of task, and the environment in general: these data are habitually referred as computer-aided-design (CAD) models. Any object, more or less complex, may be taken into account and managed by the simulation software, provided that it has been preventively modelled in terms of shapes and volumes.

Once the task the manipulator is intended to perform has been assigned, the manipulator operator segments the path into smaller tracts and picks a significant set of key points needed to trace the path, then inputs them to the simulation environment, where a suitable computing engine calculates the values of the joint coordinates when the TCP reaches those points. As for teach mode, the complete motion is eventually programmed by means of a proper interpolation of the stored tool poses.

Simulating the manipulator movements makes more difficult to take into account the manipulator joints strokes and links constraints, which have to be clearly reflected by dedicated software limitations; collision avoidance may also represent an issue, because geometrical models may be quite different from the physical objects they reproduce.

Unlike programming via teach pendant, in which the extraction of the joint coordinates from the tool poses is a transparent and straightforward process, in this case both a satisfactory computational power and a rather deep understanding of manipulator kinematics are respectively demanded to the computer and the user.

In spite of these minor drawbacks, off-line programming brings many benefits to manipulator users. First, it is the most natural choice when accurate CAD data are available and it represents the only way to integrate effectively these design tools into the whole productive process. Representing schematically a task by means of a tailored software provides an excellent for performing process optimisation, needed to minimise the times required by specific phases of the production process. Manipulators need not to be at programmer's disposal during coding and the whole program development can be carried out far from the target manipulator work-cell. Moreover, for the same reason, manipulators can be also reprogrammed in a much easier and faster way. In the end, when manipulators are programmed off-line, the production throughput is affected only by the program deployment time and not by the program development time.

This kind of programming strategy has not been extensively adopted in the early stage of industrial automation, for it is inherently intricate, high cost and provides a poor level of visual insight to the user, and its importance has been acknowledged only recently; in particular, it proves notably crucial whenever the manipulator is supposed to perform complicated tasks handling objects with complex geometries. Even though off-line programming is by far more widespread nowadays, on-line programming is sometimes still used in the final phase for verification purposes only.

Such a programming philosophy is based on the belief that the virtual work-cell geometry *exactly* matches the real work-cell geometry. Unfortunately, as already discussed in subsection 2.4.2, the mathematical model of a manipulator is always imprecise, because of innumerable undesired issues related to the construction of actual manipulator units, and its verisimilitude is numerically measured by the manipulator accuracy. If the performed movements are far from the expected movements, because of model poorness, the quality of the task may be severely compromised. On the other hand, the CAD models of tools and equipment required by the task have to be reliable *too*, otherwise the overall performances may be *equally* jeopardised.

Unlike teach mode, decent repeatability only is not sufficient, because the program entirely stems from the manipulator nominal model, without taking into account the target unit *whatsoever*: off-line programming demands a good level of *both* trueness and precision, that is to say of accuracy, in order to yield satisfactory results. The default accuracy of standard industrial robots is seldom adequate, thus the manipulator user willing to implement effectively off-line programming has to devise a suitable strategy to improve the manipulator accuracy to the point at which it becomes at least acceptable.

Chapter 3

Manipulator Calibration

3.1 Overview

3.1.1 Motivation

The pose of the end effector of a manipulator is normally computed on-line, by feeding the joint transducers readings into a software application which implements the kinematic function of that manipulator, already defined in section 2.3. Such an abstract mathematical relationship is derived once by either the manipulator manufacturer or the manipulator user, basing on the mechanical design or a specific specimen selected as reference; either way, it cannot be applied *exactly* to all manipulator units of the same model, owing to several unavoidable issues, such as deviations from nominal design, effect of unpredictable environmental operating conditions etc., therefore the level of absolute accuracy provided by a certain manipulator unit might be rather poor, in general, or, in any case, inadequate for a particular task.

The goal of any manipulator calibration procedure is to improve the accuracy of a manipulator modifying its kinematic model, by means of a *sheer* software correction, i.e. without the need for actually modifying its mechanical construction or redesigning it all over again. Moreover, when the manipulator structure is subject to minor changes over time, e.g., due to wear, drift, maintenance, disassembly, etc., or even worse, when the entire manipulator is replaced with another unit of the same model, the *same* program operating the robot, even if complex, can be recycled after very few adjustments.

As mentioned before, great care is taken by the manufacturer at design and construction stage, for the purpose of reducing the impact of random and unpredictable effects in joint motion as much as possible, therefore industrial manipulators typically feature a quite satisfactory repeatability, which is clearly reported in the specifications, as it is a crucial requirement to implement decently teaching method programming. In contrast, systematic effects are far trickier to avert, whereby industrial manipulators reveal definitely mediocre accuracy, of which no guarantee is provided, and in any case far worse than repeatability. This intrinsic lack prevents practitioners from effectively employing manipulators in those tasks for which off-line or shared programming is required: from

this perspective, the manipulator calibration proves especially valuable, for it attempts to bring the accuracy closer, or at least comparable, to the repeatability.

3.1.2 Definition

The process of manipulator calibration consists in selecting a suitable mathematical relationship, inside a certain function class, as kinematic function for a manipulator. Whichever function class can hardly describe the limitless complexity of a *real* manipulator unit, regardless of its level of refinement, hence the discrimination has to be performed in the light of an optimality criterion, which is of course, in this case, accuracy enhancement. Calibration focuses therefore on trimming the available mathematical model of a manipulator, in order to approximate and reflect, as closely as *possible*, the physical behaviour of that manipulator.

Since the kinematic function is usually parametrised, then the whole process is equivalent to looking for a suitable set of parameters such that the function obtained substituting these parameters is capable of modelling the manipulator kinematics with higher accuracy: from this perspective, calibration can be considered as a particular kind of parameter identification problem. Unlike adaptive control applications, where model identification must be *continually* performed in order to tune dynamically a controller, calibration is carried out only *occasionally*: if the process proves successful, the identified values of the parameters are stored and read by the application designated to manipulator positioning operations, at least until the next calibration.

Besides, manipulators intended to be employed for measurement purposes, e.g., by installing specific sensors and probes on their hand, become measurement instruments themselves, and, as such, they have to be calibrated. In the context of metrology, as indicated in [29], calibration denotes the act of assessing the accuracy of a given measurement instrument, termed device under test, by means of a comparison with a standard, i.e. an object or another device able to provide corresponding values by far more reliable; very often, the outcome of this operation leads to a correction of the measurement model, technically named **adjustment**, for which the *proper* **calibration** is a prerequisite, and a following verification, technically named **recalibration**.

3.2 Target Errors Compensation

3.2.1 Sources of Inaccuracy

The most general definition of manipulator calibration, given so far, focuses its attention on the fundamental purpose *only*, which is essentially the overall improvement of the volumetric accuracy achieved by the manipulator end effector in a specific context, through a revision of the manipulator kinematic model. On the contrary, the definition gives no clues *whatsoever* about the feasible measures that may be actually taken to achieve the desired goal practically.

As a matter of fact, there are numerous elements that might affect *significantly* the positioning accuracy of actual manipulators; nevertheless, only some of them, among

all the possible ones, have been thouroughly investigated and clearly identified until now, while others are currently a challenging subject of research. The nature and the impact of these adverse effects are so assorted that cannot be taken into account all at once, therefore a calibration procedure can be designed in order to counterbalance *only* a certain selected subset of them, basing on particular reasonable criteria. According to the targeted source of error, different levels of calibration could be defined, as already codified extensively in [15] and [16].

3.2.2 Joint-Level Calibration

This kind of calibration seeks either to establish or to refine the measurement model of the manipulator joint sensors, i.e. the mathematical relationship between the reading of position transducers installed on the manipulator joints, and the actual displacement of that joints. In the first case, the procedure is performed once by the manipulator manufacturer and possibly repeated by the manipulator user whenever the unit undergoes damage or disassembly, while in the second case, the procedure is performed several times by the manipulator user for diverse reasons.

Most of the famous kinematic conventions adopted to describe the kinematics of industrial robots, and spatial linkages in general, are based on the widely accepted assumption that joints might be mechanically modelled as pure lower-pair[1] mechanisms. Although this simplification turned out to be quite realistic for several manipulator specimens, because great care is generally taken by manufacturers with the construction of the joint mechanisms, there exist still minor imperfections in the joint drives chain that can be taken in consideration only through the development of more complex models.

If the manipulator joints feature reduction gears, the motion of the motors is not the same as the motion of the joints, because of the reduction. In such a case, which is also the most common, when the transducer is physically installed on the motor, the actual joint displacement is only a fraction of the transducer reading, due to the reduction: this effect may be modelled through a dimensionless gain coefficient. If the manipulator joints host incremental transducers, which are capable of measuring only the joint relative displacements, the joint absolute positions have to be inferred from the additional knowledge of the initial joint positions: the transducer are usually reset by moving the manipulator into a reference configuration, termed **zero position** or **home position** and identified thanks to dedicated mechanical stops or optical marks: this effect may be modelled through a homogeneous offset.

Lastly, some joints might exhibit quite strange irregular motions, due to mechanical imperfections, e.g., shaft compliance or gear eccentricity, or the transducer themselves might have some minor flaws; these defects of different nature introduce undesired effects that could be corrected only by adding suitable complex functions in the model.

[1] In mechanics, a **pair**, or **joint**, is a connection between two rigid bodies, or **links**, that allows a certain relative motion between them: when the bodies are in contact through a surface, the connection is termed **lower pair**.

3.2.3 Geometric Calibration

This kind of calibration aims to enhance the accuracy of the entire kinematic model, i.e. the mathematical relationship between the end effector pose and the joint coordinates, by adjusting *solely* the elements modelling the geometric features of a manipulator, in order to match, as much as possible, the real shapes and sizes of that manipulator. At this stage, it is assumed that a joint-level calibration has been previously performed, ergo the joint positions are already available, even though a model of the joints is often partially embedded into the complete geometric calibration. In any case, all links are considered rigid bodies and all joints are considered lower pairs, therefore no unexpected motion is taken into consideration.

The most investigated and acknowledged source of inaccuracy in the kinematics of manipulators is *indubitably* the geometric error, i.e. the discrepancy between the ideal geometry of the general manipulator design and the real geometry of the particular manipulator specimen. Any modifications in the mechanical structure of a manipulator, not related to the motion or the loading condition, may be considered geometric errors: these alterations are inescapably present in all industrial robots and, in general, all complex machines, for they are due to mechanical tolerances arising from manufacturing and assembling processes, and mechanical changes coming from natural drift, wear, and routine maintenance operations[2].

Even though whatever element of the physical structure of a manipulator might *potentially* exhibit variations with respect to the nominal design, only the errors in those geometric quantities involved in the spatial relationships between joint motion axes *actually* impact the accuracy of the manipulator kinematics: such quantities, such as the length or the width of the links, the distance and the orientation between joints, are indeed the same parameters used to describe the manipulator kinematics, thus the accuracy can be improved estimating precisely these parameters.

Extracting the geometric features of a manipulator from its mechanical design, which relies on several devised elements of the desired mechanical architecture, such as exact parallelism, incidence and orthogonality of subsequent joint motion axes, naturally leads to a very compact and elegant form of the kinematic model, involving very few parameters, because all the available analytical simplifications can be fully exploited. For this reason, in the scope of kinematic compensation, it is not sufficient to adjust only the parameters already contemplated to improve the model accuracy but it is necessary to introduce additional parameters accounting for all the feasible geometric irregularities.

3.2.4 Structure Deformation Calibration

This kind of calibration strives to *further* improve the accuracy of the entire kinematic model, by including and fitting exclusively those elements that go beyond the scope of sheer manipulator geometry, in the effort to capture the alterations in the manipulator

[2]More details about a tailor-made calibration meant to be performed after disassembly and replacement for maintenance and failure, can be found in [17].

kinematic behaviour due to the current operating conditions. At this stage, the manipulator is assumed to be already geometrically calibrated, for this procedure is normally conceived as correction of the residual errors left out by the geometric compensation.

The usual geometric calibration is based on the belief that the manipulator kinematic model could be *completely* described just by means of a set of geometric parameters and the joint displacements, for the links are regarded as rigid bodies and joints are regarded as lower pairs. Sometimes, this assumption may prove overstrict, because other more or less relevant additional effects, not explainable in terms of geometry, are thus wholly ignored: the result is an incomplete modelling that leads to only partial, and sometimes disappointing, accuracy refinement achievement.

All the physical effects and phenomena, other than flaws in shapes and sizes, able to affect the kinematics of a manipulator and thus degrade its position accuracy, are termed non geometric errors. They clearly have deeply diversified nature. Some of them are mainly related to the mechanical transmission of the joint motion, from the drive up to the load, e.g., gear backlash, bearing clearance or seal friction, hence their influence is primarily trimmed by the manipulator manufacturer directly at the production stage, in proportion to the outlined performance targets. Some others are the major sources of deformation in the overall mechanical framework, such as the temperature of the environment or the manipulator itself and the flexibility of the manipulator components. these factors cannot be taken into account by the manipulator manufacturer, for they are intimately related to the variable conditions the manipulator is required to operate in. As regard the above mentioned effects, the conditions may be the ambient temperature, the working speed and the payload.

Since deformations of the manipulator structure are extremely hard to model and closely depend on the specific operating conditions, then less study has been carried out in this area so far and such kind of calibration is thus not as popular as the standard geometric calibration. Nevertheless, there is great interest, as far as research is concerned, in weighting the diverse error sources and determining which are the most critical among them: considering that experimenter discoveries and claims are rather discordant, it is evident that the degree of relevance of a particular issue could be related to the emphasis placed on it by the manipulator manufacturer, at design, cast or assembly stage.

Elastic Deflection

As the majority of real mechanical machines, also manipulators are subject to elasticity; the end effector frequently exhibits indeed a limited elastic behaviour, that is to say a reciprocal proportionality between the stress, i.e. forces and torques, and the strain, i.e. displacement distances and twist angles: when subject to a stress, it gets deformed, and, conversely, when subject to a deformation, it develops a resistance.

The end effector is not a stand-alone object with well defined physical properties and its overall elastic behaviour is clearly determined by the flexibility of the entire manipulator. All families of manipulators ordinarily feature two separate dominant sources of flexibility: the **link compliance**, i.e. the elasticity due to the finite stiffness of the arms body and shell, and the **joint compliance**, i.e. the elasticity due to the

finite stiffness of the drives servomotors and transmission.

The compliance of the link frame is generally fairly substantial and depends on the materials employed to fabricate it. For this reason, the links of manipulators are not perfect rigid bodies whatsoever and might be slightly deformed when subjected to a considerable mechanical stress, such as tension, compression, bending or torsion.

Although the compliance of the joint motor is quite significant, this is not a real problem, because, as soon as the motor shaft gets strained, the misplacement is immediately sensed and properly compensated. On the contrary, the limited compliance of the transmission, i.e. rods, couplings and gears, might actually give rise to errors, for the joint position sensors are customarily installed at the motor side, upstream of the transmission, hence relative motions along the chain cannot be appreciated: such a problem might be solved either by placing the transducers downstream of the transmission (though at the cost of losing the resolution enhancement provided by the reduction ratio) or by mounting tailored transducers on the links themselves. On the other hand, minor displacements inconsistent with the kind of motion allowed by the joints e.g., radial and axial slide or bending in rotary joints, cannot be detected at all, regardless of the charged compliant element.

In any case, whatever compliant component is actually interested, the manipulator is elastically deformed by the same major loading contributions: the distributed weight of the the payload or the manipulator structure itself and the localized force applied to the manipulator tool. As regards link bending and joint torsion, the farther the load is applied from the proximal link end, the larger is the extent of the deflection.

Thermal Strain

As all physical systems, also manipulators are fairly sensitive to temperature. In this regard, there are two primary thermal sources in a manipulator, namely the **ambient temperature**, that is the average temperature of the environment the manipulator has to work within, and the **body temperature**, that is the temperature inside the shell of the manipulator, due to self heating.[3]

Manipulator can be required to operate in rather diverse environments, therefore the ambient temperature, which is normally the temperature of the air surrounding the manipulator, might likely range within quite large intervals and even change substantially over time. In any case, the ambient temperature should never exceed the bounds normally reported by the manipulator manufacturer in the technical specifications, for they represent alert levels for critical performance degradation.

On the contrary, the body temperature is not a single quantity, but rather a complex distribution, therefore observing or describing its effect is a far more delicate task. The temperature gradient over the frame of a manipulator is essentially due to the heat produced by two physical phenomena, namely joule effect and friction. Friction arises in gears, bearings and links, owing to the manipulator motion, for several reasons, e.g., lu-

[3] A quantitative analysis with finite element simulation of the thermal behaviour of a standard industrial robot has been carried out in [22].

bricant viscosity or contact surface asperity and strictly depends on the joint speed profile but also on the temperature. The joint servomotors dissipate power via joule heating in the windings and the wires in general, because of the high current levels, all the time the manipulator is powered, even if still, though the losses sensitively grow when the joints do move. In any case, the manipulator starts warming up, as soon as is turned on, until a steady state condition is reached; that is the reason why the manipulator manufacturer typically reports a minimum warm up period to wait in order for the manipulator performances to stabilise.

The thermal strain induced by the manipulator self heating is generally far larger than the one induced by the environment heating, though the ambient temperature changes may start to affect significantly the performances once the warm up has been completed already. Nonetheless, higher joint speed might further increase the temperature, worsening the performances even more, or, whenever the manipulator is turned off or even only left idle for a certain amount of time, the air might cool down the frame, possibly bringing the manipulator outside its thermal steady state. In any case, the amount of thermal expansion is related to the thermal properties of the materials, such as cast iron, aluminium, etc.constituting the manipulator armour.

3.3 Calibration Process

The calibration of a manipulator is a delicate and complex technical operation and should be thus treated as such: in this light, it would be highly advisable to break up the whole problem into several recognisably circumscribed operative steps, that deal with as many clearly defined aspects and are to be carefully addressed separately, one after the other.

As thoroughly pointed out in [15] and [16], any manipulator calibration process routinely goes through the same four phases, namely selection of a proper kinematic model, collection of a suitable set of significant measurement data, solution of the parameters identification problem and correction of the control program.

3.3.1 Modelling Phase

No calibration procedure can be productively developed without a clear understanding of the basic notions of manipulator kinematics, possibly narrowed down to the manipulator family under consideration. Even though black-box[4] approaches are absolutely legitimate and not so uncommon, they typically constitute *only* a fraction of the whole calibration process and still require a minimum insight about the system.

The first step in the calibration agenda is thus the investigation of the manipulator kinematics and the consequent formulation of a corresponding mathematical model. In the scope of calibration, modelling means building a suitable analytic representation of

[4]In system and control theory, black box refers to an abstract description of the external behaviour of a system, i.e. in terms of excitation (input) and response (output) *only*, without any knowledge *whatsoever* about the actual internal structure and functioning (just like black boxes hide their content from visual inspection).

the kinematic behaviour of a manipulator, that is to say an explicit and preferably closed form for the kinematic function of that manipulator.

There are infinitely many possibilities to address the problem of kinematic modelling for calibration purposes, although not all of them turn out to be worthy choices. As already pointed out above, kinematic models that blindly trust the flawless features of nominal designs are not capable of providing acceptable results, even after correction, because of their exaggerated intrinsic simplicity, and may be thus useful for speculative purposes *only*.[5] On the other hand, even the boundless inclusion of innumerable disparate elements is of very little help, for most of the them cannot be singled out de facto, therefore such models would necessarily lead to infeasible calibrations.

In any case, it is rather clear from the above admonitions that *adequately* modelling the manipulator kinematics is not a straightforward and methodical task, in fact it demands a deep level of discernment and, sometimes, even foresight. Several different modelling strategies can prove equally valid hence there exists no ultimate model for calibration. Unfortunately, no simple rule of thumb is available in this respect, although there exist some reasonable guide lines based on three fundamental criteria: **completeness**, **proportionality** and **equivalence**. This incisive terminology has been coined for the first time in [13] and derives from concept of parametric function in differential geometry, as thoroughly highlighted in [14].

Completeness

A mode is said to be **complete** whether it is sufficiently complex, i.e. it has enough elements to completely describe the kinematics, according to the target source of error. Completeness may be also defined as the capability of a model to explain *any* arbitrary deviation from the nominal kinematics. Involving the required number of parameters does not automatically make a model complete, owing to possible redundancy. In order for a model to be complete, it must contain a minimum number of *independent* parameters. For instance, as regards geometric calibration, a complete model should be able to represent all the spatial features of a manipulator capable of affecting its kinematics. From the analytical point of view, completeness corresponds to the property of surjectivity of the model parametrisation.

Definition. A parametric function is said to be **surjective** (or **onto**) if and only if the image of the parameters domain is equal to the codomain.

Proportionality

A model is said to be **proportional** whether it is able to relate corresponding variations between the kinematics and the parameters, according to the target source of error. Proportionality may be also defined as the capability of a model to reflect any *finite*

[5] This statement in no way implies that kinematic models based on the nominal design are useless in practice: as a matter of fact they are simple enough to allow straightforward manipulations and might be thus effectively employed to compute the inverse kinematics, as exemplified in subsection 2.3.6.

deviation from the nominal kinematics with *finite* changes of the parameters. A non proportional model is subject to **singularities**, i.e. configurations in which the model is discontinuous so a slight shift in the kinematics may be mirrored by huge changes in some of the parameters.[6] For instance, as regards geometric calibration, a proportional model should be capable of accounting for small errors in the manipulator geometry by means of small corrections of the parameters. From the analytical point of view, proportionality corresponds to property of regularity of the model parametrisation.

Definition. A parametric function is said to be **regular** if and only if it is differentiable with respect to the parameters and the Jacobian matrix is non singular for every value of the parameters.

Equivalence

A model is said to be **equivalent** whether it is able to transform into another model simply by changing its parameters, according to the target source of error. Equivalence can be also defined as the capability of the model to describe any deviation from the nominal kinematics just as well as other models with the same degree of complexity. Different incomplete but proportional models with the same number of parameters might account for different subsets of the possible kinematics variations and are thus not necessarily equivalent. For instance, as regards geometric calibration, equivalent models should be equally capable of describing the errors in the manipulator geometry. From the analytical point of view, proportionality corresponds to the property of congruence between model parametrisations.

Definition. Two parametric functions are said to be **congruent** if and only if there exists a differentiable change of variables between their sets of parameters able to transform one into another.

3.3.2 Measurement Phase

The developed kinematic model of the manipulator should be capable of relating the displacement of the joints to the pose of the end effector, through a suitable set of variables, *prudently* chosen at modelling stage in order to account for the target error sources. The calibration procedure aims at pinpointing a reliable value for such quantities after probing the actual kinematic behaviour of the specific manipulator unit under observation; for this purpose, the input and output quantities of the kinematic model, namely the joint coordinates and the end effector pose respectively, have to be repeatedly sampled in different kinematic configurations. The measurement step is perhaps more

[6] The terms **singularity** and **jacobian** are here used with a slightly different acceptance than in the usual context, even though their mathematical meaning is still the same. In manipulator mechanics, a kinematic singularity denotes those combinations of joint positions such that the end effector loses one or more degrees of freedom, that is to say there are no joint motions able to produce certain end effector motions; it is strictly related to the concept of kinematic Jacobian, which denotes the matrix mapping joint velocities into flange pose time derivatives and loses indeed rank at singularities.

straightforward than the modelling step, though it *still* requires great awareness, because the remaining steps strongly rely on its quality.

The measurement configuration must honour the selected model and the collected data must be relevant enough, in the sense that they have to permit the determination of the parameters included in the model: if the collection of data is scarce or not sufficiently rich and assorted, the identification step starts off disadvantaged already and may *even* fail completely. On the other hand, only the indispensable number of measurement data should be stored, because there is no *substantial* advantage in adding measurements at will beyond a certain threshold, in fact both the measurement step and the identification step would last longer and the numerical procedure could even freeze when forced to handle too large data clusters.

The region of the three dimensional volume around the manipulator containing the target set of tool points selected for calibration has to be consistent with the portion of the work-space where the improvement of the tool position accuracy is desired. Exploiting the results returned by calibration to determine the pose of the manipulator end effector very far from the region actually explored during the collection of the measurement data is typically not advisable.

By definition, the measurement phase is the only one of the four that must be necessarily carried out on line, in the field, although it may be automated in such a way that the human intervention is not strictly mandatory. There exist several configurations to collect kinematic data, though not all of them prove suitable for the task, because of different limitations, such as cost, speed, size, accuracy, and so on. Moreover, there are two alternative ways for sampling the kinematics of the end effector: the measurement method is said **active** or **open-loop**, when the end effector is detected by measuring instruments, such as theodolites, coordinate-measuring machines (CMM), cameras or laser interferometers; the measurement method is said to be **passive** or **closed-loop**, when the end effector is constrained by tailored fixtures, such as shaped plates, reference planes, cubes and tooling balls. In any case, the adopted measurement set-up should not alter the physical characteristics of the manipulator, at least the ones the calibration seeks to detect with adequate precision.

Catching the complete pose of the end effector would require, in principle, the separate measurement of its position and its orientation. Considering that the vast majority of instruments or tools actually available are not able to provide measures of both the position and the orientation of a body at the same time, whenever the data about the position alone are not enough, some work around has to be conceived. Very often, the information about the orientation of the end effector may be either literally extrapolated from or virtually embedded in the measurement of the position of more points, however not less than three, on the flange at pose of the end effector, for the coordinates of three points intrinsically define a reference frame.

Since the enhancement of the end effector accuracy is the prime goal of manipulator calibration, it follows that the pose of the end effector should be estimated with an overall corresponding level of accuracy far better than the desired level of manipulator accuracy, irrespective of whether an active measurement device or a passive reference fixture is

practically adopted. On the contrary, there is no use in employing position transducers with better specifications than the ones already integrated in the manipulator joints, because the controller relies on those devices for all operations regarding the manipulator, unless such superior transducers are mounted *permanently* on the manipulator and used in place of the built in transducers.

3.3.3 Identification Phase

The result of measurement phase is the generation of an ordered set of kinematic data or other kinds of data regarding the manipulator, corresponding to a certain number of kinematic configurations. If the measurement have been correctly performed, then the gathered data should represent different samples of the manipulator true kinematic behaviour, excluding the influence of the measurement noise, and can be thus juxtaposed with the corresponding data coming from the kinematic behaviour predicted by the model and eventually processed by a tailored algorithm that seeks to lower the discrepancy between them, in some sense, by setting the numerical value of the parameters designated at modelling stage.

As already anticipated before, the success of this step inevitably depends on the quality of the two previous steps, because they are intimately related: in particular, the selected model provides the symbolic expressions and determines the structure of the problem whereas the collected data supply the numerical values and set the size of the problem. For this reason, if either the model is redundant, incomplete or discontinuous, or the measurement data are deficient, corrupt or correlated, the identification process will *axiomatically* fail, no matter how robust and efficient it may be.

The identification of the numerical values for the model parameters, performed for the purpose of reducing the difference between the predicted kinematic behaviour and the measured kinematic behaviour, is definitely the core of the whole calibration procedure and is sometimes thus symbolically considered as the proper calibration *itself*. Nevertheless, despite its importance, identification is the only phase that does not require any insight whatsoever about the kinematics of manipulators, for it essentially consists in applying standard numerical techniques to the problem of manipulator calibration, even though a minimum knowledge of the characteristic mathematical structure may be of great help for speeding up the algorithm or improving its performances in general.

Linearity

The most widespread numerical identification techniques typically involve linear models, for they are far simpler to study and reveal some meaningful properties. On the contrary, the kinematics of manipulators is generally non linear with respect to both geometric and non geometric quantities, although, fortunately, these non linearities are often quite weak, therefore the kinematic model of manipulators is said to be **mildly non linear**. Such a property partially justifies the adoption of standard linear identification techniques, that essentially perform model linearisation, for parameter identification in the framework of manipulator calibration. On the other hand, if the performances provided

by those methods that treat the model as purely linear, are not satisfactory enough, then it might be necessary to switch to non linear identification techniques.

Recursion

If the measurement system is pretty fast, the collection of observations is not too large and the available computational resources are rather high, then data batch processing is possible: in such a case, the entire set of measurement data is stored and processed *all at once*, and the result is found right away; this kind of approach, used for instance by standard least squares or minimum variance, is called **non recursive identification**. On the contrary, if the above conditions are not concomitantly met, data sequential processing is definitely preferable, otherwise the calibration might become a dramatically slow and slack procedure: in such a case, at each step, a new single measurement is stored, added to a growing data structure, and used to correct the estimation computed at the previous step; this kind of approach, used for instance by recursive least squares or Kalman filtering, is called **recursive identification**.

Error Handling

All quantities included in the manipulator kinematic model sampled at measurement stage bring a certain degree of uncertainty and get inevitably corrupted by a minimum amount of noise, hence the data are *always* affected by measurement errors. If the measurement error is treated as unknown variable, then the parameters are determined using only the main data and ignoring the error; this kind of approach, used for instance by several least squares variations, is termed **deterministic identification**. In contrast, if the magnitude of the measurement error may be somehow estimated, for instance using the available specifications about the instrumentation employed for measurement purposes, then the parameters are computed either using only the main data or embedding also the information about the error, but, in any case, they may be characterised stochastically, i.e. the spread in their values due to noise and uncertainty is represented as random variable, with mean value and variance; this kind of approach, used for instance by minimum variance or Kalman filtering, is termed **probabilistic identification**.

Approximation

Most of the numerical identification methods make use of some differential quantities of the model. When the differentiation is performed on line, numerically, by the computer designated as solver, the identification method makes use of approximate derivatives: this approach requires *lower* modelling effort but *higher* computational effort. When the differentiation is performed off line, symbolically, either by hand or by a custom software tool, the identification method makes use of exact derivatives. This approach requires *higher* modelling effort but *lower* computational effort.

3.3.4 Adjustment Phase

The identification stage must produce, as a result, a set of values for the model parameters, or, sometimes, a set of corrections for each model parameter, and possibly, some useful information about the attained numerical performances. Once the identification has been successfully performed, the initial nominal kinematic model is updated using the new parameter values, in order to finalise the calibration procedure. This operation is carried out via software, first into the calibration specific application, for validation purposes, then into the main control application, for implementation purposes.

Validation

The identified values of the model parameters are substituted in the model function, in the same application occasionally used for the calibration of the manipulator, in order to verify the quality and the reliability of the results achieved at identification stage.

Performing another cycle of measurement, during which the manipulator must reach a set of configurations which is different but comparable to the set of configurations used for identification purposes and generating the corresponding data simulating the updated model, makes it possible to cross check the parameters values and try the attained end effector accuracy in an new independent context.

On the other hand, if the tool position error is checked in different locations pretty far from the ones that provided the data for identification, then the attained accuracy performances can be robustly sieved: even if the selected target volume of the workspace is well circumscribed, it is customary to specify the levels of the acquired accuracy in both the target volume and the entire workspace.

In any case, the collections of data resulting from the measurement round used for identification and verification purposes, are typically referred respectively as estimation data set and validation data set.

Implementation

The identified values of the model parameters are substituted in the model function, in the main application systematically used for the operation of the manipulator, in order to exploit the performance improvement brought by the results obtained at identification stage. At first sight, it would seem that such a task could be accomplished simply by incorporating the calibration results in the manipulator controller, overriding the nominal model; unfortunately, this is seldom the case. Implementation is hardly a straightforward task, for a series of reasons related to the degree of flexibility of the program running on the manipulator controller.

Some controllers do not allow the modification of their internal global variables, like the addition of new parameters or the correction of the ones already existing, hence the only way to benefit from the performances improvement of calibration is to make use of an external preprocessor, upstream of the controller, exchanging with it only joint positions data. Even when controllers do allow changes, the parameters they rely

on are seldom the same as the ones modelled and then identified by calibration, thus the user must formulate a method for reconstructing the kinematics from the calibrated parameters and pulling out again a new set of parameters compatible with the controller application. Moreover, some controllers may even further process the parameters, once fed by the user, and slightly modify them, according to some internal criteria, ineluctably undermining the performances.

Another important problem arises from the management of the manipulator positioning operations. Controllers are often able to compute the joint displacements corresponding to the desired tool poses commanded by the user, basing on the manipulator inverse kinematics; when the kinematics is based on the nominal design, the inversion of the kinematic model might be performed making the joint variables explicit, because the ideal geometric characteristics of **simple manipulators** ensure that an inverse relationship exists and has a closed form. Unfortunately, this may not always be the case for the kinematics updated by calibration, which is usually far more sophisticated, therefore the controller cannot rely on calibration data for positioning the manipulator to the requested locations with better accuracy. If a given specific task demands higher kinematic accuracy in both locating and positioning operations, then the user must design a complementary procedure to compensate the joint commands required to reach a reference tool pose, according to the calibration results, by means of suitable strategies, e.g., iterative methods, optimal control, interpolation, etc., broadly discussed in [15].

3.4 Calibration Issues

As discussed far and wide above, the calibration of a manipulator is definitely not a a straightforward operation that can be lightly performed without the proper technical background. There are several problems that arise and should be carefully considered when facing the challenge of manipulator calibration in practical terms. If such crucial issues are not cautiously taken into consideration *before* either scheduling or engaging a calibration procedure, the results could be poor if not even detrimental, to the point of jeopardising the production process the manipulator is part of.

3.4.1 Time

As most of the operations that require high precision, also manipulator calibration is a very long process; manipulators cannot be used for their routine tasks, thus becoming *unproductive*, for as long as it takes to collect and store all the measurement data required for identification.

In particular, the manipulator under calibration has to move quite slowly, in order to minimise and stabilise the heat generation, and, after reaching each location, it must remain still for a while, in order to allow its mechanical structure to damp sufficiently all the oscillations due to the motion; several configurations have to be reached and numerous samples per configuration have to be collected, in order to reduce the effect of random noise and instrument uncertainty as much as possible. Moreover, the mea-

surement equipment is typically not fixed nor permanently left close to the manipulator area and must be thus assembled and then disassembled whenever the measurement has to be performed; sometimes, it also requires precise set-up waiting times prior to use.

All the other steps of calibration can be executed off line, *without* interrupting the manipulator working schedule at all, even though the time may still represent an issue: as a matter of fact, modelling analysis is time consuming for the researcher in charge of calibration, while identification and implementation computations are time consuming for the computer devoted to calibration.

3.4.2 Cost

Precision becomes an issue also as regards the cost of the calibration: the success of calibration demands that the kinematic quantities of the end effector are measured with a considerable level of accuracy, a large investment is normally required to get *adequate* measurement equipment, including both external measurement instrumentation, e.g., measuring machines, reference tools, sensors, data acquisition systems, etc., and manipulator appliances, such as the tool mounted on the effector, specifically designed and tailored for calibration purposes, or accurate joint positions transducers. Moreover, complex devices can be generally handled *only* by specialised and trained staff.

Employing already available commercial software tools for numerical optimization such as applications, packages, libraries, routines and so on, may *significantly* raise the overall budget demanded by calibration. On the other hand, also the development of brand new custom procedures involves quite high costs in terms of human resources.

As already mentioned above, manipulators remain completely inoperative, and thus *unprofitable*, during the entire measurement operations involved by calibration, for the interruption of the productive process clearly represents an economic loss.

3.4.3 Target

It is rather evident from the above considerations that manipulator calibration is really far from being considered a fast and cheap procedure; this is the reason why it is essential to determine carefully *when* and *whether* a calibration should be performed for the application under examination: the need for calibration depends indeed on the characteristic of both the manipulator itself and the task it is entrusted with.

For instance, when the accuracy provided by a manipulator is already quite good, even if this is rarely the case, calibration would be utterly useless. Similarly, and more realistically, when the specific task to be performed does not require a given level of accuracy, very little help may derive from calibration. When the manipulator is intended to be programmed on-line by means of a teach pendant, absolute accuracy is likely not a strict requirement, for the main concern is the repetitive precision, as widely explained in section 2.5. In all of these cases, a calibration may be required only when the required standard of accuracy gets raised for whatever reason.

On the other hand, when the accuracy of a manipulator is not up to the task, calibration should be repeated periodically, in order to meet the demanded performance

standards. If the manipulator is not replaced nor dismantled, then there is typically no need for executing several calibrations; once compensated, the manipulator may continue to operate normally, even though a fast check of the accuracy might be *occasionally* performed in order to determine whether a new calibration is actually necessary.

In general, before setting up and performing a calibration, it is always advisable that the manipulator user clearly delineates the desired target, in order not to be disappointed by the results, when not compliant with the expectations.

3.4.4 Expectation

When devising or performing the calibration of a manipulator, it is very important to get a rough idea about what to expect in terms of performances achievement. The disappointment induced by the attained results does not necessarily imply a faulty compensation, in fact it typically comes simply from quite unrealistic expectations.

According to the description outlined in section 2.4, accuracy is only one of the different metrological features of manipulators, therefore other elements should be taken into consideration in order to evaluate correctly the performances provided by calibration. As a matter of fact, if the manipulator is poorly designed or cheaply constructed so that the repeatability and/or the resolution of the joint transducers are not quite decent, then the accuracy cannot be enhanced beyond a certain threshold and calibration may prove rather disappointing: as already stated, bringing the manipulator accuracy to the same order of magnitude of the manipulator repeatability represents the best achievement than could be expected from manipulator calibration.

For the same reason, the attained end effector accuracy cannot be superior, in fact not even comparable, to the accuracy of the measurement equipment employed for calibration. However, in no way the accuracy performances of a standard industrial robots can be made comparable to those of coordinate measuring machines.

The results of calibration are generally valid *only* at operating conditions similar to the ones under which calibration took place. For instance, if the manipulator were then employed far from the volume sampled during calibration, the accuracy could be rather poor, and another calibration should be performed; similarly, if a manipulator geometrically compensated under certain loading and thermal conditions were required to move at high speed and/or to lift heavy payloads, the actual tool position error could be exceptionally larger, for it would be affected by new sources of inaccuracy, not present and not compensated at calibration stage.

3.4.5 Complexity

One of the key properties of a mathematical model is complexity: In simple terms, the complexity of a model refers to the number of elements required to describe that model. As widely demonstrated, the results of calibration are deeply affected by the characteristics of model developed at the beginning. Once the target error source has been selected, a model accounting for that effect is developed by following some reasonable criteria. Some assumptions and simplifications are also made during this phase, in order

to single out the most relevant features of the problem. Eventually, the proposed model is tuned through calibration in order to provide better accuracy.

As all the next steps of the calibration process are closely sensitive to the choices made at modelling stage, if the attained performances are not satisfactory *enough*, then a reasonable choice could be to revise the model: this can be achieved by either dropping some simplifying assumptions or embedding also additional sources of error. In any case, the upgraded model is more complex, of course, but the refinement may lead to better results. The same process of revision can be repeated many times, until the outcomes start to resemble each other; at this point, if the model complexity were further increased, no significant changes could be appreciated in the results of calibration.

Such a paradoxical cycle makes clear that, even though more complex models do reflect in more precise results, there exists a limit given by the degree of relative improvement: when adding complexity to the model gives little contribution to the attained performances, the additional labour resulting from the augmented complexity is not legitimate; in this regard, [18] investigates the effect of increasing model complexity into the overall resulting accuracy improvement.

Higher model complexity does not come for free, as it implies deeper understanding, more observations, larger data sets, harder optimisation, longer procedure and overall weaker insight. A worthy calibration procedure must thus rely on a suitable trade off between the complexity of the model and the efficiency of the method.

3.4.6 Reliability

The calibration of a manipulator is not only a technical operation, but also a scientific experiment, as it involves the assessment of a hypothesis by means of empirical verification: in the case of calibration, the hypothesis is the possibility to describe the manipulator kinematics for improving its accuracy.

As all valid scientific experiments, a good calibration should be reproducible: other researchers must be able to find the same results using the same data and the same method. Moreover, unlike several unrepeatable experiments, a good calibration should be also replicable: other researchers must be able to perform again the same procedure, finding consistent results when using comparable data.

Beyond the technical speeches, a worthwhile manipulator calibration procedure must be sufficiently reliable, that is to say capable of yielding similar results at similar conditions, when repeated *in time*, i.e. on the same manipulator unit in different times, and *in space*, i.e. on different manipulator units of the same model.

3.4.7 Autonomy

As extensively argued above, designing a full procedure of manipulator calibration from scratch demands massive research effort: first, the practitioner has to attentively examine the manipulator specimen to calibrate, assess its performances with respect to the desired task, identify the significant elements eligible for correction and formulate a preliminary

target; then he has to engineer all the different steps of the complete procedure, put the pieces together, and possibly debug them both separately and jointly, if necessary.

Nevertheless, the actual process of calibration is a *sheer* sequence of operations, therefore it can be made automatic. Once a satisfactory and reliable procedure has been successfully set up, all the involved executive steps may be safely delegated to machines, without the need for direct human intervention. Whether the calibration can be fully or partially automated depends, of course, on the capability of the available hardware: the manipulator motion pattern might be programmed off-line, the tool might be fastened by the manipulator itself, a computer program might perform the computations as soon as the measurement data are available, and so on; moreover, nowadays, there even exist measurement devices capable of interfacing with a computer, and synchronising their activity to perform automatically the acquisition of measurements.

If the measurement equipment is already a part of the manipulator work-cell, for instance, when the tool position error is assessed by means of compact fixed references, and the manipulator is able to switch between tools rather quickly, for instance, when a tool changer is available, then the calibration of that manipulator may become a fairly fast automated operation, that can be *seamlessly* performed, when needed, as an integral part of the entire production process.

Chapter 4

Experimental Case Study

4.1 Study Outline

4.1.1 Activity Objective

The experimental research work is entirely aimed at developing a calibration strategy in order to improve considerably the accuracy performances of articulated robot manipulators with moderate payload, by correcting the modelling flaws due to *both* geometric and thermal errors. Nearly every operation regarding robot manipulators relies on their kinematic model, i.e. the abstract mathematical description of their spatial motion. For this reason, the overall working performances of robot manipulators, not only position accuracy, might *indirectly* benefit from the refinement of the model achieved through by means of error compensation.

From a merely numerical point of view, the desired goal is considered to be adequately accomplished as soon as the level of position accuracy attained by manipulator calibration becomes comparable with the level of position repeatability declared by the robot manufacturer, for striving for a further improvement would be an utterly pointless aspiration, according to the argumentations presented in chapter 3.

Unlike geometry, temperature is not a fixed source of inaccuracy, therefore its effect on kinematics may not be corrected once and for all. As broadly explained in subsection 3.2.4, the accuracy performances of robots typically degrade dramatically when the joints start to dissipate electrical and mechanical power loss as heat and the resulting temperature rise induces links deformation; on the other hand, they practically settle down with the temperatures as soon as the rate of heat removed by air and ground balances the rate of heat produced by friction and current.

A standard workaround to such a problem normally consists in waiting for a suitable warm up time for the manipulator to reach an adequately stable thermal condition before actually operating it, though with a significant waste of time, money and energy. For these reasons, a *real time* thermal compensation, allowing users to operate robot manipulators with neither working interruptions nor performance degradation, would remarkably improve the efficiency of the production process throughput.

4.1.2 Research Work

All the research activities have been conducted within the facilities of Axist R&D department, which commissioned the study. Axist deals with articulated robots on a regular basis, exploiting their peculiar dexterity and flexibility and installing specialised tools to implement a wide range of tasks for dimensional measurement and inspection.

Since the position accuracy of the manipulator end effector unavoidably affects the overall accuracy of the measuring system made up of the robot and the equipped tool, then perfecting the manipulator kinematic model by means of an efficient calibration scheme is of paramount importance in order to guarantee a good quality of the provided dimensional measurement service. Furthermore, the reliability of the kinematic model is also highly desired because several tasks are programmed off line in order to exploit workpiece CAD models and to share applications among either different robot manipulators or different work-cells.

The manipulator under study is a 6 DOF articulated robot COMAU NJ-130-2.0, a member of the high payload line of the COMAU robot fleet, featuring 130kg maximum wrist payload, 2050mm maximum horizontal reach and 0.07mm declared repeatability. The model unit has been delivered by the manufacturer precisely for calibration development purposes.

Figure 4.1: COMAU NJ-130-2.0 articulated robot.

The development of the geometric compensation scheme has been carried out first simulated numerically and then assessed testing another manipulator specimen, namely a 6 DOF articulated robot STÄUBLI TX90, routinely employed in the department for development purposes.

Figure 4.2: STÄUBLI TX90 articulated robot.

4.1.3 Approach Scheme

The two target sources of kinematic inaccuracy are tackled separately: first, the geometric error, i.e. the tool position error due to imperfect knowledge of the robot geometric features, then, the thermal error, i.e. the tool position error due to thermal strain of the robot body components, are compensated. Such a course of action is automatically inspired by the straightforward observation that geometry statically affects the model accuracy, thus its effect must be compensated only once, whereas temperature dynamically affects the model accuracy, thus its effect must be compensated from time to time.

On the other hand, the geometry of a robot manipulator actually changes with temperature, hence, thermal errors are technically still geometric errors, but subject to change over time. In this regard, a clarification needs to be made in order to avoid confusion: taking the room temperature as reference for the entire manipulator body, then the geometric error is the model error at reference temperatures whereas the thermal error is the difference between the model error at some temperatures and the model error at reference temperatures. Hereinafter, the compensation of the errors thus defined is respectively referred simply as **geometric calibration** and **thermal calibration**.

As regards geometric calibration, a parametric approach is here adopted, that is to say the actual geometric features of the manipulator are not directly inferred by performing single joint movements and virtually rendering the corresponding motion axes,

but indirectly determined by identifying the numerical value of a given set of kinematic parameters. Since the characteristics of the kinematic model chosen for calibration dramatically reflect in the performances of the identification procedure, the parametric method inevitably demands a mindful modelling effort crucial in order to lay the foundations of a valiant compensation strategy.

As regards thermal calibration, since the error genesis is ruled by several complex physical laws, such as heat transfer and thermal expansion, applied to an extremely heterogeneous and anisotropic mechanical structure, no precise modelling can be proposed. Nevertheless, sensing the temperature gradient in a small number of key points and presuming a simple model for the thermal strain might represent a reasonable approximation of the actual robot thermal behaviour. As already said before, temperature indirectly affects accuracy by altering the geometry, hence identification of the coefficients selected to render the thermal error may be naturally hinged on standard geometric calibration performed at different temperature conditions.

In summary, the overall calibration is eventually made up as follows: the kinematic parameters are identified several times repeatedly, sampling the manipulator kinematics as the temperatures across the robot body span over a quite wide range, then the thermal coefficients are estimated basing on the empirical correlation between temperatures and kinematics. Following the systematic paradigm outlined in chapter 3, this experimental study of robot calibration is divided in four specific phases, each of which needs special care thus it is reported separately in the following.

4.2 Kinematic Modelling

4.2.1 Kinematic Rules

Whenever calibration is aimed at geometric error compensation only, the modelling process consists in developing an expression of the end effector pose as function of the joints coordinates, using solely geometric quantities, namely angles and distances, with respect to a certain fixed reference structure, such as the room framework, through rigid body kinematics tools. The first moves of kinematic modelling stage retrace indeed the steps of basic manipulator kinematics, outlined in section 2.3.

Considering that a joint allows a certain degree of relative motion between two links, each link is a sheer body with its own motion, even if it is not completely free, being mechanically constrained by its previous links. This is the reason why a reference frame is routinely attached to each link to describe its position and the orientation; moreover, an *additional* reference frame fixed to the floor of the work-cell is used as reference for both the robot and the objects within its work-space: 8 frames are thus required for a 6DOF manipulator. Link frames are related to each other by rigid transformations, i.e. combinations of rotations and translations. Among all transformations, it is sufficient to know just the relative transformation between subsequent link frames, because the transformation between any other couple of link frames can be immediately derived by chaining all transformations between the interposing link frames.

Reference frames transformations may be described using several different mathematical tools, such as dual quaternions, homogeneous matrices or screw matrices, properly parametrised by some geometric quantities. Calibration seeks to determine the value of these quantities, hence the construction of the kinematic model requires the selection of a set of parameters. The fundamental criteria of completeness, equivalence and proportionality must be borne in mind during this step, in order to model the manipulator kinematics employing the *proper* parametrisation, i.e. the one that is both capable to describe comprehensively the manipulator geometry and suitable to allow the identification of the parameters.

Joint Motion Axis

A complete kinematic model must feature a given level of complexity, which is translated into a minimum cardinality of the parameter set. The number of parameters required to describe the relationship between link frames have to be determined. In general, an arbitrary 3D rigid transformation requires six parameters, as it defines an arbitrary position and an arbitrary orientation, or equivalently, it consists of an arbitrary translation and an arbitrary rotation. However, link frames cannot be arbitrarily selected, because they have to be compliant with the constraints set by the mechanical configuration of the kinematic chain.

As stated in subsection 3.2.3, the joints are assumed to be already calibrated and are modelled as lower pairs. Nonetheless, even though the joint sensors measurement model is neither investigated nor refined, the static shift in the joints motion is a *genuine* geometric error, and, as such, has to be included in the kinematic model. Lower pairs are simple mechanisms providing a unidirectional motion, i.e. either a translation along (for prismatic joints) or a rotation about (for revolute joints) a certain axis, termed **motion axis** or **axis of motion**. By virtue of this simplification, *only* one quantity is needed to describe the motion of a joint, provided that one axis of the reference frame attached to the link that joint is moving, is oriented with its axis of motion. In the industrial robotics literature, z axis has been conventionally aligned to the motion axis with no loss of generality, as the same reasoning would apply also for x or y axes.

Link Frames Constraints

Link frames are not freely assigned, being subject to some kinematic constraints: first of all, as explained above, to represent the motion of a joint, the z axis must be oriented with the joint motion axis; more specifically, to portray the positive direction of motion, the z axis must have the same direction of the motion axis, that is to say both orientation and sense, defined intuitively for translations and according to the right hand rule for rotations. Furthermore, to represent a rotation about the *actual* joint motion axis, the origin must lie on such axis, or, in other words, the z axis must coincide exactly with the motion axis, whenever the joint is revolute.

The minimum number of *independent* parameters required to specify the transformation yielding a given link frame, is equal to the number of kinematic constraints that

frame is subject to, or, equivalently, the complement of the number of degrees of freedom left to that frame with respect to the total number. As regards revolute joints, because z axis has to lie on the motion axis, four constraints are set and two degrees of freedom, out of the original six, remain, hence four parameters are needed.

This statement can be mathematically justified in several ways, by identifying either the constraints *set* or the degrees of freedom *left* to a pair of subsequent link frames through the geometric tools described in section A.5; three equivalent arguments are here proposed. The new link frame is identified with respect to the old link frame, by means of four vectors, namely its origin $\boldsymbol{o}$ and its coordinate axes $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$.

1. Once z axis is made coincident with the motion axis, the link frame is not fully set yet, as neither the position of its origin $\boldsymbol{o}$ on the motion axis nor the orientation of its other axes $\boldsymbol{i}$ and $\boldsymbol{j}$ on the plane perpendicular to the motion axis are constrained. Two further transformations, namely a translation along and a rotation about $\boldsymbol{k}$, can be still performed, because they give rise to other valid link frames. Two degrees of freedom are thus left.

2. The motion axis line is described by the parametric equation $\boldsymbol{r} = \boldsymbol{r}_0 + t\boldsymbol{v}$, where $\boldsymbol{r}_0$ is a point belonging to the axis and $\boldsymbol{v}$ is the direction of the axis. The z axis must be parallel to the line direction, then

$$\boldsymbol{k} = \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|}$$

   which gives two constraints, and the origin must lie on the line, then

$$\|(\boldsymbol{o} - \boldsymbol{r}_0) \times \boldsymbol{v}\| = 0$$

   which gives another two constraints. Four constraints are thus set.

3. The motion axis line is perpendicular to the plane described by the parametric equation $\boldsymbol{r} = \boldsymbol{r}_0 + s\boldsymbol{u} + t\boldsymbol{v}$, where $\boldsymbol{r}_0$ is a point belonging to the axis and $\boldsymbol{u}, \boldsymbol{v}$ are two non-parallel vectors orthogonal to the axis. The z axis must be parallel to the vector normal to the plane, then

$$\boldsymbol{k} = \frac{\boldsymbol{u} \times \boldsymbol{v}}{\|\boldsymbol{u} \times \boldsymbol{v}\|}$$

   which gives two constraints, and the segment joining the origin and the point $\boldsymbol{r}_0$ must be orthogonal to the plane, then

$$(\boldsymbol{o} - \boldsymbol{r}_0) \cdot \boldsymbol{u} = 0,\ (\boldsymbol{o} - \boldsymbol{r}_0) \cdot \boldsymbol{v} = 0$$

   which gives another two constraints. Four constraints are thus set.

Number of Parameters

The same reasoning can be obviously applied also to the transformation between the world frame and the first link frame. This further transformation is required because the absolute reference frame cannot be bound by a joint motion axis. In contrast, the end frame, i.e. the the reference frame mirroring the pose of the manipulator end effector, can be arbitrarily assigned, because there are is no joint motion axis after the last link. It is unconstrained and six degrees of freedom are available, therefore the last transformation requires six independent parameters.

In summary, the kinematic model of serial manipulators with p prismatic joints and r revolute joints must include

$$n = 2p + 4r + 6$$

parameters, then the kinematic model of articulated manipulators requires

$$n = 4 \cdot 6 + 6 = 30$$

parameters. The same formula, or an equivalent expression, has been derived also in [13], [14], [15] and [21], following various lines of reasoning.

Any kinematic model featuring at least 30 independent kinematic parameters is complete. Incomplete models may still provide satisfactory accuracy enhancement, but they lack the capability to take into account all possible errors due to the manipulator geometry. On the other hand, adding more parameters cannot *further* improve the accuracy because those parameters would be not independent and thus redundant. A certain degree of redundancy gives room for less specificity, making models suitable for generalization; owing to their simplicity, redundant models may be easier to study but usually create great troubles in the identification procedure, because the same geometric errors might be explained by *infinite* different combinations of parameters values.

4.2.2 Kinematic Parametrisation

There exist many mathematical tools, such as dual quaternions, homogeneous matrices, screw matrices, product of exponentials, etc., available to describe rigid transformations, poses or reference frames. Homogeneous matrices are here adopted to represent the transformations involved in manipulator kinematics, following the formalism outlined in section B.3. Specifically, the relationship between the $(i-1)$-th link frame $\mathcal{R}_{i-1}$ and the i-th link frame $\mathcal{R}_i$ is described by the matrix

$$\boldsymbol{T}_i^{i-1} = \begin{pmatrix} \boldsymbol{R}_i^{i-1} & \boldsymbol{t}_i^{i-1} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{i}_i^{i-1} & \boldsymbol{j}_i^{i-1} & \boldsymbol{k}_i^{i-1} & \boldsymbol{o}_i^{i-1} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where $\boldsymbol{o}_i^{i-1}$ is the position of the origin of $\mathcal{R}_i$ in $\mathcal{R}_{i-1}$, and $\boldsymbol{i}_i^{i-1}$, $\boldsymbol{j}_i^{i-1}$, $\boldsymbol{k}_i^{i-1}$ are the directions of the coordinate axes of $\mathcal{R}_i$ in $\mathcal{R}_{i-1}$.

As shown above, four independent variables are needed to specify adjacent link frames transformations, therefore this homogeneous transformation matrix must be a function

of four kinematic parameters. As the origin and the orientation of link frames are constrained, both angles and distances must be included in the matrix parametrisation. Even if the task of collecting four geometric quantities to parametrise homogeneous matrices is rather trivial per se, great care has to be paid in this phase, in order to render kinematic transformations by means of parameters that vary *continuously* with the geometry of joint axes. In other words, the target parametric transformation must be able to track smoothly whichever variation of the kinematic model from the nominal configuration, used to select the parametrisation in the first place.

Among all the possible choices of kinematic parametrisations, only those guaranteeing proportionality are suitable to set up models for calibration purposes, because proportionality is a crucial prerequisite for stable and reliable numerical optimisation. Non proportional models miss the possibility to map the spatial configuration of the joint motion axes evenly and without jumps in the parameters, hence they may account for slight geometric variations by means of huge parametric changes.

Near Perpendicular Joint Motion Axes: Denavit-Hartenberg

The most popular and widespread parametrisation for manipulator kinematics is certainly the Denavit Hartenberg convention, extensively used even by robot controllers owing to its simplicity. It proposes a selection of link frames absolutely compliant with the rules given so far and involves indeed four independent parameters, therefore it is the first reasonable candidate for kinematic modelling.

As described in subsection 2.3.4, the Denavit-Hartenberg convention is essentially based upon the identification of the common normal, i.e. the line orthogonal to both motion axes. A couple of transformations with respect to z axis brings the link frame to the intersection between the first motion axis and the common normal and directs x axis with it; a couple of transformations with respect to x axis, brings the link frame to the intersection between the common normal and the second motion axis and directs z axis with it. The DH homogeneous transformation matrix is thus given by

$$\boldsymbol{T}_{\mathrm{DH}} \coloneqq \begin{pmatrix} +\mathrm{c}_\theta & -\mathrm{s}_\theta & 0 & 0 \\ +\mathrm{s}_\theta & +\mathrm{c}_\theta & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & +\mathrm{c}_\alpha & -\mathrm{s}_\alpha & 0 \\ 0 & +\mathrm{s}_\alpha & +\mathrm{c}_\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} \mathrm{c}_\theta & -\mathrm{s}_\theta\mathrm{c}_\alpha & \mathrm{s}_\theta\mathrm{s}_\alpha & a\mathrm{c}_\theta \\ \mathrm{s}_\theta & \mathrm{c}_\theta\mathrm{c}_\alpha & -\mathrm{c}_\theta\mathrm{s}_\alpha & a\mathrm{s}_\theta \\ 0 & \mathrm{s}_\alpha & \mathrm{c}_\alpha & d \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

cascading all these elemental transformations in the proper sequence.

The four DH parameters represents only one of the many possible choices to bind geometrically link frames between each other: the orientation and position constraints described above are respectively rendered by two angular parameters, θ and α, and by two linear parameters, d and a, even though the position is affected by both groups

of parameters, as the elemental transformations are not decoupled. In case of revolute joints, the parameter θ is the joint angle, so it is actually a variable rather than a fixed parameter; this observation would leave only three out of four kinematic parameters prescribed by the requirement of completeness. Nonetheless, as mentioned before, the joint displacement may be affected by a static error, i.e. a geometric bias present all over the work-space, irrespective of the actual posture. For this reason, the joint angle is usually decomposed in two contributions: a *variable* term, accounting for the angular displacement measured by the incremental transducer, and a constant term, accounting for the initial angular position. The total joint angle is then given by

$$\theta \coloneqq \theta_{\mathrm{o}} + \theta_{\mathrm{r}}$$

where θ_{o} is the joint angle offset and θ_{r} is the joint angle reading; the former is the actual parameter included in the kinematic model liable to correction via calibration.

As pointed out in subsection 2.3.4, two situations cannot be handled by means of the DH convention rules. If two consecutive motion axes intersect, all DH parameters are completely defined, as the common normal degenerates into a single point, then the common normal distance is zero and the direction of x axis is defined by the cross product constraint. In contrast, if two consecutive motion axes are parallel, there exist *infinite* segments orthogonal to both axes, thus the common normal is actually undefined.

In such a case, it is typically up to the designer or the user make a choice; the offset distance along the motion axis automatically comes out once a certain common normal is selected. As there are no reasonable preference criteria, it is customary to set the common normal in such a way that no translation along z is required, i.e. $d = 0$. Although such a choice proves very convenient for the simplification of the kinematic function, for it does not add any arbitrary complication, it fails to reflect the robot geometry: even the *slightest* relative misalignment between motion axes, making them barely skew, would yield a well defined and unique common normal, possibly very far from the one conventionally set.

This issue has a dramatic practical repercussion on the numerical identification procedure: at the beginning, the joint motion axes are assumed to be *exactly* parallel, as per design, and the nominal DH parameters are used, i.e. zero distance offset and common normal length equal to the length of the link between the joints; because the joint motion axes are lightly slanted, they might almost intersect far away from the physical location of the joints, hence the offset distance d suddenly becomes very large and the common normal length a suddenly becomes very small. More in general, in this situation, the numeric value of all DH parameters may be subjected to a sharp jump discontinuity. Conversely, when two non parallel joint motion axes become near parallel, the location of the common normal grows *indefinitely*. Tiny modifications of the manipulator geometry reflects in huge modifications of the kinematic parameters, therefore the proportionality is lost when using DH parametric transformation for near parallel axes. Mathematically speaking, parallelism of adjacent joint motion axes is the singularity of Denavit-Hartenberg parametrisation.

Even when the manipulator design involves exact parallelism, consecutive joint axes

are *virtually* never parallel. All 6DOF articulated robots always feature two nominally parallel motion axes, namely the second and the third, among all nominally perpendicular motion axes. This is the reason why the DH kinematic parametrisation cannot be universally applied to the whole kinematic chain and a different parametrisation, able to handle parallel axes while satisfying all model requirements, has to be found.

Geometric Insight

In order to understand geometrically the issue just mentioned, two generic skew and non orthogonal joint motion axes are considered. The unit vectors $\boldsymbol{u}$, bmv and $\boldsymbol{n}$ respectively represents the directions of the first motion axis, the second motion axis and the common normal. The second motion axis is the line with equation $\boldsymbol{r} = \boldsymbol{p} + t\boldsymbol{v}$, where $\boldsymbol{p}$ is the intersection between the axis and the plane $\boldsymbol{r} \cdot \boldsymbol{u} = 0$ perpendicular to the first motion axis and passing through the origin of the first link frame.

The origin of the second link frame may be reached, starting from the origin of the first link frame, through a couple of translations, in two different ways: moving first to $\boldsymbol{p}$ and then sliding along $\boldsymbol{v}$, or, sliding first along $\boldsymbol{u}$ and then moving on $\boldsymbol{n}$, giving

$$d\boldsymbol{u} + a\boldsymbol{n} = \boldsymbol{p} + t\boldsymbol{v} \tag{4.1}$$

where d is the displacement along the first motion axis, a is the distance between the two motion axes on the common normal, t is the displacement along the second motion axis. Since $\boldsymbol{n}$ is the common normal direction, then $\boldsymbol{n} \cdot \boldsymbol{u} = \boldsymbol{n} \cdot \boldsymbol{v} = 0$, by definition. Performing the dot product of both sides of eq. (4.1), first with the unit vector $\boldsymbol{u}$

$$d\boldsymbol{u} \cdot \boldsymbol{u} + a\boldsymbol{n} \cdot \boldsymbol{u} = \boldsymbol{p} \cdot \boldsymbol{u} + t\boldsymbol{v} \cdot \boldsymbol{u} \quad \Leftrightarrow \quad d = (\boldsymbol{u} \cdot \boldsymbol{v})\, t$$

then with the unit vector $\boldsymbol{v}$

$$d\boldsymbol{u} \cdot \boldsymbol{v} + a\boldsymbol{n} \cdot \boldsymbol{v} = \boldsymbol{p} \cdot \boldsymbol{v} + t\boldsymbol{v} \cdot \boldsymbol{v} \quad \Leftrightarrow \quad d\,(\boldsymbol{u} \cdot \boldsymbol{v}) = \boldsymbol{p} \cdot \boldsymbol{v} + t$$

and substituting the former into the latter multiplied by $\boldsymbol{u} \cdot \boldsymbol{v}$

$$d\,(\boldsymbol{u} \cdot \boldsymbol{v})^2 = (\boldsymbol{p} \cdot \boldsymbol{v})\,(\boldsymbol{u} \cdot \boldsymbol{v}) + t\,(\boldsymbol{u} \cdot \boldsymbol{v}) = (\boldsymbol{p} \cdot \boldsymbol{v})\,(\boldsymbol{u} \cdot \boldsymbol{v}) + d$$

gives in the end the expression of the offset distance

$$d = \frac{(\boldsymbol{p} \cdot \boldsymbol{v})\,(\boldsymbol{u} \cdot \boldsymbol{v})}{(\boldsymbol{u} \cdot \boldsymbol{v})^2 - 1}$$

which, as $\boldsymbol{u} \cdot \boldsymbol{v} = \|\boldsymbol{u}\|\,\|\boldsymbol{v}\| \cos\varphi = \cos\varphi$ and $\boldsymbol{p} \cdot \boldsymbol{v} = \|\boldsymbol{p}\|\,\|\boldsymbol{v}\| \cos\left(\frac{\pi}{2} \pm \varphi\right) = \mp\,\|\boldsymbol{p}\| \sin\varphi$, may be also expressed as

$$d = \frac{\mp\,\|\boldsymbol{p}\| \sin\varphi \cos\varphi}{\cos^2\varphi - 1} = \frac{\mp\,\|\boldsymbol{p}\| \sin\varphi \cos\varphi}{-\sin^2\varphi} = \pm\frac{\|\boldsymbol{p}\|}{\tan\varphi}$$

like a function of the angle $\varphi \coloneqq |\alpha| \in [0, \pi]$ between the joint motion axes. Since

$$\lim_{\boldsymbol{v} \to \pm \boldsymbol{u}} |d| = \lim_{\varphi \to 0, \pi} \frac{\|\boldsymbol{p}\|}{\tan \varphi} = \infty$$

then, as the the direction of the second joint motion axis approaches the direction of the first joint motion axis, the length of this distance grows, becoming infinitely large as the axes become either parallel ($\varphi = 0$) or anti parallel ($\varphi = \pi$).

Near Parallel Joint Motion Axes: Hayati-Mirmirani

The Hayati-Mirmirani convention is a very simple and elegant workaround, skilfully described in [10], which exploits a minor modification of the Denavit-Hartenberg convention, in order to accommodate parallel joint motion axes without giving rise to parametric singularities. As exemplified in subsection 2.3.4, the prime obstacle of Denavit-Hartenberg formalism is the strict reliance on the concept of common normal; since the identification of such an abstract geometric entity is not possible for parallel axes, and anyway *practically* infeasible for near parallel axes, the most reasonable solution consists in dropping this abstract concept completely.

The Hayati-Mirmirani formalism relies instead on the normal plane, i.e. the xy-plane perpendicular to the first joint motion axis and passing through the origin. A rotation about z orients x axis with the segment joining the two motion axes on the normal plane, then a translation along x axis brings the origin of the reference frame to the intersection between the normal plane and the next motion axis, finally a rotation about x, followed by a rotation about y, aligns z axis with the next motion axis. The HM homogeneous transformation matrix is thus given by

$$\begin{aligned} \boldsymbol{T}_{\mathrm{HM}} \coloneqq & \begin{pmatrix} +\mathrm{c}_\theta & -\mathrm{s}_\theta & 0 & 0 \\ +\mathrm{s}_\theta & +\mathrm{c}_\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & +\mathrm{c}_\alpha & -\mathrm{s}_\alpha & 0 \\ 0 & +\mathrm{s}_\alpha & +\mathrm{c}_\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} +\mathrm{c}_\beta & 0 & +\mathrm{s}_\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\mathrm{s}_\beta & 0 & +\mathrm{c}_\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \\ = & \begin{pmatrix} \mathrm{c}_\theta \mathrm{c}_\beta - \mathrm{s}_\theta \mathrm{s}_\alpha \mathrm{s}_\beta & -\mathrm{s}_\theta \mathrm{c}_\alpha & \mathrm{c}_\theta \mathrm{s}_\beta + \mathrm{s}_\theta \mathrm{s}_\alpha \mathrm{c}_\beta & a\mathrm{c}_\theta \\ \mathrm{s}_\theta \mathrm{c}_\beta + \mathrm{c}_\theta \mathrm{s}_\alpha \mathrm{s}_\beta & \mathrm{c}_\theta \mathrm{c}_\alpha & \mathrm{s}_\theta \mathrm{s}_\beta - \mathrm{c}_\theta \mathrm{s}_\alpha \mathrm{c}_\beta & a\mathrm{s}_\theta \\ -\mathrm{c}_\alpha \mathrm{s}_\beta & \mathrm{s}_\alpha & \mathrm{c}_\alpha \mathrm{c}_\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{aligned}$$

cascading all these elemental transformations in the proper sequence.

Quite evidenty, the HM transformation is almost the same as the DH transformation, except it misses a translation along z at the beginning and it adds a rotation about y at the end. The reason behind such modifications is pretty easy to understand: the DH sequence needs the first translation along z in order to reach the location of the common normal, while the HM sequence works *directly* on the normal plane $z = 0$; the DH sequence directs x axis along the common normal, thus ensuring that both z axis and the next joint motion axis lie on a plane perpendicular to it, just a simple rotation about x suffices to make them overlap, whereas the HM sequence aligns x axis on the

normal plane towards the intersection with the second motion axis, hence a rotation about x may bring z axis on the plane perpendicular to y, so that a further rotation about y is required to align z axis with the next joint motion axis.

The four HM parameters are an alternative choice for defining the geometric constraints required by kinematic transformations. Remembering the discussion about the joint angle offset contribution, HM transformation provides four independent kinematic parameters too, in compliance with completeness requirement. Besides, the transformed link frame is again consistent with the conventional rules given before, as its origin gets placed on the motion axis and its z axis gets oriented with the motion axis. Thanks to the absence of the common normal, no *abrupt* parameters changes occur if it emerges that two consecutive joint motion axes are not exactly parallel, hence this parametrisation restores the missing proportionality requirement.

Anyhow, HM parameters are just a replacement of DH parameters for near parallel joint motion axes and cannot be applied to the whole kinematic chain *either*. In a complementary way to DH parametrisation, there exists a situation that cannot be handled by HM parametrisation. If two motion axes are perpendicular, the second one is parallel the normal plane, so there are either infinite or no points in common, depending on whether or not the second axis lies on the plane. Even when that axis is embedded in the normal plane, no unique intersection point can be identified, and, without entering too much into the detail, an arbitrary selection would lead to the same numerical problems discussed before about DH transformation between parallel motion axes. Mathematically speaking, the perpendicularity of adjacent joint motion axes is the singularity of Hayati-Mirmirani parametrisation.

End Frame

As mentioned before when discussing model completeness, the end frame is not constrained whatsoever, because there is no joint at the distal end of the last link, therefore the last transformation requires six parameters. In principle, any sequence of six independent elemental transformations may be employed to model the relationship between the last two link frames; however, since the z axis of the second to last link frame coincides with the last joint motion axis, then only those sequences of transformations that begin with a rotation about z axis may be taken into consideration, for the motion of the last joint must be represented by a single variable, which is indeed the reason why that link frame is thus constrained.

Considering that a complete rotation and a complete translation must be included, the end transformation might consist of three independent rotations, such as z-x-z, z-y-z, z-y-x, followed by three independent translations, the order of which is irrelevant. On the other hand, even though the end frame does not have to meet any particular constraints given by kinematic conventions, when the nominal model of the whole kinematic chain relies on a particular formalism, the last group of parameters might be saved and still used, simply by adding other two parameters to complete the transformation.

As the robot is entirely modelled using the DH convention, a further rotation about y and a further translation along y are appended to the last transformation. The end

homogeneous matrix is thus given by

$$\boldsymbol{T}_{\mathrm{E}} \coloneqq \begin{pmatrix} +\mathrm{c}_\theta & -\mathrm{s}_\theta & 0 & 0 \\ +\mathrm{s}_\theta & +\mathrm{c}_\theta & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & +\mathrm{c}_\alpha & -\mathrm{s}_\alpha & 0 \\ 0 & +\mathrm{s}_\alpha & +\mathrm{c}_\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} +\mathrm{c}_\beta & 0 & +\mathrm{s}_\beta & 0 \\ 0 & 1 & 0 & b \\ -\mathrm{s}_\beta & 0 & +\mathrm{c}_\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} \mathrm{c}_\theta\mathrm{c}_\beta - \mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{s}_\beta & -\mathrm{s}_\theta\mathrm{c}_\alpha & \mathrm{c}_\theta\mathrm{s}_\beta + \mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{c}_\beta & a\mathrm{c}_\theta - b\mathrm{s}_\theta\mathrm{c}_\alpha \\ \mathrm{s}_\theta\mathrm{c}_\beta + \mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{s}_\beta & \mathrm{c}_\theta\mathrm{c}_\alpha & \mathrm{s}_\theta\mathrm{s}_\beta - \mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{c}_\beta & a\mathrm{s}_\theta + b\mathrm{c}_\theta\mathrm{c}_\alpha \\ -\mathrm{c}_\alpha\mathrm{s}_\beta & \mathrm{s}_\alpha & \mathrm{c}_\alpha\mathrm{c}_\beta & d + b\mathrm{s}_\alpha \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

cascading all these elemental transformations in the proper sequence.

4.2.3 Kinematic Singularities

A more rigorous analysis about the singularities of the proposed kinematic parametrisations can be laid out recollecting the mathematical definition of proportionality given in subsection 3.3.1: a parametric transformation is said to be proportional if the parametrisation is regular. The regularity of a parametric function can be investigated by checking the rank of the Jacobian matrix of the function with respect to the parameters: the values of parameters such that the Jacobian matrix loses rank give the singularities of the parametric function.

As clarified above, the kinematic transformations are defined up to another two more degrees the freedom, because the position of the origin along the motion axis, and the orientation of the x and y axes about the motion axis, are not set. There are two possibilities to bypass this obstacle: the first one consists in taking into account only the elements actually constrained by the parametric transformation, that is to say the direction of z axis and the coordinates of the position of the origin on a plane perpendicular to z, which are four quantities indeed; the second one consists in adding two other independent transformations in order to achieve an *arbitrary* complete kinematic transformation. The latter approach is here followed.

Denavit-Hartenberg Singularity

The DH transformation may be completed by adding another translation along z and another rotation about z. The resulting frame is thus described by the homogeneous transformation matrix

$$\begin{aligned} &\mathbf{Trans}\,(\boldsymbol{k}, d)\,\mathbf{Rot}\,(\boldsymbol{k}, \theta)\,\mathbf{Trans}\,(\boldsymbol{i}, a)\,\mathbf{Rot}\,(\boldsymbol{i}, \alpha)\,\mathbf{Trans}\,(\boldsymbol{k}, t)\,\mathbf{Rot}\,(\boldsymbol{k}, \gamma) = \\ &= \begin{pmatrix} \boldsymbol{R}(\boldsymbol{k}, \theta)\,\boldsymbol{R}(\boldsymbol{i}, \alpha)\,\boldsymbol{R}(\boldsymbol{k}, \gamma) & d\boldsymbol{k} + a\boldsymbol{R}(\boldsymbol{k}, \theta)\,\boldsymbol{i} + t\boldsymbol{R}(\boldsymbol{k}, \theta)\,\boldsymbol{R}(\boldsymbol{i}, \alpha)\,\boldsymbol{k} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} = \\ &= \begin{pmatrix} \boldsymbol{R}(\boldsymbol{k}, \theta)\,\boldsymbol{R}(\boldsymbol{i}, \alpha)\,\boldsymbol{R}(\boldsymbol{k}, \gamma) & (a\mathrm{c}_\theta + t\mathrm{s}_\theta\mathrm{s}_\alpha)\,\boldsymbol{i} + (a\mathrm{s}_\theta - t\mathrm{c}_\theta\mathrm{s}_\alpha)\,\boldsymbol{j} + (d + t\mathrm{c}_\alpha)\,\boldsymbol{k} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} \end{aligned}$$

or, alternatively, by the pose vector

$$\boldsymbol{p} = \begin{pmatrix} \boldsymbol{r} \\ \boldsymbol{\phi} \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ \theta \\ \alpha \\ \gamma \end{pmatrix} = \begin{pmatrix} a\mathrm{c}_\theta + t\mathrm{s}_\theta\mathrm{s}_\alpha \\ a\mathrm{s}_\theta - t\mathrm{c}_\theta\mathrm{s}_\alpha \\ d + t\mathrm{c}_\alpha \\ \theta \\ \alpha \\ \gamma \end{pmatrix}$$

where $\boldsymbol{r}$ is the Cartesian position of the origin and $\boldsymbol{\phi}$ includes the z-x-z Euler angles giving the orientation of the coordinate axes. Stacking all parameters into a single vector

$$\boldsymbol{\eta} \coloneqq \begin{pmatrix} d \\ a \\ t \\ \theta \\ \alpha \\ \gamma \end{pmatrix}$$

makes the pose a vector valued function of a vector variable. Its Jacobian matrix

$$\boldsymbol{J}_{\boldsymbol{\eta}}\boldsymbol{p} = \begin{pmatrix} 0 & \mathrm{c}_\theta & +\mathrm{s}_\theta\mathrm{s}_\alpha & -a\mathrm{s}_\theta + t\mathrm{c}_\theta\mathrm{s}_\alpha & +t\mathrm{s}_\theta\mathrm{c}_\alpha & 0 \\ 0 & \mathrm{s}_\theta & -\mathrm{c}_\theta\mathrm{s}_\alpha & +a\mathrm{c}_\theta + t\mathrm{s}_\theta\mathrm{s}_\alpha & -t\mathrm{s}_\theta\mathrm{c}_\alpha & 0 \\ 1 & 0 & \mathrm{c}_\alpha & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

is rank deficient when the determinant is zero

$$\det \boldsymbol{J}_{\boldsymbol{\eta}}\boldsymbol{p} = -\mathrm{c}_\theta^2\mathrm{s}_\alpha - \mathrm{s}_\theta^2\mathrm{s}_\alpha = -\mathrm{s}_\alpha = 0 \;\Leftrightarrow \alpha = 0 \vee \pi$$

that is to say when the two joint motion axes are parallel. In this case, the homogeneous transformation matrix degenerates into

$$\begin{pmatrix} \boldsymbol{R}(\boldsymbol{k}, \theta \pm \gamma) & a\mathrm{c}_\theta\boldsymbol{i} + a\mathrm{s}_\theta\boldsymbol{j} + (d \pm t)\,\boldsymbol{k} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}$$

hence the two rotations about z and two translations along z are not decoupled anymore, making two parameters redundant.

Hayati-Mirmirani Singularity

The HM transformation may be completed by adding another translation along z at the beginning and another translation along y at the end. The resulting frame is thus

described by the homogeneous transformation matrix

$$\begin{aligned}
&\mathbf{Trans}\,(\boldsymbol{k},d)\,\mathbf{Rot}\,(\boldsymbol{k},\theta)\,\mathbf{Trans}\,(\boldsymbol{i},a)\,\mathbf{Rot}\,(\boldsymbol{i},\alpha)\,\mathbf{Rot}\,(\boldsymbol{j},\beta)\,\mathbf{Trans}\,(\boldsymbol{j},b) = \\
&= \begin{pmatrix} \boldsymbol{R}(\boldsymbol{k},\theta)\,\boldsymbol{R}(\boldsymbol{i},\alpha)\,\mathbf{Rot}\,(\boldsymbol{j},\beta) & d\boldsymbol{k} + a\boldsymbol{R}(\boldsymbol{k},\theta)\,\boldsymbol{i} + b\boldsymbol{R}(\boldsymbol{k},\theta)\,\boldsymbol{R}(\boldsymbol{i},\alpha)\,\boldsymbol{j} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix} = \\
&= \begin{pmatrix} \boldsymbol{R}(\boldsymbol{k},\theta)\,\boldsymbol{R}(\boldsymbol{i},\alpha)\,\boldsymbol{R}(\boldsymbol{j},\beta) & (a\mathrm{c}_\theta - b\mathrm{s}_\theta\mathrm{c}_\alpha)\,\boldsymbol{i} + (a\mathrm{s}_\theta + b\mathrm{c}_\theta\mathrm{c}_\alpha)\,\boldsymbol{j} + (d + b\mathrm{s}_\alpha)\,\boldsymbol{k} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix}
\end{aligned}$$

or, alternatively, by the pose vector

$$\boldsymbol{p} = \begin{pmatrix} \boldsymbol{r} \\ \boldsymbol{\phi} \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ \theta \\ \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} a\mathrm{c}_\theta - b\mathrm{s}_\theta\mathrm{c}_\alpha \\ a\mathrm{s}_\theta + b\mathrm{c}_\theta\mathrm{c}_\alpha \\ d + b\mathrm{s}_\alpha \\ \theta \\ \alpha \\ \beta \end{pmatrix}$$

where $\boldsymbol{r}$ is the Cartesian position of the origin and $\boldsymbol{\phi}$ includes the z-x-y Cardan angles giving the orientation of the coordinate axes. Stacking all parameters into a single vector

$$\boldsymbol{\eta} \coloneqq \begin{pmatrix} d \\ a \\ t \\ \theta \\ \alpha \\ \gamma \end{pmatrix}$$

makes the pose a vector valued function of a vector variable. Its Jacobian matrix

$$\boldsymbol{J}_{\boldsymbol{\eta}}\boldsymbol{p} = \begin{pmatrix} 0 & \mathrm{c}_\theta & -\mathrm{s}_\theta\mathrm{c}_\alpha & -a\mathrm{s}_\theta - b\mathrm{c}_\theta\mathrm{c}_\alpha & +b\mathrm{s}_\theta\mathrm{s}_\alpha & 0 \\ 0 & \mathrm{s}_\theta & +\mathrm{c}_\theta\mathrm{c}_\alpha & +a\mathrm{c}_\theta - b\mathrm{s}_\theta\mathrm{c}_\alpha & -b\mathrm{c}_\theta\mathrm{s}_\alpha & 0 \\ 1 & 0 & \mathrm{c}_\alpha & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

is rank deficient when the determinant is zero

$$\det \boldsymbol{J}_{\boldsymbol{\eta}}\boldsymbol{p} = \mathrm{c}_\theta^2\mathrm{c}_\alpha + \mathrm{s}_\theta^2\mathrm{c}_\alpha = \mathrm{c}_\alpha = 0 \;\Leftrightarrow \alpha = \pm\frac{\pi}{2}$$

that is to say when the two joint motion axes are perpendicular. In this case, the homogeneous transformation matrix degenerates into

$$\begin{pmatrix} \boldsymbol{R}(\boldsymbol{k},\theta \pm \beta)\,\boldsymbol{R}\big(\boldsymbol{i},\pm\frac{\pi}{2}\big) & a\mathrm{c}_\theta\boldsymbol{i} + a\mathrm{s}_\theta\boldsymbol{j} + (d \pm b)\,\boldsymbol{k} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{pmatrix}$$

hence the rotations about z and y and the translations along z and y are not decoupled anymore, making two parameters redundant.

4.2.4 Kinematic Parameters

No modelling had been already developed and no kinematic parameters were already available for the robot manipulator unit under calibration. The manufacturer provided only a schematics reporting the values of the most relevant robot links dimensions, shown in fig. 4.3. Basing on these raw dimensional values, a nominal kinematic model has been created from scratch, following the rules of Denavit-Hartenberg convention.

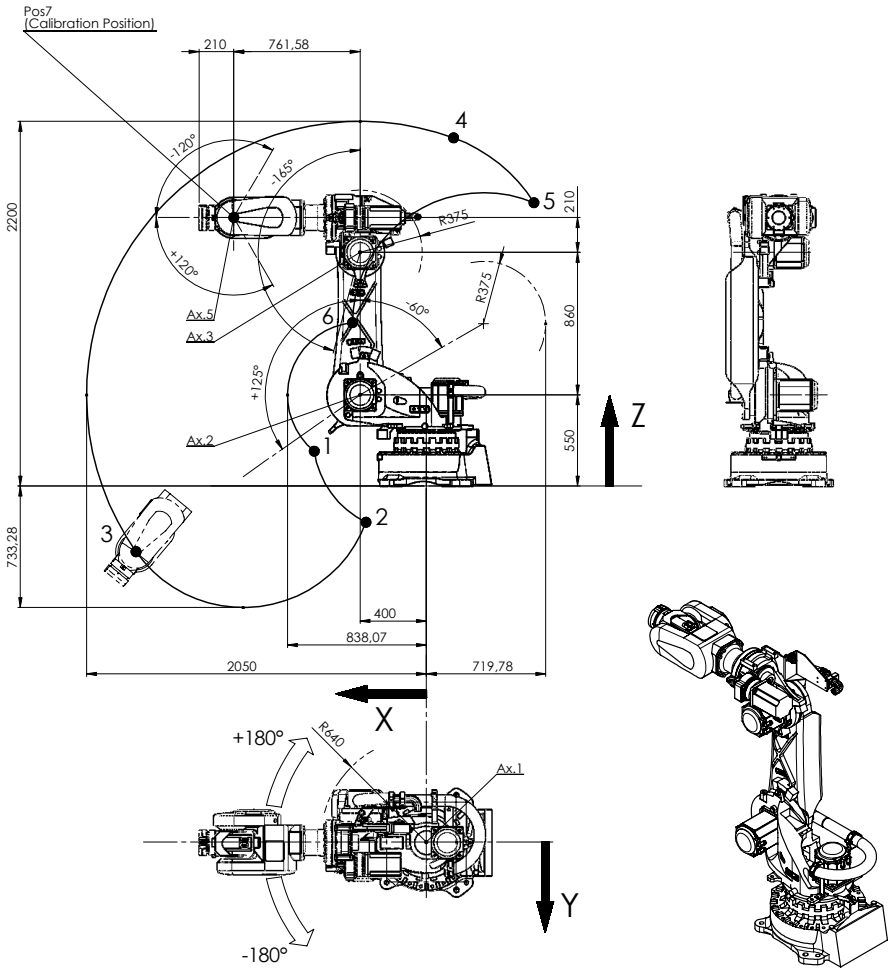

Figure 4.3: COMAU NJ 130-2.0 links size and workspace volume.

As extensively discussed above, for calibration purposes, DH parameters are not suitable for the entire kinematic chain, hence the model structure has been properly modified, adding and removing parameters according to the fundamental rules of proportionality and completeness. In the end, the complete and proportional target kinematic model features a total of thirty kinematic parameters, arranged as follows: Denavit-Hartenberg kinematic parameters are used for $\boldsymbol{T}_1^0$, $\boldsymbol{T}_3^2$, $\boldsymbol{T}_4^3$ and $\boldsymbol{T}_5^4$, kinematic Hayati-Mirmirani pa-

rameters are used for $\boldsymbol{T}_0^{\mathrm{B}}$ and $\boldsymbol{T}_2^1$, and six kinematic parameters are used for $\boldsymbol{T}_6^5$. The numeric values of these parameters are displayed table 4.1.

j	$\theta_{\mathrm{o}}\,(\mathrm{rad})$	$d\,(\mathrm{m})$	$\alpha\,(\mathrm{rad})$	$a\,(\mathrm{m})$	$\beta\,(\mathrm{rad})$	$b\,(\mathrm{m})$
0	$-\pi/2$	\	π	-2.000	0	\
1	0	-0.550	$+\pi/2$	0.400	\	\
2	$-\pi/2$	\	π	0.860	0	\
3	$+\pi/2$	0	$-\pi/2$	0.210	\	\
4	0	-0.76158	$-\pi/2$	0	\	\
5	0	0	$+\pi/2$	0	\	\
6	π	-0.210	π	0	0	0

Table 4.1: Nominal Kinematic Parameters for Complete and Proportional Model.

4.2.5 Thermal Deformation

As anticipated in subsection 3.2.4, evaluating the detrimental effect of temperature over manipulator kinematics is a quite challenging task. The ambient temperature of the work cell displays rather small variations hence it is expected to play a minor role in thermal error generation. On the contrary, the effect of the body temperature is far more critical for at least two reasons: quite far points of the robot shell may be at totally different temperature because self heating takes place not uniformly; the temperatures of the robot frame may range over fairly large intervals because the joint motors are really strong thermal sources. The inconsistency between ambient temperature and body temperatures denotes a state of thermal imbalance inducing heat flow between the robot manipulator and the surrounding environment, which may occur via conduction through the floor, convection through air and radiation through infra-red emission.

An exact quantitative analysis of the robot manipulator thermal behaviour is nearly infeasible, for it would require to know shapes, sizes and materials of every single robot component, and process this huge amount of information through an immensely powerful simulation environment able to replicate virtually the thermal phenomena that take place in the robot manipulator work-cell, exploiting the thermal properties of materials and the laws of thermodynamics. On the other hand, even if it were possible, meticulously examining every thermal phenomenon regarding robot manipulators would be of very little practical help in the scope of volumetric accuracy compensation, since the analysis of manipulator kinematics essentially revolves around the construction of an abstract geometrical model featuring a set of conventional parameters the value of which is numerically assigned and may or may not have a strict physical equivalent.

This is the reason why it is certainly preferable to investigate the effect of temperature dynamics on manipulator kinematics indirectly, that is to say by checking and evaluating the relationship between the kinematic parameters and the thermal gradient. All links, in general, are subject to thermal deformation, therefore, at least in theory, all kinematic parameters might vary owing to temperature gradient time evolution. In

practice, only some of them, i.e. those with a more definite physical meaning, actually display a significant correlation with temperatures changes: more specifically, uniform thermal variations induce link elongation *only*, resulting in a modification of linear parameters, whereas non uniform thermal variations induce link bending *too*, resulting in a modification of angular parameters.

Bearing in mind the simplified law of thermal expansion[1], the kinematic parameters are reasonably expected to vary in a roughly linear fashion with respect to the temperature rise. In more formal terms, the thermally induced error in the j-th kinematic parameter might be expressed approximately as

$$\Delta k_j \approx \gamma_{j1}\Delta T_1 + \ldots + \gamma_{jn_S}\Delta T_{n_S} \tag{4.2}$$

where $\Delta T_1, \ldots, \Delta T_{n_S}$ are the temperature changes of n_S relevant robot points.

4.3 Measurement Set-up

The experimental set-up for the designed kinematic compensation strategy of the target robot manipulator unit consists of a 3D position measurement system, the standard equipment required for geometric error compensation, and a real time temperature monitor, the additional hardware needed for thermal error compensation. Within the limits of available precision, the set of experimental data collected in this step represents a snapshot of the actual kinematics of the robot manipulator, that is indeed at the very heart of the concept of calibration.

The robot equips a tool plate holding four **optical targets**, whose 3D Cartesian position is accurately measured, at each flange pose, by means of a **laser tracker**. As said earlier, the position of *at least* three tool points per joint posture should be measured, in order to incorporate virtually the information of the complete end effector pose; four targets are here employed to honour the quadrangular geometry of the tool.

As reasonably prescribed in subsection 3.3.2, the mechanical tolerances of the tool components and the measurement performances of the laser tracker have to be far better than the level of position accuracy provided by the end effector, otherwise the success of calibration would be inexorably compromised. Furthermore, for the same reason, the tool plate is entirely constructed of aluminium, in order to limit the overall weight of the tool burdening on the robot wrist as much as possible.

The temperatures distribution across the robot body is monitored by means of an array of five micro-controller boards, anchored respectively to the five largest links, each of which collects and sends via Wi-Fi the data of four temperature transducers attached to some selected spots of the link armour. Such a wireless configuration has been preferred for the temperature data acquisition system, because the manipulator is required to move vigorously, in order to stress adequately the joints and thus engender enough heat. Since all adjacent links experience relative motion between each other, wiring all

[1] A beam of length l, when subjected to a small temperature change ΔT, shows a relative strain given by $\frac{\Delta l}{l} = \alpha \Delta T$, where α is the coefficient of thermal expansion of the material the beam is made of.

the sensors to a common data acquisition board would be a definitely uncomfortable and hazardous solution.

4.3.1 Tool Targets

The tool installed on the end effector of the robot manipulator under study is a base mounting surface clutching the optical targets, namely four corner cube retroreflectors (CCR). The whole mounting surface is composed of two elements anchored together: a round plate bolted to the robot flange and a square plate, bound to the targets.

The square tool hosts four target holders in its sides and four round pins in its vertices. Each reflective target is fastened to a pin, that serves as sort of spherical joint, allowing the rotation of the target with respect to a fixed pivot: such a degree of freedom is very important to orient properly all the four retroreflectors towards a common point, that is the source of the beam emitted by the optical measuring instrument.

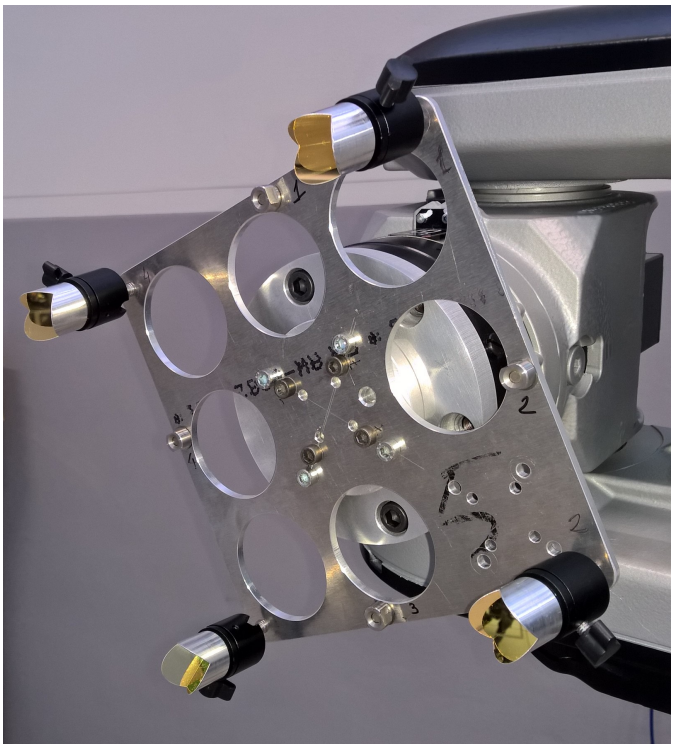

Figure 4.4: Tool plate with Corner Cube Retroreflectors.

The tool plate has been geometrically calibrated in advance, probing, by means of a coordinate measuring machine, the position of the four target holders with respect to some mechanical references on the mounting interface, that define the coordinate frame of the robot flange. Once the tool has been firmly installed, by means of the round plate interface, on the robot flange, and the orientation of all targets has been set, by tightly

fastening the appropriate screw knobs, then the position of the actual reflective targets in the end effector frame has to be determined.

To this aim, a further optical target, namely a $0.5''$ spherically mounted retroreflector (SMR), is magnetically attached, in sequence, to the four target holders, and their position in the absolute coordinate frame is measured through the laser tracker; then the flange coordinate frame can be recreated, from the knowledge of the target holders position in that frame, already evaluated by means of the CMM. Eventually, the *actual* Cartesian position of the tool targets, i.e. the centre of the retroreflectors, can be accurately determined in the world frame by the laser tracker and then transformed into the flange frame, thanks to the relationship found before.

The round plate mounted on the robot flange is also endowed with some dedicated holes, designed to host suitable dowel pins, allowing to place, remove or replace the tool plate on the robot wrist, without modifying *significantly* their relative position and orientation. Such a sophisticated mechanical interface is of paramount importance, not only for the calibration process *per se*, but also to take advantage of the volumetric accuracy enhancement provided by calibration, in different practical operating contexts, with the robot manipulator equipping other application specific tools.

4.3.2 3D Coordinate Measurement

A FARO® ION, a high precision interferometer-based laser tracker, is employed to read the position of the tool targets in the work-space; it features a resolution of 0.5µm and an accuracy (half of the maximum permissible error) of $16\mu m + 0.8\mu m/m$.

Figure 4.5: FARO® Laser Tracker ION.

The flexible measurement software platform SpatialAnalyzer®, able to interface *con-*

currently with multiple instruments and integrate their data together, is entrusted with the synchronisation of the laser tracker with the robot controller. Whenever the robot manipulator is stopped at a given desired pose, the laser tracker is instructed to record accurately the Cartesian position of the four reflecting targets on the flange.

All joint postures are roughly selected so that the flange plate faces the laser tracker column, which lies approximately in front of the robot base. Furthermore, basing on the nominal kinematic model of the robot manipulator, the measurement software environment processes each flange pose and generates the corresponding joint posture, in such a way that the four targets are correctly pointed at the laser tracker *simultaneously*, that is to say the laser beam is able to reach all of the corner retroreflectors.

The position measurement of each 3D point is computed by averaging multiple repeated readings of that point, in order to minimise the random error due to instrument precision, environmental conditions, and robot residual vibrations. The laser tracker is set to record the position of 3D points with a sample rate of 100Hz, and 50 samples per point are collected, therefore each point measurement requires 0.5 sec. Averaging over 50 samples brings the residual RMS error down to about $1 \div 3\mu$m.

4.3.3 Temperature Monitor

The package of each temperature sensor board consists of a thermal transducer embedded within a digital integrated module featuring an I^2C serial communication interface: the main chip is a human body temperature sensor device integrating a 16 bit analog-to-digital converter, able to provide $0 \div 64$°C operating range, 0.3°C accuracy and 0.00390625°C resolution. Three dedicated external pins have to be properly connected to certain logic voltage levels, in order to assign the sensor board a specific address.

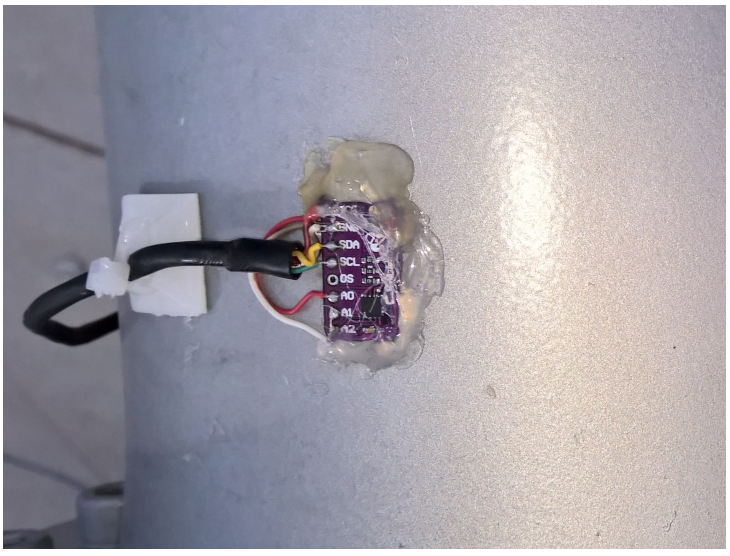

Figure 4.6: Thermal sensor board glued to a robot link.

All the thermal sensors adhering to selected spots of the same robot link are wired, by means of tailored strip headers, to a common breadboard hosting an Arduino® inspired development board equipped with an integrated 802.11 b/g/n protocol based Wi-Fi chip. A compact rechargeable lithium-ion battery powers the micro-controller board, which supplies in turn the power to the sensor boards connected to it.

Figure 4.7: Breadboard with connected micro-controller, wire terminals and power supply.

Each micro-controller has been programmed in order to query *sequentially* all thermal sensors through their specific I^2C address, decode their raw output and send the corresponding temperature data as numeric strings over the air, via a dedicated wireless local network. The computer tasked with all robot operations continuously scans all micro-controllers, logging the temperature readings coming from all the thermal sensors distributed throughout the robot framework.

It is worth noting that the adopted experimental set-up for monitoring temperatures provides great flexibility, thanks to its enhanced modularity: the whole temperature monitor indeed consists of an array of several identical modules, each one including a fixed number of temperature sensors wired to a common Wi-Fi capable board. If, for instance, some more thermal sensors were needed for sensing the temperature of the

robot itself or other equipment on the production line, other complete modules may be handily added, without altering the pre-existing configuration *whatsoever*.

4.3.4 Operations

A single computer is dedicated to all measurement operations concerning the calibration of the manipulator unit under study. It is entrusted with commanding the robot motion, instructing the laser tracker to measure the position of the tool targets and downloading the temperature data from the micro-controllers via Wi-Fi local network.

The complete test designed for thermal calibration consists of a warm up phase and cool down phase, each of which in turn composed of a measurement step and a movement step, repeated in alternating fashion. In the movement step, the robot manipulator continually performs a series of few simple repetitive movements, planned in the joint-space especially for exercising all joints. In the measurement step, the robot manipulator reaches the fifteen joint postures selected for calibration and the laser tracker measures the position of the tool targets at each corresponding flange pose.

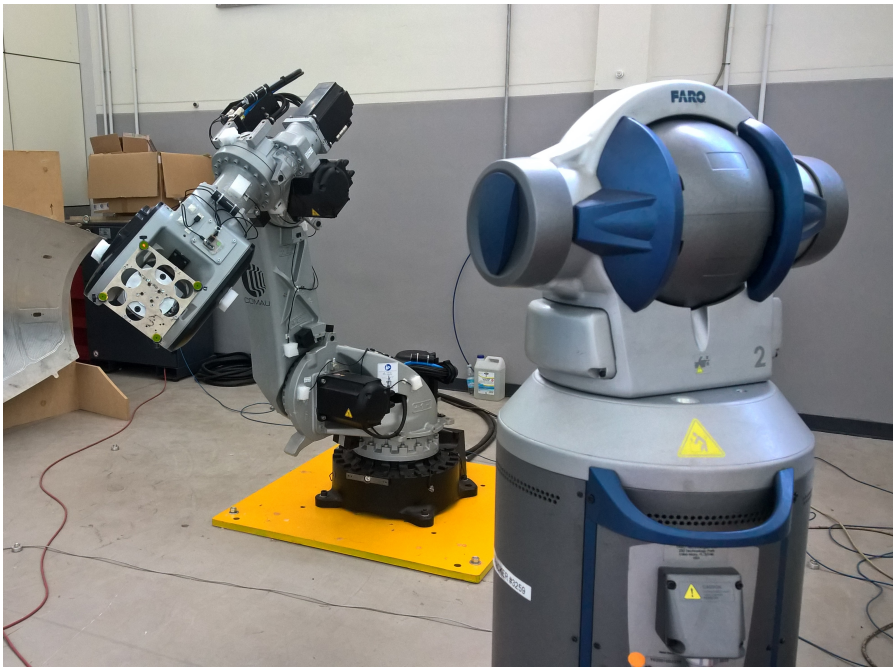

Figure 4.8: Laser tracker pointing at a tool target.

The two phases are qualitatively equal, what differentiates them is the speed override set for the robot joints. During warm up, because of the faster motion, the temperatures rise up to their upper steady state value, whereas during cool down, because of the slower motion, the temperatures drop down to their lower steady state value. Considering that, after a full cycle of warming and cooling, the temperatures return to the same

value, acquiring kinematic and thermal data during both stages may sound apparently superfluous. The time rate of change of robot temperatures can be rendered realistically by means of simple exponential rises and falls: they grow quickly at the beginning and slowly at the end, during warm up, whereas they drop quickly at the beginning and slowly at the end, during cool down. For this reason, performing measurements on both stages gives the possibility to sample the robot kinematics while spanning the entire temperature range with a good level of granularity.

After performing the first experimental tests, an interesting phenomenon has been observed: the temperature readings of most of the thermal transducers grew softly while the robot was fidgeting and sharply as the robot was stopping at the desired postures. This paradoxical behaviour may be explained by looking at the time trend of the temperature values yielded by three thermal sensors attached to links 0, 2 and 4. As evidently shown in fig. 4.9, no spikes are present in the temperature reading of the sensor attached to link 0, that is the robot base, therefore the phenomenon was likely related to the links motion.

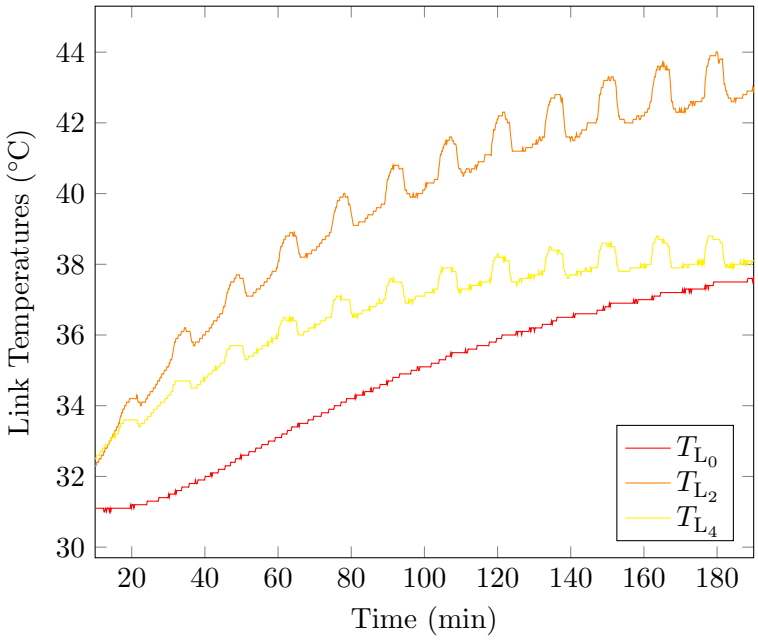

Figure 4.9: Spikes in readings of links temperature sensors.

Owing to the quite large thermal inertia of the links frame, the heat generated by motors and gears, albeit massive, does not get immediately translated in a corresponding temperature rise; on the other hand, the temperature sensors have instead fast thermal response, because of their very small sizes. For this reason, as long as the robot was moving quickly, the air flow generated directly chilled the sensor *themselves*, whereas, as soon as the robot became still again and the cooling effect of air ventilation was over, the transducers began to sense the *actual* links temperatures. Basing on such considerations,

this thermal issue has been addressed by covering all sensors with blocks made of foam rubber, properly sized and shaped in order not to impede any movement of the robot links, clearly visible in fig. 4.8: acting as insulators, the blocks prevent the sensor boards from being cooled faster than the robot frame.

Moreover, a more or less small pause has been inserted between a movement step and a measurement step of each cycle, in order to further damp any annoying ripple of the link temperatures during the whole acquisition of tool targets positions, which typically requires some minutes. The sum of the time required by the movement step and the time required by the measurement step including the pause, gives the full cycle time. Some minor corrections have been made at a later stage to extend the spread of the robot thermal gradient and, at the same time, to keep the temperature changes between consecutive cycles quite moderate. To this aim, both times and speeds have been finely sized: in the end, the cycle time has been set between 10 min and 20 min, while the speed override has been set between 50% and 100% for warm up and between 0% and 50% for cool down.

4.4 Numerical Identification

The whole calibration of the robot manipulator under study is carried out by means of a genuine program entirely written in MATLAB® software environment. The program is based on basic functions regarding matrices handling only, and no built-in MATLAB® functions or packages whatsoever have been employed; hence, by virtue of this choice, the application designed for calibration may be easily translated into any desired programming language and encapsulated into a larger software project as a separate library, according to the company desires.

The result of the measurement stage is a huge mass of heterogeneous data regarding the robot manipulator under calibration, namely the joints angular displacements encoder readings, the tool targets position laser tracker readings and the links temperatures thermometers readings. A supplementary routine is dedicated to scan all files and pack the measurement data they contain into several suitable structures with manageable format, for numerical processing purposes. The large data collection actually originates from sampling the same quantities in different moments, hence it is important to arrange these numeric structures in such a way that all groups of data referring to a given cycle may be easily indexed.

Each measurement cycle accounts for a different thermal condition, therefore putting the temperatures aside, the other data, which regard the robot kinematics only, are used to perform geometric calibration many times, and subsequently these results are used to perform also thermal calibration. Along the lines of what already outlined before, first the kinematic parameters variations are identified for each temperatures level, then the thermal coefficients are identified by fitting the kinematic parameters changes with respect to the corresponding test temperatures changes. The total error compensation is eventually achieved by modifying the *nominal* kinematic parameters using the data obtained from identification.

In more formal terms, gathering all kinematic parameters into a large vector, the calibrated value $\boldsymbol{k}$ is obtained from the nominal value $\boldsymbol{k}_n$ as

$$\boldsymbol{k} = \boldsymbol{k}_n + \Delta\boldsymbol{k}|_{\text{geom}} + \Delta\boldsymbol{k}|_{\text{therm}} \tag{4.3}$$

where $\Delta\boldsymbol{k}|_{\text{geom}}$ is the amount of correction needed for adherence to (cold) geometrical traits, while $\Delta\boldsymbol{k}|_{\text{therm}}$ is the amount of correction accounting for thermal alteration. Only the values of those parameters present in the kinematic model devised for calibration are indeed subject to amendment.

4.4.1 Kinematic Parameters Estimation

The first step of calibration is geometric error compensation consisting in the numerical estimation of the kinematic parameters; model identification is carried out as an optimisation problem, through the minimisation of the all target position errors over the set of the kinematic parameters: according to the reasoning made in section 5.2, the residual sum of squares of all position errors is chosen as performance index.

The i-th target position error is the difference

$$\Delta\boldsymbol{r}_i \coloneqq \overline{\boldsymbol{r}}_i - \boldsymbol{r}_i(\boldsymbol{k})$$

where $\overline{\boldsymbol{r}}_i$ is the i-th target position read by the laser tracker while $\boldsymbol{r}_i(\boldsymbol{k})$ is the t-tharget position given by the kinematic model. The loss function is then

$$L(\boldsymbol{k}) = \sum_{i=1}^{N} \frac{1}{2} \left\| \Delta\boldsymbol{r}_i(\boldsymbol{k}) \right\|^2$$

where $N = 4 \cdot 15 = 60$ is the total number of target positions. The gradient

$$\nabla L(\boldsymbol{k}) = -\sum_{i=1}^{N} \boldsymbol{J}\boldsymbol{r}_i(\boldsymbol{k})^{\mathsf{T}} \Delta\boldsymbol{r}_i(\boldsymbol{k})$$

and the approximate Hessian matrix

$$\boldsymbol{H}L(\boldsymbol{k}) \approx \sum_{i=1}^{N} \boldsymbol{J}\boldsymbol{r}_i(\boldsymbol{k})^{\mathsf{T}} \boldsymbol{J}\boldsymbol{r}_i(\boldsymbol{k})$$

are computed piece by piece, exploiting the Jacobian matrix of a target position with respect to the kinematic parameters.

According to eq. (5.25), at least 5 tool poses in the work-space would be strictly needed to identify 30 independent kinematic parameters, then 15 joint postures abundantly suffice and may help to reduce any undesired random effect. Thanks to the gift of proportionality guaranteed to the model designed for calibration, the kinematic parameters taken in consideration induce differential variations uncorrelated from each other, hence the Jacobian matrix is always full rank. As explained in section 5.2, the

computational effort required to calculate the full Hessian matrix (83700 second derivatives have to be evaluated to compute the second order remainder in eq. (5.24)) does not get reflected in better convergence performances whatsoever, in fact the algorithm is heavily slowed down. For these reasons, the parameter vector is updated, according to the Gauss-Newton method, as

$$\Delta \boldsymbol{k} = -\boldsymbol{H} L(\boldsymbol{k})^{-1} \nabla L(\boldsymbol{k})$$

and the precess is repeated iteratively, posing $\boldsymbol{k} + \Delta \boldsymbol{k}$ as initial value for the next step, until $|\Delta L(\boldsymbol{k})| \leq \max\{\delta, \epsilon\,|L(\boldsymbol{k})|\}$ or $\|\nabla L(\boldsymbol{k})\| \leq \gamma$ where δ, ϵ are respectively the absolute and relative tolerance on the function change and γ is the tolerance on the gradient norm.

The Jacobian matrix, fundamental to evaluate the gradient and the Hessian matrix, is exactly calculated by following closely the methodological guidelines of the differential error model developed in section 6.2. The differential matrices of any group of kinematic parameters d, θ, a, α, β are first evaluated in the local reference frame via sheer numerical substitution into eqs. (6.38) to (6.42), and then brought back in the global reference frame performing properly the products with the homogeneous transformation matrices $\boldsymbol{T}_0^{\mathrm{B}}, \boldsymbol{T}_1^0, \ldots, \boldsymbol{T}_6^5$. For the sake of conformity, the differential matrices thus obtained are used to evaluate, remembering eq. (6.36), the derivatives of the target positions with respect to all kinematic parameters, giving rise to a larger Jacobian matrix, though all columns corresponding to those parameters not involved in the model, namely d_0, β_1, d_2, β_3, β_4, β_5, are simply deleted. Finally, the Jacobian matrix of the i-th tool target position with respect to the 30 kinematic parameters is given by

$$\boldsymbol{J}\boldsymbol{r}_i(\boldsymbol{k}) = \begin{pmatrix} \dfrac{\partial \boldsymbol{r}_i(\boldsymbol{k})}{\partial k_1} & \cdots & \dfrac{\partial \boldsymbol{r}_i(\boldsymbol{k})}{\partial k_{30}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{S}\big(\boldsymbol{\delta}_{k_1}\big)\, \boldsymbol{r}_i(\boldsymbol{k}) + \boldsymbol{d}_{k_1} & \cdots & \boldsymbol{S}\big(\boldsymbol{\delta}_{k_{30}}\big)\, \boldsymbol{r}_i(\boldsymbol{k}) + \boldsymbol{d}_{k_{30}} \end{pmatrix}$$

where $\boldsymbol{\delta}_{k_j}$ and $\boldsymbol{d}_{k_j}$ are respectively the differential orientation and position error vectors induced by the j-th kinematic parameter.

The designed algorithmic procedure implementing the numerical optimisation for kinematic parameters identification, has been exhaustively assessed, first by means of a numerical simulation, where it managed to find all parameters up to the machine precision, then by means of an actual test on a smaller robot with another articulated robot unit, a Stäubli TX90.

4.4.2 Thermal Coefficients Estimation

The second step of calibration is thermal error compensation consisting in the numerical identification of the thermal coefficients. In principle, this further step might be addressed as an optimisation problem too, that is to say looking for the values of some additional parameters that minimise the prediction error of a the thermal model. However, as already said before, since no precise analysis of the robot thermal behaviour of may be proposed, then there is no information whatsoever about the structure of the model describing the relationship between temperatures and kinematics. That is

the reason why the thermal coefficients, relating the variation of the kinematic parameters versus the variation of the links temperatures outlined in eq. (4.2), are eventually estimated by means of a simple linear regression technique.

The previous step returned a collection of corrections of the kinematic parameters numerical values at different temperatures. According to the result of the repeated calibrations, practically all kinematic parameters are subject to changes over the entire time span. Nevertheless, after observing thoroughly the trend of parameters values and comparing it with the trend of the link temperatures, it is deemed reasonable to assume that only some of the parameters are significantly correlated with thermal effects. In particular, the largest thermal strain observed in the robot manipulator unit under study is the stretching of links 0, 2 and 4 and the skewing between joints 2 and 3, described respectively by kinematic parameters d_1, a_2, d_4 and β_2.

The general expression of eq. (4.2) induces the thermally induced parametric error

$$\Delta \boldsymbol{k}_j = \Delta \boldsymbol{T}_j \boldsymbol{c}_j \tag{4.4}$$

where $\Delta \boldsymbol{k}_j$ is an array made up of the j-th kinematic parameter corrections identified at different moments, $\Delta \boldsymbol{T}_j$ is a matrix made up of the variations of the link temperatures reasonably associated to the j-th kinematic parameter with respect to the initial reference temperatures measured at different times, and $\boldsymbol{c}_j$ is an array of thermal coefficients. If several calibrations at different thermal conditions are performed and the corresponding link temperatures are recorded, then the j-th thermal coefficients can be computed, applying least square regression, as

$$\hat{\boldsymbol{c}}_j = \Delta \boldsymbol{T}_j^{+} \Delta \boldsymbol{k}_j \tag{4.5}$$

where $\Delta \boldsymbol{T}_j$ is the pseudoinverse of $\Delta \boldsymbol{T}_j$.

4.5 Correction Outcomes

4.5.1 Regression Data

Applying eq. (4.5) to the target kinematic parameters d_1, a_1, a_2, β_2, d_4, using only the readings of the thermal sensors attached to spots of the robot framework reasonably connected to the value of those parameters, returns a collection of thermal coefficients. The corresponding predicted parametric errors are thus given by

$$\Delta \hat{\boldsymbol{k}}_j \coloneqq \Delta \boldsymbol{T}_j \hat{\boldsymbol{c}}_j$$

that is to say substituting the estimated thermal coefficients into eq. (4.4). The thermal parametric corrections predicted by such a relationship are then compared with the thermal parametric corrections identified by repeated calibrations, as shown in figs. 4.10 to 4.14, in order to verify the success of the performed regression.

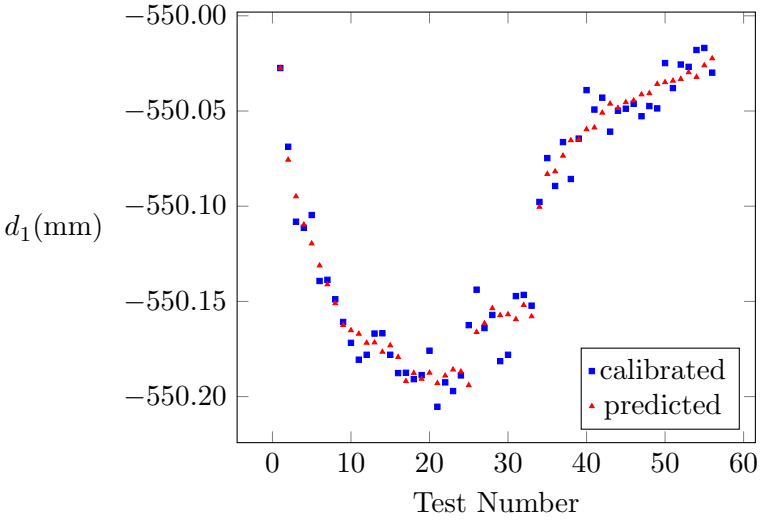

Figure 4.10: Calibrated and Predicted value of d_1.

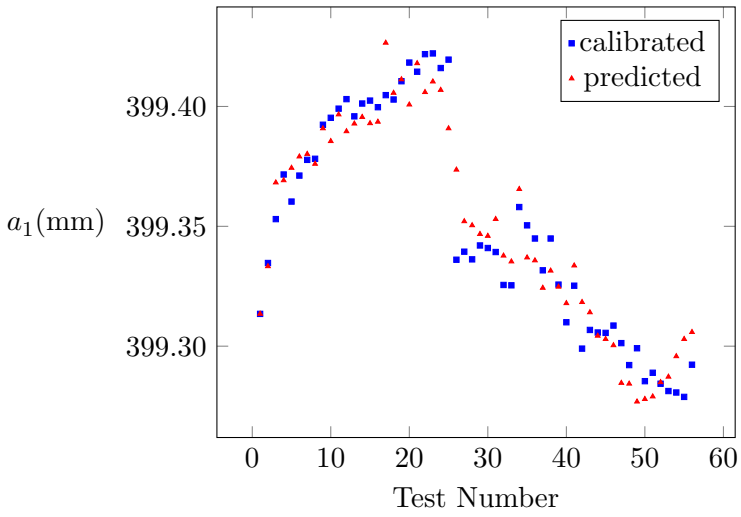

Figure 4.11: Calibrated and Predicted value of a_1.

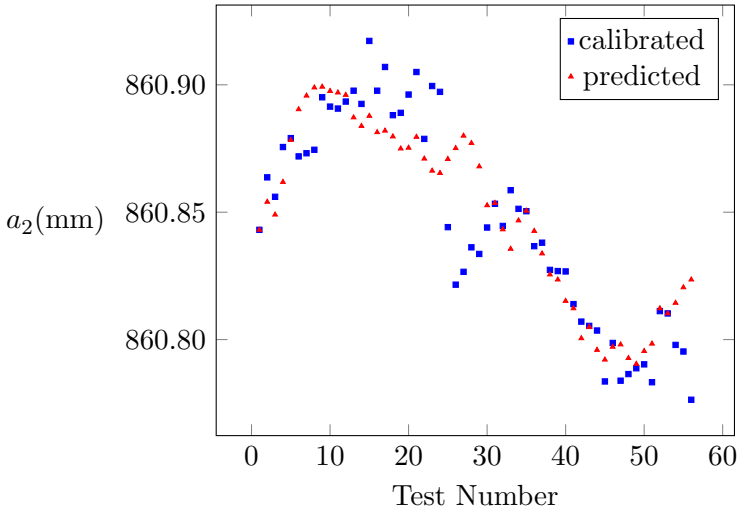

Figure 4.12: Calibrated and Predicted value of a_2.

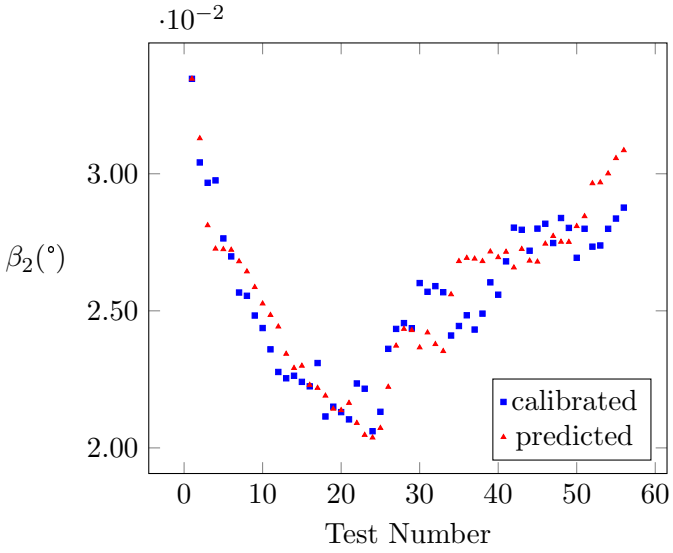

Figure 4.13: Calibrated and Predicted value of β_2.

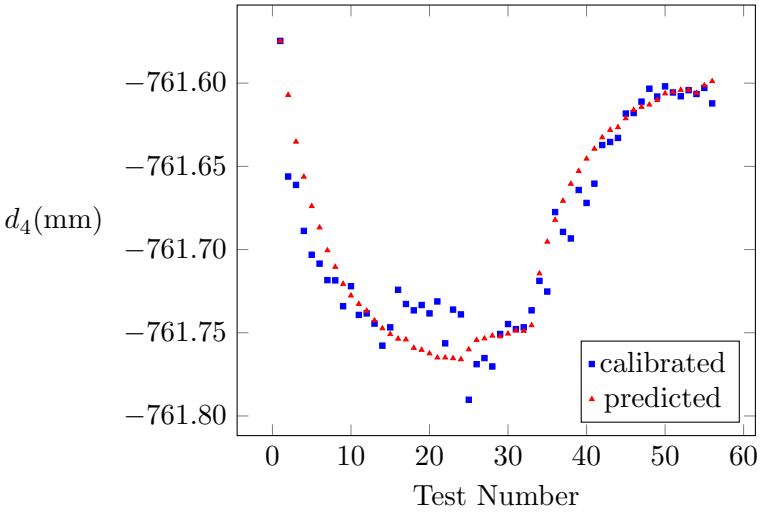

Figure 4.14: Calibrated and Predicted value of d_4.

4.5.2 Validation Results

The assessment of the quality of the designed error compensation scheme is performed following precisely the methodology described in subsection 3.3.4: the accuracy performances provided by the updated kinematic model are evaluated at conditions different from the ones under which calibration took place.

The reliability of the geometric calibration performed over the *restricted* front working volume is verified moving the end effector and checking the corresponding position error given by the kinematic model with the corrected parameters, over a new set of poses still consistent with the portion of work-space explored at calibration stage. The maximum and average tool position errors of the STÄUBLI TX90 are respectively lowered from 1.158mm and 0.676mm down to 0.116mm and 0.059mm; the maximum and average errors of the COMAU NJ-130-2.0 are respectively lowered from 5.703mm and 2.810mm down to 0.578mm and 0.265mm. The plots of the tool position errors at all flange poses are shown in figs. 4.15 and 4.16.

The reliability of the thermal calibration performed over the *entire* working volume is verified by executing two trials of the complete test described in subsection 4.3.4, at different values of speed override, different times of the day and different cycle periods, in order to induce different profiles of the link temperatures. The movements required by warm up and cool down are performed respectively at 75% and 25% of the rated joint speed during the run intended for estimation purposes and at 90% and 30% of the rated joint speed during the run intended for validation purposes. As a result, the results of the thermal calibration are double-checked over a broader span of the link temperature and an opposite trend of the ambient temperature, as evidently shown in fig. 4.17. The initial tool position error of about 3.4mm is reduced by more than 10 times after geometric calibration at room temperature; from this point on, the tool position error given by the

kinematic model with the kinematic parameters calibrated at room temperature grows up to 0.5mm owing to the effect of the links temperature rise, whereas the tool position error given by the kinematic model with the thermally predicted kinematic parameters remains practically constant at about 0.26mm, as shown in the plot of fig. 4.18.

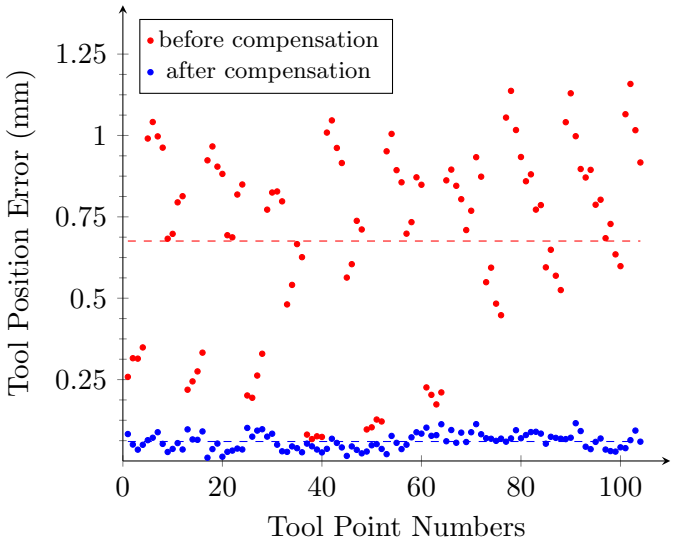

Figure 4.15: Validation of the STÄUBLI TX90 calibration

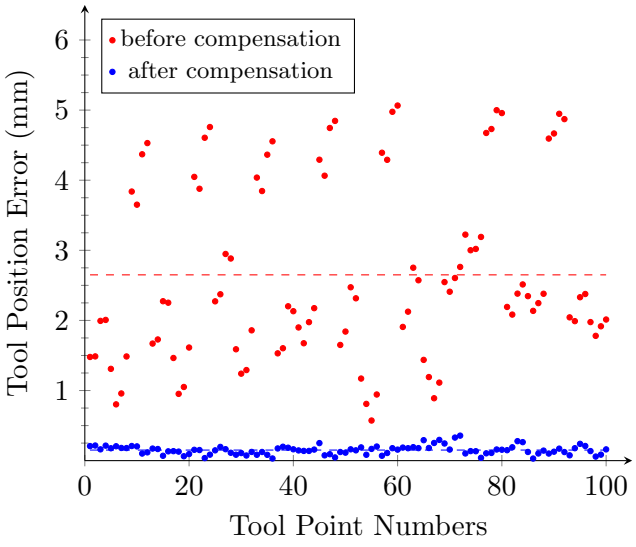

Figure 4.16: Validation of the COMAU NJ-130-2.0 calibration

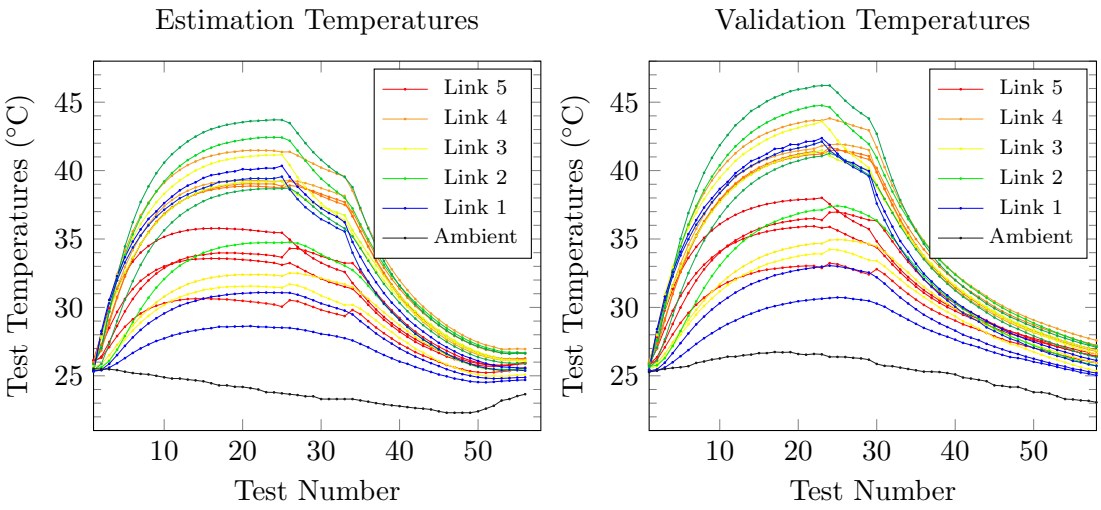

Figure 4.17: Estimation and Validation test temperatures.

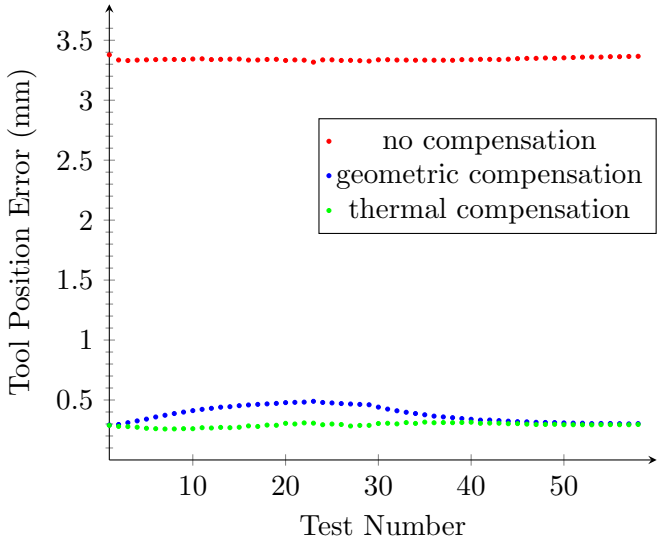

Figure 4.18: Average tool position error at different thermal conditions.

Chapter 5

Optimisation Strategies

5.1 Unconstrained Optimisation for Regression

5.1.1 Non Linear Least Squares Unconstrained Optimisation

Model Identification

Numerical methods may be effectively employed to carry out **model identification**, that is to discover the model of a certain system. The mathematical model of a physical system is a quantitative description of some of its relevant phenomena, needed to investigate that system. A physical system is typically modelled by means of a mathematical function, termed **model function**, representing the relationship between quantities involved in that system. From a purely analytical point of view, the model function is a vector valued function of vector variable; nevertheless, since only some of the quantities are *actually* subject to variation while some others remain fixed, it may be better described as parametric function. From this perspective, the model function gives a numerical relationship between some input quantities and some output quantities, according to the value of some parameters: as the parameters change, the overall relationship changes. In analytical terms, the model function can be expressed as

$$\boldsymbol{z} = \boldsymbol{\phi}\left(\boldsymbol{x}, \boldsymbol{y}\right) \tag{5.1}$$

where $\boldsymbol{x}$ is a vector of model parameters, $\boldsymbol{y}$ is a vector of input variables and $\boldsymbol{z}$ is a vector of output variables.

The quality of a given model depends on how close it manages to predict the actual behaviour of the physical system it seeks to describe. No model, regardless of how accurate it could be, may account for *every* single aspect of a real system; for this reason, the actual value of a certain system quantity may differ from the value of the same quantity predicted by the model. In general, it is not possible to determine perfectly the model function, although in practice it is usually assumed that the functional form of the relationship is sufficiently correct, whereas the values of the parameters are unknown, or at least only *roughly* known. Improving the quality of a model thus means to look for a value for the parameters in such a way that the predicted behaviour gets closer

to the exhibited behaviour. In this sense, model identification is in fact equivalent to parameter identification.

For a given value $\boldsymbol{y}$ of the input quantities, the actual value $\boldsymbol{\zeta}$ of the output quantities may be estimated directly or indirectly, basing on the measurement of some quantities, and compared to the nominal value $\boldsymbol{z}$ of the output quantities given by the model function in eq. (5.1). The difference between these two values

$$\boldsymbol{e}(\boldsymbol{x}) \coloneqq \boldsymbol{\zeta} - \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) \tag{5.2}$$

is called **residual**. Performing several measurements in different conditions and generating the corresponding residuals $\boldsymbol{e}_i(\boldsymbol{x}) \coloneqq \boldsymbol{\zeta}_i - \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}_i)$, may give a reasonable indication of the overall prediction error given by the model for a certain value $\boldsymbol{x}$ of the parameters. The goal of model identification is to find a value of the parameters that gives the lowest possible prediction error.

The problem of fitting a parametric function to a collection of experimental data by tuning the value of the parameters, is called **regression**. As regards model identification, the model parameters are the variables of the regression problem, whereas all the other model quantities are sampled to generate the data of the regression problem. The model function is non linear, in general, therefore, if no assumptions about the linearity can be made, the problem is termed **non linear regression**. It is worth noting that all the model quantities may be thus either constants or variables, depending on the specific context: when the model is employed, the output quantities $\boldsymbol{z}$ are computed processing the input quantities $\boldsymbol{y}$, basing on the value of the parameters $\boldsymbol{x}$, whereas, when the model is identified, the parameters $\boldsymbol{x}$ are estimated basing on the samples of the output quantities $\boldsymbol{z}$ and the input quantities $\boldsymbol{x}$.

Function Minimisation

The term optimisation comes from the Latin word *optimum*, which means best: **mathematical optimisation** indeed consists in looking for the best element among some several possible alternatives, according to some given criteria. Optimising means thus improving something, in general; specifically, it implies either to increase or to decrease the value of a certain performance index changing the value of some quantities, meeting certain constraints. The performance index is analytically rendered by means of a real valued function, called **objective function**, and the target quantities are represented by its independent variables, called **decision variables**, which must lie inside a certain region, called **feasible set**.

When the problem requires to reduce a certain indicator gauging the performances, the objective function is also termed cost function or loss function, in order to highlight that it is a measure of the penalty associated to a given value for the decision variables. In such a case, optimisation is essentially a synonym of function minimisation, hence it is important to define clearly the concept of minimum of a function.

Let $f\colon X \subseteq \mathbb{R}^n \to \mathbb{R}$ be a real valued function defined over the domain X and let

$\boldsymbol{x} \in \mathring{X}$ be an interior point[1]. When

$$\exists \delta > 0 \colon f(\boldsymbol{x} + \Delta\boldsymbol{x}) \geq f(\boldsymbol{x}) \,,\ \forall \Delta\boldsymbol{x} \colon \boldsymbol{x} + \Delta\boldsymbol{x} \in X, \|\Delta\boldsymbol{x}\| \leq \delta$$

the point $\boldsymbol{x}$ is said to be a **local minimum point** of f; the value of the function at a local minimum point is termed **local minimum**. When

$$f(\boldsymbol{x} + \Delta\boldsymbol{x}) \geq f(\boldsymbol{x}) \,,\ \forall \Delta\boldsymbol{x} \colon \boldsymbol{x} + \Delta\boldsymbol{x} \in X$$

the point $\boldsymbol{x}$ is said to be a **global minimum point** of f; the value of the function at a global minimum point is termed **global minimum**. A function may have several local minima, in general, but only a global minimum. Depending on whether the inequality is actually strict or not, the value $f(\boldsymbol{x})$ is called **strong minimum** or **weak minimum**, respectively. At strong minima, the value of the function locally increases along every directions. In contrast, at weak minima, there exists *at least* a direction along which the value of the function remains locally constant.

When the feasible set coincides with the entire domain of the function, the optimisation problem is said to be **unconstrained**, as the decision variable is not subject to any constraint. An unconstrained optimisation problem may be analytically represented as

$$\boldsymbol{x}^* = \underset{\boldsymbol{x} \in X}{\operatorname{argmin}} f(\boldsymbol{x})$$

that is to say $\boldsymbol{x}^*$ is the argument of the function f that minimises its value. Such a problem might be extremely hard and complex; for this reason, it may not always be solved exactly, and sometimes finding a local minimum may be satisfactory enough in many practical situations.

Problem Structure

As mentioned before, model identification and regression in general may be addressed through the tools of numerical analysis: parameter regression can be indeed seen as a particular case of optimisation problem, using the model parameters as decision variables, and some gauge of the magnitude of the model residuals as objective function. From this perspective, parameter identification consists in finding the *optimal* values for the parameters, namely the ones minimising the model prediction error in some sense. Since the parameters of a model are typically not required to be bounded, parameter identification is an unconstrained optimisation problem.

Even when the model output quantity under observation is more than one, it is still better to consider each component of the residual vector in eq. (5.2) separately, that is to say as a different scalar quantity; by virtue of this choice, there are, in general, m residuals $e_i(\boldsymbol{x}) \,, i = 1, \ldots, m$ that may come from either different observing conditions or different model quantities. The selected performance index should account for the *overall*

[1] In topology, the interior $\mathring{X}$ of a set $X \subseteq \mathbb{R}^n$ is the *open* set of all the points of that set which do not belong to the its boundary ∂X, i.e. $\mathring{X} \coloneqq X \setminus \partial X$. The points lying within the interior are called interior points.

model prediction error, therefore all the residuals have to be considered simultaneously. To this aim it is useful to define the m-dimensional residual vector

$$\boldsymbol{e}(\boldsymbol{x}) \coloneqq \big(e_i(\boldsymbol{x})\big)_{1\leq i\leq m}$$

obtained gathering all the m scalar residuals together. Several different measures of the size of the model prediction error might be devised, for instance by making use of some mathematical norm of this residual vector, among which the most common choice is undoubtedly the Euclidean norm: its square coincides with the sum of all the squared residuals and is thus commonly called **residual sum of squares** (**RSS**). In this case, the regression problem consists in the minimisation of a sum of squares and is thus ordinarily called also **least squares**.

The objective function is given by

$$f(\boldsymbol{x}) \coloneqq \frac{1}{2}\left\|\boldsymbol{e}(\boldsymbol{x})\right\|^2 = \frac{1}{2}\sum_{k=1}^{m} e_k^2(\boldsymbol{x})$$

that is half the residual sum of squares. Such function features some peculiar convenient traits: thanks to the square, it is *always* non negative and *everywhere* differentiable; besides these analytical properties, it is also particularly suited for regression problems as it tends to amplify residuals in proportion to their magnitude.

Most of the methods aimed at function minimisation, rely upon some differential quantities, therefore the derivatives of the objective function have to be be either derived exactly or estimated approximately. The gradient vector of a function $f(\boldsymbol{x})$ is defined as

$$\nabla f(\boldsymbol{x}) \coloneqq \left(\frac{\partial f(\boldsymbol{x})}{\partial x_i}\right)_{1\leq i\leq n}$$

that is the vector made up of all its first partial derivatives. The Hessian matrix of a function $f(\boldsymbol{x})$ is defined as

$$\boldsymbol{H} f(\boldsymbol{x}) \coloneqq \left(\frac{\partial^2 f(\boldsymbol{x})}{\partial x_i \partial x_j}\right)_{\substack{1\leq i\leq n\\ 1\leq j\leq n}}$$

that is the matrix made up of all its second partial derivatives. Thus, in order to give an expression of such differential quantities, all the first and second partial derivatives of the objective function have to be evaluated first.

The first partial derivative of the objective function with respect to the i-th decision variable is given by

$$\begin{aligned}\frac{\partial f(\boldsymbol{x})}{\partial x_i} = \frac{1}{2}\frac{\partial}{\partial x_i}\left\|\boldsymbol{e}(\boldsymbol{x})\right\|^2 = \frac{\partial}{\partial x_i}\frac{1}{2}\sum_{k=1}^{m} e_k^2(\boldsymbol{x}) = \frac{1}{2}\sum_{k=1}^{m}\frac{\partial e_k^2(\boldsymbol{x})}{\partial x_i} = \\ = \sum_{k=1}^{m} e_k(\boldsymbol{x})\,\frac{\partial e_k(\boldsymbol{x})}{\partial x_i} = \frac{\partial \boldsymbol{e}(\boldsymbol{x})}{\partial x_i}^{\mathsf{T}}\boldsymbol{e}(\boldsymbol{x})\end{aligned}$$

and the second partial derivative of the objective function with respect of the i-th and the j-th decision variables is given by

$$\begin{aligned}\frac{\partial^2 f(\boldsymbol{x})}{\partial x_i \partial x_j} &= \frac{\partial}{\partial x_i}\frac{\partial f(\boldsymbol{x})}{\partial x_j} = \frac{\partial}{\partial x_i}\sum_{k=1}^{m} e_k(\boldsymbol{x})\,\frac{\partial e_k(\boldsymbol{x})}{\partial x_j} = \sum_{k=1}^{m}\frac{\partial e_k(\boldsymbol{x})}{\partial x_i}\frac{\partial e_k(\boldsymbol{x})}{\partial x_j} + e_k(\boldsymbol{x})\,\frac{\partial^2 e_k(\boldsymbol{x})}{\partial x_i \partial x_j} \\ &= \frac{\partial \boldsymbol{e}(\boldsymbol{x})}{\partial x_i}^{\mathsf{T}}\frac{\partial \boldsymbol{e}(\boldsymbol{x})}{\partial x_j} + \sum_{k=1}^{m} e_k(\boldsymbol{x})\,\frac{\partial^2 e_k(\boldsymbol{x})}{\partial x_i \partial x_j}\end{aligned}$$

exploiting the chain rule. Eventually, the gradient vector and the Hessian matrix of the objective function may be further rewritten more succinctly as

$$\nabla f(\boldsymbol{x}) = \boldsymbol{J}\boldsymbol{e}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{e}(\boldsymbol{x}) \tag{5.3a}$$

$$\boldsymbol{H}f(\boldsymbol{x}) = \boldsymbol{J}\boldsymbol{e}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{J}\boldsymbol{e}(\boldsymbol{x}) + \sum_{i=1}^{m} e_i(\boldsymbol{x})\,\boldsymbol{H}e_i(\boldsymbol{x}) \tag{5.3b}$$

where

$$\boldsymbol{J}\boldsymbol{e}(\boldsymbol{x}) = \left(\frac{\partial e_i(\boldsymbol{x})}{\partial x_j}\right)_{\substack{1\le i\le m\\ 1\le j\le n}} = \begin{pmatrix} \frac{\partial e_1(x_1)}{\partial x_1} & \cdots & \frac{\partial e_1(x_n)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial e_m(x_1)}{\partial x_1} & \cdots & \frac{\partial e_m(x_n)}{\partial x_n} \end{pmatrix} = \begin{pmatrix} \frac{\partial \boldsymbol{e}(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial \boldsymbol{e}(\boldsymbol{x})}{\partial x_n} \end{pmatrix}$$

is the Jacobian matrix of the aggregated residual $\boldsymbol{e}(\boldsymbol{x})$.

5.1.2 Iterative Methods

Differential Approximation

Unlike direct search methods, such as Fibonacci search or Golden Section search, which look for a local minimum of the objective function by evaluating only the function at a proper grid of points, indirect search methods, also called **iterative methods**, exploit also the information about the derivatives of the objective function. Indirect methods thus require that the objective function is smooth enough and its derivatives are available, by means of either symbolic derivation or numeric approximation, whereas direct methods may also be employed also when the objective function is non smooth or, for some reason, its derivatives are not available whatsoever. The particular approach to optimisation adopted by iterative methods provides a tool to explore the behaviour of the objective function in more consecutive steps, starting from a certain initial point and stopping when a local minimum is spotted. For this reason, there is no guarantee that the solution found is necessarily a global minimum of the objective function.

The function derivatives are used to build a suitable local approximation of the objective function, i.e. valid only in the neighbourhood of the *current* value of the decision variables at a given step: the approximated function $\tilde{f}(\boldsymbol{x})$ is minimised at each step in

place of the actual objective function $f(\boldsymbol{x})$. In formal terms, an iterative method is required to solve, at each step, the following sub problem

$$\Delta\boldsymbol{x}^* = \underset{\Delta\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmin}}\, \tilde{f}(\boldsymbol{x}+\Delta\boldsymbol{x}) \tag{5.4}$$

which is equivalent to determine the increment of the decision variable that minimises the local approximation of the objective function. The solution $\Delta\boldsymbol{x}_k$ to the sub problem at a certain step, corresponding to the value $\boldsymbol{x}_k$ for the decision variable, is fed back to the relaxed problem and the updated variable $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta\boldsymbol{x}_k$ is used as starting point for the next step; the procedure is thus *iteratively* repeated, starting from an initial guess $\boldsymbol{x}_0$, until a certain condition, set by one or more suitable **stopping criteria**, is met.

The most used local differential approximation of the objective function is normally based on the first order Taylor series expansion

$$f(\boldsymbol{x}+\Delta\boldsymbol{x}) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^{\mathsf{T}}\Delta\boldsymbol{x} + \mathrm{o}\big(\|\Delta\boldsymbol{x}\|\big),\ \Delta\boldsymbol{x}\to\boldsymbol{0} \tag{5.5}$$

or also on the second order Taylor series expansion

$$f(\boldsymbol{x}+\Delta\boldsymbol{x}) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^{\mathsf{T}}\Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^{\mathsf{T}}\boldsymbol{H}f(\boldsymbol{x})\,\Delta\boldsymbol{x} + \mathrm{o}\big(\|\Delta\boldsymbol{x}\|^2\big),\ \Delta\boldsymbol{x}\to\boldsymbol{0} \tag{5.6}$$

when the objective function is twice differentiable and its second derivatives are available.

Optimality Conditions

The derivatives of the objective function satisfy some important conditions at minimum points. In order to investigate them, let $g_{\boldsymbol{v}}\colon \mathbb{R}\to\mathbb{R}$ be the auxiliary function defined as $g_{\boldsymbol{v}}(t) \coloneqq f(\overline{\boldsymbol{x}}+t\boldsymbol{v})$, where $\overline{\boldsymbol{x}}\in\mathring{X}$ is an interior point, $\boldsymbol{v}\in\mathbb{R}^n$ is a vector and $t\in\mathbb{R}$ is a scalar. In this way, the value of the objective function at $\overline{\boldsymbol{x}}$ corresponds to the value of the auxiliary function at the origin, i.e. $g_{\boldsymbol{v}}(0) = f(\overline{\boldsymbol{x}})$. Its first derivative is

$$\frac{\mathrm{d}g_{\boldsymbol{v}}(t)}{\mathrm{d}t} = \frac{\partial f(\overline{\boldsymbol{x}}+t\boldsymbol{v})}{\partial t} = \nabla f(\overline{\boldsymbol{x}}+t\boldsymbol{v})^{\mathsf{T}}\boldsymbol{v}$$

thus giving, in particular, $g'_{\boldsymbol{v}}(0) = \nabla f(\overline{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{v}$. Its second derivative is

$$\frac{\mathrm{d}^2 g_{\boldsymbol{v}}(t)}{\mathrm{d}t^2} = \frac{\partial^2 f(\overline{\boldsymbol{x}}+t\boldsymbol{v})}{\partial t^2} = \frac{\partial \nabla f(\overline{\boldsymbol{x}}+t\boldsymbol{v})^{\mathsf{T}}}{\partial t}\boldsymbol{v} = \boldsymbol{v}^{\mathsf{T}}\boldsymbol{H}f(\overline{\boldsymbol{x}}+t\boldsymbol{v})\,\boldsymbol{v}$$

thus giving, in particular, $g''_{\boldsymbol{v}}(0) = \boldsymbol{v}^{\mathsf{T}}\boldsymbol{H}f(\overline{\boldsymbol{x}})\,\boldsymbol{v}$.

If $\overline{\boldsymbol{x}}$ is a minimum point of $f(\boldsymbol{x})$, then 0 must be a minimum point of $g_{\boldsymbol{v}}(t)$. At interior minimum points, functions of single variable feature horizontal tangent line, i.e. $g'_{\boldsymbol{v}}(0) = 0$, and non negative concavity, i.e. $g''_{\boldsymbol{v}}(0) \geq 0$. Since the reasoning holds independently of the selected direction $\boldsymbol{v}$, eventually

$$\nabla f(\overline{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{v} = 0, \forall\boldsymbol{v}\in\mathbb{R}^n \Rightarrow \nabla f(\overline{\boldsymbol{x}}) = \boldsymbol{0} \tag{5.7}$$

that is to say $\overline{\boldsymbol{x}}$ is a **stationary point** or **critical point**, which represents the first order necessary condition for an interior point $\overline{\boldsymbol{x}}$ to be a minimum point and

$$\boldsymbol{v}^{\mathsf{T}}\boldsymbol{H}f(\overline{\boldsymbol{x}})\,\boldsymbol{v} \geq 0, \forall \boldsymbol{v} \in \mathbb{R}^n \Rightarrow \boldsymbol{H}f(\overline{\boldsymbol{x}}) \succeq \boldsymbol{O} \tag{5.8}$$

that is to say the Hessian at $\overline{\boldsymbol{x}}$ is positive semi definite, which represents the second order necessary condition for an interior point $\overline{\boldsymbol{x}}$ to be a minimum point.

When considered together, they give rise to a sufficient condition for an interior point $\overline{\boldsymbol{x}}$ to be a minimum point. The two **necessity optimality conditions** are valid irrespectively of whether the minimum is weak or strong. On the other hand, the strict inequality in the second order condition, implying positive definite Hessian at $\overline{\boldsymbol{x}}$, together with the first order condition, represents a sufficient condition for an interior point $\overline{\boldsymbol{x}}$ to be a strong minimum point.

Rate of Change

More in general, when $\overline{\boldsymbol{x}}$ is not a stationary point, i.e. $\nabla f(\overline{\boldsymbol{x}}) \neq \boldsymbol{0}$, remembering that $g'_{\boldsymbol{v}}(0) = \nabla f(\overline{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{v}$, the auxiliary function $g_{\boldsymbol{v}}(t) = f(\overline{\boldsymbol{x}} + t\boldsymbol{v})$ may have any local behaviour at 0, depending on the specific direction $\boldsymbol{v}$. The scalar product $\nabla f(\overline{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{v}$ is termed **rate of change** of f at $\overline{\boldsymbol{x}}$ along $\boldsymbol{v}$.

When the vector $\boldsymbol{v}$ is perpendicular to the gradient, i.e. $\nabla f(\overline{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{v} = 0$, then the objective function remains locally constant moving along the direction $\boldsymbol{v}$, which is thus tangent to the level set $\{\boldsymbol{x} \in \mathbb{R}^n \colon f(\boldsymbol{x}) = f(\overline{\boldsymbol{x}})\}$. When $\nabla f(\overline{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{v} > 0$, the function is locally increasing moving along the direction $\boldsymbol{v}$, whereas, when $\nabla f(\overline{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{v} < 0$, the function is locally decreasing moving along the direction $\boldsymbol{v}$. Because of eq. (A.3), the rate of change is bounded by the following inequality

$$-\|\nabla f(\overline{\boldsymbol{x}})\|\,\|\boldsymbol{v}\| \leq \nabla f(\overline{\boldsymbol{x}})^{\mathsf{T}}\boldsymbol{v} \leq +\|\nabla f(\overline{\boldsymbol{x}})\|\,\|\boldsymbol{v}\|$$

and, in particular, remembering eq. (A.4), the function displays the highest rate of increase at $\overline{\boldsymbol{x}}$, when the direction $\boldsymbol{v}$ is parallel to the gradient and the highest rate of decrease at $\overline{\boldsymbol{x}}$, when the direction $\boldsymbol{v}$ is anti parallel to the gradient. Such a direction, called anti-gradient, plays an important role in function minimisation. In the scope of optimisation, $\boldsymbol{v}$ is usually a unit vector, called **search direction** and t is usually a positive scalar, called **step size**.

Termination Criteria

Except for few trivial objective functions, the optimal solution cannot be exactly found in a finite number of steps, thus iterative methods might also go on *indefinitely*. For this reason, some reasonable criteria are required to terminate the algorithm whenever additional iterations would not be of any benefit to the problem. First, in order to prevent the algorithm from performing too iterations, regardless of whether the convergence is actually achieved or not, a maximum number of iterations $k_{\max}$ should be set, giving the simple stopping condition $k = k_{\max}$.

In general, in the final steps, the objective function should start to change less and less, hence it may be reasonable to check whether the absolute variation or the relative variation of the objective function go below a given threshold at a each step, giving respectively the stopping conditions

$$|\Delta f(\boldsymbol{x}_k)| = |f(\boldsymbol{x}_{k+1}) - f(\boldsymbol{x}_k)| \leq \delta_f \tag{5.9a}$$

$$\frac{|\Delta f(\boldsymbol{x}_k)|}{|f(\boldsymbol{x}_k)|} = \frac{|f(\boldsymbol{x}_{k+1}) - f(\boldsymbol{x}_k)|}{|f(\boldsymbol{x}_k)|} \leq \epsilon_f \tag{5.9b}$$

where δ_f is the absolute tolerance and ϵ_f is the relative tolerance on the objective function variation. The above conditions may be checked in parallel, avoiding to divide by too small numbers, as $|\Delta f(\boldsymbol{x}_k)| \leq \max\{\delta_f, \epsilon_f\, |f(\boldsymbol{x}_k)|\}$.

On the other hand, in the final steps also the decision variable begin to change less and less, therefore, if it is composed of homogeneous terms, it might be reasonable to check also whether the absolute variation or the relative variation of the decision variable go below a given threshold at a each step, giving respectively the stopping conditions

$$\|\Delta \boldsymbol{x}_k\| = \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\| \leq \delta_x \tag{5.10a}$$

$$\frac{\|\Delta \boldsymbol{x}_k\|}{\|\boldsymbol{x}_k\|} = \frac{\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|}{\|\boldsymbol{x}_k\|} \leq \epsilon_x \tag{5.10b}$$

where δ_x is the absolute tolerance and ϵ_x is the relative tolerance on the decision variable variation. As for the objective function, the two conditions may be checked in parallel as $\|\Delta \boldsymbol{x}_k\| \leq \max\{\delta_x, \epsilon_x\, \|\boldsymbol{x}_k\|\}$.

In virtue of the first order optimality condition in eq. (5.7) and the continuity of the objective function, as the decision variable approaches a stationary point, the gradient of the objective function vanishes. For this reason, it might be reasonable to check, at each step, whether the norm of the gradient go below a given threshold, giving the additional stopping condition

$$\|\nabla f(\boldsymbol{x}_k)\| \leq \gamma \tag{5.11}$$

where γ is the tolerance on the gradient norm.

Steepest Descent Method

Gradient methods seek to find a suitable search direction exploiting *only* the first derivatives of the objective function; they are based on the fundamental concept of **descent direction**, that is to say a search direction along which the value of the objective function locally decreases. In formal terms, a vector $\boldsymbol{w}_k \in \mathbb{R}^n$ is said to be a descent direction of f at $\boldsymbol{x}_k$, if there exists a positive scalar $\overline{\alpha}_k > 0$, such that $f(\boldsymbol{x}_k + \alpha_k \boldsymbol{w}_k) \leq f(\boldsymbol{x}_k)\,, \forall \alpha_k \in [0, \overline{\alpha}_k]$. Defining the decision variable increment $\Delta \boldsymbol{x}_k$ using the above criterion at each step, i.e. as $\Delta \boldsymbol{x}_k = \alpha_k \boldsymbol{w}_k, \forall k$, where $\boldsymbol{w}_k \in \mathbb{R}^n$ is a descent direction and α_k is a suitable step size, generates a sequence of values $f(\boldsymbol{x}_k)$ for the objective function, called relaxation sequence, which is convergent whenever the function is bounded below. If the step size is not chosen properly, the relaxation sequence does

not necessarily converge to a local minimum, although, it still provides an improvement of the initial value $f(\boldsymbol{x}_0)$.

As already shown, the **anti-gradient**, that is the search direction $\boldsymbol{w}_k$ opposite to the gradient

$$\boldsymbol{w}_k = -\nabla f(\boldsymbol{x}_k) \tag{5.12}$$

provides the steepest descent of the objective function, hence it is also known as **steepest descent direction**. For this reason, if a small value for the step size is chosen, the resulting increment on the decision variable is able to produce the largest *local* decrease of the objective function value. The decision variable is thus updated as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha_k \nabla f(\boldsymbol{x}_k)$$

giving rise to the steepest descent method for unconstrained optimisation. As regards non linear least squares, remembering the expression of the gradient in eq. (5.3a), the steepest descent direction is given by

$$\boldsymbol{w}_k = -\boldsymbol{J}\boldsymbol{e}(\boldsymbol{x}_k)^\mathsf{T}\boldsymbol{e}(\boldsymbol{x}_k)$$

therefore it requires *only* the knowledge of the residuals first derivatives.

The update of the algorithm could have been obtained also by considering the linear approximation of the objective function. The objective function locally behaves as the linear function in the decision variable increment

$$l(\Delta\boldsymbol{x}) \coloneqq f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^\mathsf{T}\Delta\boldsymbol{x}$$

coming from the first order approximation of the objective function in eq. (5.5). Minimising this function over the set of small increments gives exactly the same result.

Several different schemes may be devised to select the proper step size α_k at each iteration. The ideal approach would be to solve the single variable optimisation problem

$$\alpha_k^* = \underset{\alpha>0}{\operatorname{argmin}}\, f(\boldsymbol{x}_k + \alpha_k \boldsymbol{w}_k)$$

called **exact line search**, although it is *practically* infeasible. In any case, the optimal step size satisfies the condition

$$\frac{\partial}{\partial \alpha_k} f(\boldsymbol{x}_k + \alpha_k \boldsymbol{w}_k) = \nabla f(\boldsymbol{x}_k + \alpha_k \boldsymbol{w}_k)^\mathsf{T}\boldsymbol{w}_k = 0$$

that, combined with eq. (5.12), yields the constraint $\nabla f(\boldsymbol{x}_{k+1})^\mathsf{T}\nabla f(\boldsymbol{x}_k) = 0$: the gradient at the $(k+1)$-th iteration becomes orthogonal to the gradient at the k-th iteration, giving rise, on the whole, to an undesired zigzag path in $\mathbb{R}^n$, from the initial point $\boldsymbol{x}_0$ to the optimum point $\boldsymbol{x}^*$.

Since the extent of the update strictly depends on the size of the gradient, it is fairly evident that the steepest descent method *dramatically* slows down as the gradient vector becomes rather small; also for this reason, the algorithm should stop whenever the norm

of the gradient, which is a measure of its size, goes below a certain set threshold, as in the termination criterion of eq. (5.11).

Unfortunately, even when only descent directions are taken, there is no guarantee whatsoever that the critical point found is actually a minimum point: as a matter of fact, the method might also get trapped in saddle points, for it ignores the second order optimality condition of eq. (5.8). In order to find out whether a minimum point or a saddle point has been found, it is necessary to evaluate the Hessian matrix of the objective function at that stationary point. When there is no way of estimating or even only approximating the Hessian matrix, some additional iterations checking search directions different from the last one, are required in order to rule out the existence of negative curvature directions at critical points.

Newton-Raphson Method

Newton methods exploit also of the second derivatives of the objective function to find directly a minimiser, using the optimality condition of stationary points directly, without looking for search directions. **Newton-Raphson** was originally a numerical root finding algorithm: formally, the problem consists in looking for the zeros of a vector field $\boldsymbol{h}\colon \mathbb{R}^n \to \mathbb{R}^n$, that is solving the system of non linear equations $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}$. Taking the first order Taylor series expansion of the function $\boldsymbol{h}(\boldsymbol{x})$ at $\boldsymbol{x}$

$$\boldsymbol{h}(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{J}\boldsymbol{h}(\boldsymbol{x})\,\Delta\boldsymbol{x} + \mathrm{o}\left(\Delta\boldsymbol{x}\right), \Delta\boldsymbol{x} \to \boldsymbol{0} \tag{5.13}$$

the problem may be numerically addressed by solving iteratively the linearised system of equations $\boldsymbol{h}(\boldsymbol{x}_k + \Delta\boldsymbol{x}_k) \approx \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{J}\boldsymbol{h}(\boldsymbol{x}_k)\,\Delta\boldsymbol{x}_k = \boldsymbol{0}$, called **Newton system**. At each iteration, the estimate of the root is thus updated as $\Delta\boldsymbol{x}_k = -\boldsymbol{J}\boldsymbol{h}(\boldsymbol{x}_k)^{-1}\,\boldsymbol{h}(\boldsymbol{x}_k)$, whenever the Jacobian matrix is full rank.

Remembering that minimum points have to be stationary points, in view of eq. (5.7), employing the gradient of the objective function as vector field, such a method may be easily adapted to unconstrained optimisation problems. In this case, the series expansion of eq. (5.13) becomes

$$\nabla f(\boldsymbol{x} + \Delta\boldsymbol{x}) = \nabla f(\boldsymbol{x}) + \boldsymbol{H}f(\boldsymbol{x})\,\Delta\boldsymbol{x} + \mathrm{o}\big(\,\|\Delta\boldsymbol{x}\|\,\big),\ \Delta\boldsymbol{x} \to \boldsymbol{0}$$

because $\boldsymbol{J}\nabla f(\boldsymbol{x}) = \boldsymbol{H}f(\boldsymbol{x})^{\mathsf{T}}$ and the Hessian matrix is symmetric. If the Hessian matrix is non singular, the increment is computed as

$$\Delta\boldsymbol{x}_k = -\boldsymbol{H}f(\boldsymbol{x}_k)^{-1}\,\nabla f(\boldsymbol{x}_k) \tag{5.14}$$

therefore the decision variable is updated as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{H}f(\boldsymbol{x}_k)^{-1}\,\nabla f(\boldsymbol{x}_k)$$

giving rise to the **Newton-Raphson method** for unconstrained optimisation. As regards non linear least squares, remembering the expression of the gradient and the

Hessian in eqs. (5.3a) and (5.3b) respectively, the Newton-Raphson update is given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \left(\boldsymbol{J}\boldsymbol{e}(\boldsymbol{x})^\mathsf{T} \boldsymbol{J}\boldsymbol{e}(\boldsymbol{x}) + \sum_{i=1}^{m} e_i(\boldsymbol{x})\, \boldsymbol{H}e_i(\boldsymbol{x}) \right)^{-1} \boldsymbol{J}\boldsymbol{e}(\boldsymbol{x}_k)^\mathsf{T} \boldsymbol{e}(\boldsymbol{x}_k) \tag{5.15}$$

therefore it requires the knowledge of the residuals second derivatives *too*.

The update of the algorithm could have been obtained also by considering the quadratic approximation of the objective function. The objective function locally behaves as the quadratic function in the decision variable increment

$$q(\Delta\boldsymbol{x}) \coloneqq f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^\mathsf{T} \Delta\boldsymbol{x} + \frac{1}{2} \Delta\boldsymbol{x}^\mathsf{T} \boldsymbol{H}f(\boldsymbol{x})\, \Delta\boldsymbol{x}$$

coming from the second order approximation of the objective function in eq. (5.6). Differentiating this function and zeroing the gradient gives exactly the same result.

Again, the extent of the update is partly determined by the size of the gradient, therefore also the Newton-Raphson method tends to slow down considerably as the gradient becomes smaller. Again, a suitable lower bound on the norm of the gradient, as in eq. (5.11), may constitute a reasonable termination criterion for the algorithm.

The Newton-Raphson method makes use only of the first order optimality condition, thus the solution might be whichever critical point, not necessarily a minimum point: the optimum point $\boldsymbol{x}^*$ is a minimum point only when the Hessian matrix of the objective function there evaluated $\boldsymbol{H}f(\boldsymbol{x}^*)$ is positive semi definite, according to eq. (5.8). If the Hessian matrix becomes indefinite or negative semi definite somewhere, the algorithm gets attracted to a saddle point or a maximum point, respectively; hence it is crucial to check whether $\boldsymbol{H}f(\boldsymbol{x}_k)$ is positive semi definite at each step, and switch to a different strategy, e.g., the gradient method, otherwise. Moreover, when $\boldsymbol{H}f(\boldsymbol{x}_k)$ is only positive semi definite, it may even be singular, therefore eq. (5.14) cannot be employed and the decision variable increment must be computed solving the system of linear equations

$$\nabla f(\boldsymbol{x}_k) + \boldsymbol{H}f(\boldsymbol{x}_k)\, \Delta\boldsymbol{x}_k = \boldsymbol{0} \tag{5.16}$$

using a suitable method.

Gauss-Newton Method

The Newton-Raphson method typically provides better convergence performances than the steepest descent method, but each iteration is definitely far more time and resource consuming, as it requires the evaluation of *all* the second derivatives of the objective function with respect to the decision variables, in order to build the Hessian matrix.

When the Newton-Raphson method is applied to a non linear least squares problem, the Hessian matrix of the objective function given by eq. (5.3b) is made up of the sum of two terms. It includes a term involving the Hessian matrices $\boldsymbol{H}e_i(\boldsymbol{x})$ of all m individual residuals; remembering the number of free entries in a symmetric matrix given by eq. (A.18), it thus requires the computation of $mn(n+1)/2$ second derivatives in

total. On the other hand, it includes also the term $\boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})$ involving the Jacobian matrix of the aggregated residual, that is to say the first derivatives of the individual residuals *only*. However, the Jacobian matrix of the aggregated residual is required to evaluate also the gradient of the sum of squares function, as put in evidence by eq. (5.3a), therefore it is *already* available for computing that term of the Hessian matrix. In light of this consideration, it might be reasonable to neglect all the second derivatives, resulting in the following estimation of the Hessian matrix of the sum of squares function

$$\boldsymbol{H}f(\boldsymbol{x}) = \boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x}) + \sum_{i=1}^{m} e_i(\boldsymbol{x})\,\boldsymbol{H}e_i(\boldsymbol{x}) \approx \boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})$$

called Gauss-Newton approximation.

If the product $\boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})$ is non singular, the increment can be computed as

$$\Delta\boldsymbol{x}_k = -\big(\boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})\big)^{-1}\boldsymbol{Je}(\boldsymbol{x}_k)^{\mathsf{T}}\boldsymbol{e}(\boldsymbol{x}_k) \tag{5.17}$$

therefore the decision variable is updated as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \big(\boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})\big)^{-1}\boldsymbol{Je}(\boldsymbol{x}_k)^{\mathsf{T}}\boldsymbol{e}(\boldsymbol{x}_k)$$

giving rise to the so called **Gauss-Newton method** for non linear least squares optimisation. More in general, the increment must satisfy the system of linear equations

$$\boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})\,\Delta\boldsymbol{x}_k = -\boldsymbol{Je}(\boldsymbol{x}_k)^{\mathsf{T}}\boldsymbol{e}(\boldsymbol{x}_k)$$

also called **least squares normal equation**. Such a matrix equation has solution

$$\Delta\boldsymbol{x}_k = -\boldsymbol{Je}(\boldsymbol{x})^{+}\boldsymbol{e}(\boldsymbol{x}_k)$$

where $\boldsymbol{Je}(\boldsymbol{x})^{+}$ is the pseudo inverse of $\boldsymbol{Je}(\boldsymbol{x})$. When the Jacobian matrix is full rank, its pseudo inverse is indeed given by $\boldsymbol{Je}(\boldsymbol{x})^{+} = \big(\boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})\big)^{-1}\boldsymbol{Je}(\boldsymbol{x}_k)^{\mathsf{T}}$.

The Gauss-Newton approximation of the Hessian matrix significantly reduces the computational effort for each iteration and it is especially valid under certain circumstances, namely when the individual residuals are quite small, when the model function is only mildly non linear or when the terms in the summation annihilate one another each other owing to sign differences. In all of these situations the second term of the Hessian matrix becomes *almost* negligible, hence the Gauss-Newton method converges to the Newton-Raphson method.

The update of the algorithm could have been obtained also by considering the first order Taylor series expansion of the aggregated residual

$$\boldsymbol{e}(\boldsymbol{x}+\Delta\boldsymbol{x}) = \boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{Je}(\boldsymbol{x})\,\Delta\boldsymbol{x} + \mathrm{o}\left(\|\Delta\boldsymbol{x}\|\right),\ \Delta\boldsymbol{x}\to\boldsymbol{0}$$

to build the second order Taylor series expansion of the objective function

$$f(\boldsymbol{x}+\Delta\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{e}(\boldsymbol{x}+\Delta\boldsymbol{x})\|^2 = \frac{1}{2}\|\boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{Je}(\boldsymbol{x})\,\Delta\boldsymbol{x} + \mathrm{o}\left(\|\Delta\boldsymbol{x}\|\right)\|^2 =$$

$$= \frac{1}{2}\Big(\boldsymbol{e}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{e}(\boldsymbol{x}) + 2\boldsymbol{e}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})\,\Delta\boldsymbol{x} + \big(\boldsymbol{Je}(\boldsymbol{x})\,\Delta\boldsymbol{x}\big)^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})\,\Delta\boldsymbol{x} + \mathrm{o}\big(\|\Delta\boldsymbol{x}\|^2\big)\Big) =$$

$$= \frac{1}{2}\|\boldsymbol{e}(\boldsymbol{x})\|^2 + \big(\boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{e}(\boldsymbol{x})\big)^{\mathsf{T}}\Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^{\mathsf{T}}\big(\boldsymbol{Je}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{Je}(\boldsymbol{x})\big)\Delta\boldsymbol{x} + \mathrm{o}\big(\|\Delta\boldsymbol{x}\|^2\big),\ \Delta\boldsymbol{x}\to\boldsymbol{0}$$

using only the first derivatives of the residuals. Differentiating it with respect to $\Delta\boldsymbol{x}$ and setting the gradient to zero gives exactly the same solution.

Levenberg-Marquardt Method

Using the Newton-Raphson update as search direction $\boldsymbol{w} \coloneqq -\boldsymbol{H}f(\boldsymbol{x})^{-1}\nabla f(\boldsymbol{x})$ with a small step size $\alpha > 0$, gives the asymptotic expansion

$$f(\boldsymbol{x}+\alpha\boldsymbol{w}) = f(\boldsymbol{x}) - \alpha\nabla f(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{H}f(\boldsymbol{x})^{-1}\nabla f(\boldsymbol{x}) + \mathrm{o}(\alpha),\ \alpha \to 0$$

therefore $-\boldsymbol{H}f(\boldsymbol{x})^{-1}\nabla f(\boldsymbol{x})$ is a descent direction whenever $\boldsymbol{H}f(\boldsymbol{x})$ is positive semi definite. In contrast, the search direction $\boldsymbol{w} \coloneqq -\big(\boldsymbol{J}\boldsymbol{e}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{J}\boldsymbol{e}(\boldsymbol{x})\big)^{-1}\boldsymbol{J}\boldsymbol{e}(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{e}(\boldsymbol{x})$, corresponding to the Gauss-Newton update, is always a descent direction, as the approximate Hessian matrix is *intrinsically* positive semi definite.

Unfortunately, the update of the Newton methods, in either case, does not guarantee a decrease of the value of the objective function, because, even if it provides a descent direction, the 'equivalent' unit step size $\alpha = 1$ implicitly adopted by such these methods, might be also excessively large. All of the above considerations suggest the possibility to employ either the Newton-Raphson method or the Gauss-Newton method only to yield a valid search direction, and then to perform a line search to single out the *proper* step size, giving rise to so called **damped Newton methods**.

In any case, such an enhancement is of no help when the Hessian matrix, either exact or approximate, is only positive semi definite. Such a problem may be though overcome by slightly modifying the matrix subject to inversion as

$$\Delta\boldsymbol{x}_k = -\big(\boldsymbol{H}f(\boldsymbol{x}_k) + \mu_k\boldsymbol{I}\big)^{-1}\nabla f(\boldsymbol{x}_k) \tag{5.18}$$

where $\mu_k > 0$ is a suitable correction factor. Provided that the factor μ_k is large enough, the additional correction based on the identity matrix ensures that the overall matrix is positive definite[2] and thus invertible. The decision variable is then updated as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \big(\boldsymbol{H}f(\boldsymbol{x}_k) + \mu_k\boldsymbol{I}\big)^{-1}\nabla f(\boldsymbol{x}_k) \tag{5.19}$$

giving rise to the so called **Levenberg-Marquardt method** for unconstrained optimisation.

The factor μ_k has to be properly sized on line at each iteration. In this regard, it is worth noting that, as the factor gets smaller and smaller

$$\Delta\boldsymbol{x}_k = -\big(\boldsymbol{H}f(\boldsymbol{x}_k) + \mu_k\boldsymbol{I}\big)^{-1}\nabla f(\boldsymbol{x}_k) \sim -\boldsymbol{H}f(\boldsymbol{x}_k)^{-1}\nabla f(\boldsymbol{x}_k),\ \mu_k \to 0$$

it degenerates into standard Newton method, while, as the factor gets larger and larger

$$\Delta\boldsymbol{x}_k = -\big(\boldsymbol{H}f(\boldsymbol{x}_k) + \mu_k\boldsymbol{I}\big)^{-1}\nabla f(\boldsymbol{x}_k) \sim -\frac{1}{\mu_k}\nabla f(\boldsymbol{x}_k),\ \mu_k \to \infty$$

[2] If $\boldsymbol{v} \in \mathbb{R}^n$ is an eigenvector associated to the eigenvalue λ of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, then $\lambda + \mu$ is an eigenvalue of $\boldsymbol{A} + \mu\boldsymbol{I}$, since $(\boldsymbol{A} + \mu\boldsymbol{I})\boldsymbol{v} = \boldsymbol{A}\boldsymbol{v} + \mu\boldsymbol{v} = \lambda\boldsymbol{v} + \mu\boldsymbol{v} = (\lambda + \mu)\boldsymbol{v}$.

it degenerates into standard gradient method. Bearing in mind these considerations, a reasonable strategy could be to adopt higher values in the initial phases, far from the optimum, in order to guarantee a decrease by taking small steps along the steepest descent direction, and switch to lower values in the final phase, near the optimum, in order to speed up the convergence by taking into account also the curvature of the objective function. In any case, the choice might even be adjusted from time to time, e.g., by lowering the coefficient when the objective function does not exhibit a decrease of its value.

The Levenberg-Marquardt method could have also been derived considering, at each step, a modified version of the optimisation sub problem in eq. (5.4), namely

$$\Delta\boldsymbol{x}^* = \underset{\Delta\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmin}} \tilde{f}(\boldsymbol{x}+\Delta\boldsymbol{x}) + \frac{1}{2}\mu\,\|\Delta\boldsymbol{x}\|^2$$

called **regularised optimisation problem**. Such a modified formulation involves also a penalty for larger step sizes, controlled by the value of the coefficient μ. The second order Taylor series expansion of the regularised function is

$$\begin{aligned} &f(\boldsymbol{x}+\Delta\boldsymbol{x}) + \frac{1}{2}\mu\,\|\Delta\boldsymbol{x}\|^2 = \\ &= f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^\mathsf{T}\Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^\mathsf{T}\boldsymbol{H}f(\boldsymbol{x})\,\Delta\boldsymbol{x} + \frac{1}{2}\mu\Delta\boldsymbol{x}^\mathsf{T}\Delta\boldsymbol{x} + \mathrm{o}\big(\,\|\Delta\boldsymbol{x}\|^2\,\big) = \\ &= f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^\mathsf{T}\Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^\mathsf{T}\big(\boldsymbol{H}f(\boldsymbol{x}) + \mu\boldsymbol{I}\big)\Delta\boldsymbol{x} + \mathrm{o}\big(\,\|\Delta\boldsymbol{x}\|^2\,\big),\ \Delta\boldsymbol{x}\to\boldsymbol{0} \end{aligned}$$

therefore applying the Newton method to this function gives exactly the same result.

5.2 Kinematic Parameters Identification

5.2.1 Problem Elements

Decision Variables

The ultimate goal of geometric calibration is the accurate determination of the manipulator kinematic model, therefore the problem may be addressed following the reasoning outlined in subsection 5.1.1. In this context, mathematical optimisation is called into question for the numerical identification of the kinematic parameters.

There are 7 groups of parameters $\boldsymbol{k}_0,\ldots,\boldsymbol{k}_6$ in the articulated manipulator kinematic model selected for calibration purposes, corresponding to the different reference frame transformations of the manipulator kinematic chain. They are packed all together into a large array

$$\boldsymbol{k} \coloneqq \begin{pmatrix} \boldsymbol{k}_0 \\ \vdots \\ \boldsymbol{k}_6 \end{pmatrix} \in \mathbb{R}^{n_\mathrm{k}}$$

which constitutes the vector decision variable of the optimisation problem set up for manipulator calibration.

The actual kinematic parameters in each group $\boldsymbol{k}_j$ with $j = 0, \ldots, 6$, depend on the specific parametrisation of the corresponding kinematic transformation. In particular, for the **complete**, **minimal** and **proportional model** described in section 4.2, the kinematic parameters are

$$\boldsymbol{k}_j \coloneqq \begin{pmatrix} d_j \\ \theta_{\mathrm{o}j} \\ a_j \\ \alpha_j \end{pmatrix},\ j = 1,3,4,5, \quad \boldsymbol{k}_j \coloneqq \begin{pmatrix} \theta_{\mathrm{o}j} \\ a_j \\ \alpha_j \\ \beta_j \end{pmatrix},\ j = 0,2, \quad \boldsymbol{k}_j \coloneqq \begin{pmatrix} d_j \\ \theta_{\mathrm{o}j} \\ a_j \\ \alpha_j \\ \beta_j \\ b_j \end{pmatrix},\ j = 6$$

because Denavit-Hartenberg convention is adopted by default but in the case of parallel axes, for which Hayati-Mirmirani convention is adopted instead, whereas the last transformation requires a full set of six parameters, therefore the total number of kinematic parameters is $n_{\mathrm{k}} = 5 \cdot 4 + 6 = 30$.

Objective Function

As described in subsection 5.1.1, the identification of the kinematic parameters is carried out through minimisation of the position error given by the targets on the manipulator end effector, over a certain set of kinematic configurations. The h-th position error $\Delta\boldsymbol{r}_h \in \mathbb{R}^3$ is the 3D vector defined as

$$\Delta\boldsymbol{r}_h \coloneqq \overline{\boldsymbol{r}}_h - \boldsymbol{r}_h \tag{5.20}$$

where $\overline{\boldsymbol{r}}_h$ is the actual h-th position, measured through an external instrument, and $\boldsymbol{r}$ is the nominal h-th position computed through the kinematic model.

The different points come from either different manipulator postures or different tool targets; specifically, if the manipulator is in the i-th encoder joint angle configuration, and the position measuring instrument is pointing at j-th tool target, then the position index is given by $h \coloneqq (i-1)n_{\mathrm{T}} + j$, where n_{T} is the total number of tool targets, i.e. $j = 1, \ldots, n_{\mathrm{T}}$. Because the kinematic model depends on the values of the kinematic parameters, eq. (5.20) can be rewritten as

$$\Delta\boldsymbol{r}_h(\boldsymbol{k}) \coloneqq \overline{\boldsymbol{r}}_h - \boldsymbol{r}_h(\boldsymbol{k})$$

therefore each position error is actually a function of the parameter vector $\boldsymbol{k}$. If the manipulator moves in N_{P} different joint postures and the coordinates of the tool targets are recorded for all corresponding flange poses, then $n_{\mathrm{T}}N_{\mathrm{P}}$ positions are available. The $N = n_{\mathrm{T}}N_{\mathrm{P}}$ position errors are stacked and bundled into a compact vector

$$\Delta\boldsymbol{r}(\boldsymbol{k}) \coloneqq \begin{pmatrix} \Delta\boldsymbol{r}_1(\boldsymbol{k}) \\ \vdots \\ \Delta\boldsymbol{r}_N(\boldsymbol{k}) \end{pmatrix} \in \mathbb{R}^{3N} \tag{5.21}$$

called piled or aggregated position error.

In the framework of robot calibration, the goal of optimisation is to reduce the position errors in all 3D points *concurrently*. To this aim, the Euclidean norm of the piled position error $\Delta\boldsymbol{r}$ is selected as performance index; since the norm is a non negative quantity by definition, minimising it is equivalent to minimising its square, then the standard residual sum of squares

$$L(\boldsymbol{k}) \coloneqq \frac{1}{2} \left\| \Delta\boldsymbol{r}(\boldsymbol{k}) \right\|^2 = \frac{1}{2} \Delta\boldsymbol{r}\left(\boldsymbol{k}\right)^{\mathsf{T}} \Delta\boldsymbol{r}\left(\boldsymbol{k}\right)$$

is ultimately designated as loss function. Thanks to the properties of the squared norm, the loss function is separable into the sum of N loss terms, as

$$L(\boldsymbol{k}) = \sum_{i=1}^{N} L_i(\boldsymbol{k}) = \sum_{i=1}^{N} \frac{1}{2} \left\| \Delta\boldsymbol{r}_i(\boldsymbol{k}) \right\|^2$$

where each

$$\left\| \Delta\boldsymbol{r}_i(\boldsymbol{k}) \right\|^2 = \left(\overline{x}_i - x_i(\boldsymbol{k})\right)^2 + \left(\overline{y}_i - y_i(\boldsymbol{k})\right)^2 + \left(\overline{z}_i - z_i(\boldsymbol{k})\right)^2$$

is the square of the i-th distance error.

The differential elements required for optimisation may be indeed computed applying eqs. (5.3a) and (5.3b) to all individual terms. The gradient vector of the loss function is

$$\nabla L(\boldsymbol{k}) = \sum_{i=1}^{N} \nabla L_i(\boldsymbol{k})$$

where

$$\nabla L_i(\boldsymbol{k}) = -\boldsymbol{J}\boldsymbol{r}_i(\boldsymbol{k})^{\mathsf{T}} \Delta\boldsymbol{r}_i(\boldsymbol{k}) \tag{5.22}$$

is the gradient vector of $L_i(\boldsymbol{k})$. The Hessian matrix of the loss function is

$$\boldsymbol{H}L(\boldsymbol{k}) = \sum_{i=1}^{N} \boldsymbol{H}L_i(\boldsymbol{k})$$

where

$$\boldsymbol{H}L_i(\boldsymbol{k}) = \boldsymbol{J}\boldsymbol{r}_i(\boldsymbol{k})^{\mathsf{T}} \boldsymbol{J}\boldsymbol{r}_i(\boldsymbol{k}) - \Delta x_i(\boldsymbol{k})\, \boldsymbol{H}x_i(\boldsymbol{k}) - \Delta y_i(\boldsymbol{k})\, \boldsymbol{H}y_i(\boldsymbol{k}) - \Delta z_i(\boldsymbol{k})\, \boldsymbol{H}z_i(\boldsymbol{k}) \tag{5.23}$$

is the Hessian matrix of $L_i(\boldsymbol{k})$.

5.2.2 Iterative Update

Identification Jacobian

The unconstrained non linear least squares optimisation problem is solved using the iterative methods described in section 5.1; in view of this aim, the differential quantities computed above may be rewritten by making use of a more convenient notation,

exploiting eq. (5.21). Definining the aggregated Jacobian matrix,

$$\boldsymbol{Jr}(\boldsymbol{k}) \coloneqq \begin{pmatrix} \boldsymbol{Jr}_1(\boldsymbol{k}) \\ \vdots \\ \boldsymbol{Jr}_N(\boldsymbol{k}) \end{pmatrix} \in \mathbb{R}^{3N \times n_\mathrm{k}}$$

also called identification Jacobian, the overall gradient vector of the loss function, defined by eq. (5.22), can be compactly rewritten, as

$$\nabla L(\boldsymbol{k}) = -\sum_{i=1}^{N} \boldsymbol{Jr}_i(\boldsymbol{k})^\mathsf{T} \Delta \boldsymbol{r}_i(\boldsymbol{k}) = -\boldsymbol{Jr}(\boldsymbol{k})^\mathsf{T} \Delta \boldsymbol{r}(\boldsymbol{k})$$

and the overall Hessian matrix of the loss function, defined by eq. (5.23), can be compactly rewritten as

$$\boldsymbol{H}L(\boldsymbol{k}) = \sum_{i=1}^{N} \boldsymbol{Jr}_i(\boldsymbol{k})^\mathsf{T} \boldsymbol{Jr}_i(\boldsymbol{k}) + \sum_{i=1}^{N} \boldsymbol{C}_i = \boldsymbol{Jr}(\boldsymbol{k})^\mathsf{T} \boldsymbol{Jr}(\boldsymbol{k}) + \boldsymbol{C} \tag{5.24}$$

where $\boldsymbol{C}_i \coloneqq -\Delta x_i(\boldsymbol{k})\, \boldsymbol{H}x_i(\boldsymbol{k}) - \Delta y_i(\boldsymbol{k})\, \boldsymbol{H}y_i(\boldsymbol{k}) - \Delta z_i(\boldsymbol{k})\, \boldsymbol{H}z_i(\boldsymbol{k})$.

Numerical Issues

In general, the number of samples has to be large enough to allow identification of *all* model parameters. Specifically, since each group of tool position errors related to the same posture may be seen as a complete kinematic constraint, corresponding thus to six independent scalar constraints[3], then the number of points must spring from

$$6N_\mathrm{P} \geq n_\mathrm{k} \tag{5.25}$$

where N_P is the number of joint postures and n_k is the number of kinematic parameters. Such a lower bound is though only necessary, but not sufficient, because some of the postures might correlate the differential parametric variations.

In theory, only 5 robot postures, cleverly selected, would thus suffice to achieve complete identification of the 30 target kinematic parameters; in practice some more measurements may help to reduce the undesired effects of measurement noise, gravity, vibrations, and so on. In any case, whenever eq. (5.25) is met, since $n_\mathrm{T} \geq 3$, the total number of residuals $3N = 3n_\mathrm{T}N_\mathrm{P} \geq 9N_\mathrm{P}$ is far larger than the number of parameters n_k, therefore the identification Jacobian $\boldsymbol{Jr}(\boldsymbol{k}) \in \mathbb{R}^{3N \times n_\mathrm{k}}$ is *naturally* a tall matrix.

When the kinematic model adopted for calibration purposes iscarefully selected and corrected in order to meet all the modelling requirements, as deeply explained in section 4.2, the aggregated Jacobian matrix $\boldsymbol{Jr}(\boldsymbol{k})$, which represents the differential variation of the target positions in all configurations with respect to all parameters variation,

[3] In principle, each group might indeed be processed to yield three tool position errors and three tool orientation errors.

is full rank, therefore the square matrix $\boldsymbol{Jr}(\boldsymbol{k})^{\mathsf{T}}\,\boldsymbol{Jr}(\boldsymbol{k}) \in \mathbb{R}^{n_\mathrm{k}\times n_\mathrm{k}}$ is non singular. The direct kinematic function is mildly non linear, sines and cosines being the only non linear contributions, and the position errors are reasonably small, hence the remainder $\boldsymbol{C}$ of the Hessian matrix based on the second derivatives is made up of relatively modest entries, and the full Hessian matrix is *mostly* due to the product $\boldsymbol{Jr}(\boldsymbol{k})^{\mathsf{T}}\,\boldsymbol{Jr}(\boldsymbol{k})$: it follows that, whenever the identification Jacobian is full rank, the Hessian matrix is usually non singular and thus invertible.

That is the reason why, with a proper kinematic model, the parameters update can be computed, according to either the Newton-Raphson or the Gauss-Newton method, by means of *sheer* matrix inversion. On the contrary, when the kinematic model is quite poor or not good enough to guarantee that the identification Jacobian is full rank, the Hessian matrix, either exact or approximate, may become almost singular, or ill-conditioned in general, thus more prudent iterations based on Levenberg-Marquardt method are definitely advisable in such cases.

Performance Evaluation

In the framework of geometric calibration, as for other regression problems, iterative methods based on the second order expansion of the cost function display far better convergence performances than those based exclusively on the concept of descent direction. Following on from the above argument about the rank of the identification Jacobian, the increment of the kinematic parameters may be computed as

$$\varDelta\boldsymbol{k} = -\boldsymbol{H}L(\boldsymbol{k}_0)^{-1}\nabla L(\boldsymbol{k}_0) = -\big(\boldsymbol{Jr}(\boldsymbol{k}_0)^{\mathsf{T}}\,\boldsymbol{Jr}(\boldsymbol{k}_0) + \boldsymbol{C}\big)^{-1}\boldsymbol{Jr}(\boldsymbol{k}_0)^{\mathsf{T}}\,\varDelta\boldsymbol{r}(\boldsymbol{k}_0)$$

if the exact Hessian is used, or as

$$\varDelta\boldsymbol{k} = -\big(\boldsymbol{Jr}(\boldsymbol{k}_0)^{\mathsf{T}}\,\boldsymbol{Jr}(\boldsymbol{k}_0)\,\big)^{-1}\boldsymbol{Jr}(\boldsymbol{k}_0)^{\mathsf{T}}\,\varDelta\boldsymbol{r}(\boldsymbol{k}_0) = -\boldsymbol{Jr}(\boldsymbol{k}_0)^{+}\varDelta\boldsymbol{r}(\boldsymbol{k}_0)$$

if the approximate Hessian is used. The kinematic parameters are then updated as

$$\boldsymbol{k} = \boldsymbol{k}_0 + \varDelta\boldsymbol{k}$$

and the process is repeated, by setting $\boldsymbol{k}_0 = \boldsymbol{k}$ for the next iteration. The iterative optimisation algorithm starts with the nominal values as initial guess, and stops as soon as the attained values are sufficiently close to the optimal ones, according to some termination criterion, such as the ones defined by eqs. (5.9) and (5.11); the termination criterion defined by eq. (5.10) is not that suitable because the decision variable is not homogeneous, being composed of both angles and lengths.

Depending on the selected type of Hessian matrix, that is to say exact or approximate, the iterative update of the kinematic parameters is computed according to the Newton-Raphson method or the Gauss-Newton method, respectively. In principle, up to $3Nn_\mathrm{k}(n_\mathrm{k}+1)/2$ (corresponding to $1395N$ for a 6DOF articulated manipulator) second derivatives must be computed, at each iteration, to evaluate the term $\boldsymbol{C}$ in the exact Hessian matrix, even though such number is usually reduced, for most of the second order terms, either pure or mixed, are identically zero.

Besides, there is no significant enhancement of the algorithm performance, in terms either of required iterations number, at equal value of the cost function reached, or conversely of value of the attained cost function value, at equal number of iterations performed, in fact quite often such performances are even downgraded when using the exact Hessian matrix of the cost function.[4] The considerable computational effort required both from the practitioner, at design time, and from the computer, at execution time, by the evaluation of the kinematic parameters second derivatives, is then not repaid by a boost in the convergence performances of the iterative algorithm, therefore, in this case, it is not sufficiently justified.

[4]Such apparently paradoxical phenomenon of Newton-based iterative methods for least squares optimisation has been investigated, from a theoretical and practical point of view, in [27].

Chapter 6

Kinematic Error Model

6.1 Rigid Transformation Error Model

As explained in appendix B.3, homogeneous transformation matrices are a powerful mathematical tool often used to represent **frames of reference** in the three-dimensional space. If a reference frame is firmly attached to a given rigid body, the corresponding homogeneous transformation matrix might represent kinematically the body *itself*, that is to say its position and its orientation in the three-dimensional space; whenever the body moves or its actual configuration differs from the expected configuration, that is a kinematic variation, either real (displacement) or virtual (error), takes place, the associated homogeneous transformation matrix is subject to a corresponding variation.

6.1.1 Additive Error

In order to investigate the variation of a kinematic transformation, two generic homogeneous transformation matrices $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$ are considered. The homogeneous transformation matrix variation is simply defined as the arithmetic difference between the two matrices $\Delta\boldsymbol{T} = \boldsymbol{T}_2 - \boldsymbol{T}_1$: in this way, the second matrix can be expressed by adding the matrix variation to the first matrix

$$\boldsymbol{T}_2 = \boldsymbol{T}_1 + \Delta\boldsymbol{T}$$

therefore the matrix variation represents how much the second matrix differs from the first matrix in terms of additive error.

When the homogeneous transformation matrix under consideration is a function of a certain variable ξ, then a change in the independent variable $\xi + \Delta\xi$ naturally yields the homogeneous transformation matrix variation

$$\Delta\boldsymbol{T} \coloneqq \boldsymbol{T}(\xi + \Delta\xi) - \boldsymbol{T}(\xi) \tag{6.1}$$

which is, in this case, a proper function increment.

From now on, regardless of the actual reason behind the kinematic variation, $\boldsymbol{T}$ will represent a reference homogeneous transformation matrix while $\boldsymbol{T} + \Delta\boldsymbol{T}$ will denote

the same homogeneous transformation matrix subjected to a change. Even though the two homogeneous transformation matrices might be *completely* arbitrary, this kind of notation proves to be especially useful when they are different, and *yet* similar: in such a case, the two matrices have nearly equal entries, therefore the matrix difference $\Delta\boldsymbol{T}$ is a relatively small matrix.

6.1.2 Multiplicative Error

The modification of the matrix may be achieved either through matrix sum or through matrix product: the former determines the additive error already considered before, whereas the latter determines the multiplicative error. The same alteration of the transformation matrix $\boldsymbol{T}$ could be represented by means of a small roto-translation, i.e. a combination of a small translation and a small rotation, pre multiplying the homogeneous transformation matrix

$$\boldsymbol{T} + \Delta\boldsymbol{T} = (\mathbf{Trans}\,(\boldsymbol{v}, \Delta t)\,\mathbf{Rot}\,(\boldsymbol{u}, \Delta\theta))\,\boldsymbol{T} \tag{6.2}$$

if the error rotation and translation take place in the original frame, or post multiplying the homogeneous transformation matrix

$$\boldsymbol{T} + \Delta\boldsymbol{T} = \boldsymbol{T}\,(\mathbf{Trans}\,(\boldsymbol{v}, \Delta t)\,\mathbf{Rot}\,(\boldsymbol{u}, \Delta\theta)) \tag{6.3}$$

if the error rotation and translation take place in the transformed frame. The actual values of the quantities defining the error rigid transformation are different in the two cases, in general. In any case, the multiplicative error matrix is a proper homogeneous transformation matrix *itself*: its expression can be easily derived by packing the rotation and the translation transformations into a single matrix

$$\mathbf{Trans}\,(\boldsymbol{v}, \Delta t)\,\mathbf{Rot}\,(\boldsymbol{u}, \Delta\theta) = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{v}\Delta t \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{R}\,(\boldsymbol{u}\Delta\theta) & \boldsymbol{0} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}\,(\boldsymbol{u}\Delta\theta) & \boldsymbol{v}\Delta t \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix}$$

as explained in subsection B.3.2.

The following discussion is based on the multiplicative error matrix coming from pre multiplication. Rearranging eq. (6.2) as

$$\Delta\boldsymbol{T} = \big(\mathbf{Trans}\,(\boldsymbol{v}, \Delta t)\,\mathbf{Rot}\,(\boldsymbol{u}, \Delta\theta)\,\big)\boldsymbol{T} - \boldsymbol{T} = \big(\mathbf{Trans}\,(\boldsymbol{v}, \Delta t)\,\mathbf{Rot}\,(\boldsymbol{u}, \Delta\theta) - \boldsymbol{I}\big)\boldsymbol{T}$$

suggests to define the following matrix

$$\boldsymbol{F} \coloneqq \mathbf{Trans}\,(\boldsymbol{v}, \Delta t)\,\mathbf{Rot}\,(\boldsymbol{u}, \Delta\theta) - \boldsymbol{I} \tag{6.4}$$

hereinafter called **full error matrix**, in order to write

$$\Delta\boldsymbol{T} = \boldsymbol{F}\boldsymbol{T} \tag{6.5}$$

and, substituting this value back in eq. (6.2), also

$$\boldsymbol{T} + \Delta\boldsymbol{T} = (\boldsymbol{I} + \boldsymbol{F})\,\boldsymbol{T}$$

that is an expression of the multiplicative error based on the full error matrix. On the other hand, post multiplying both sides of eq. (6.5) by the inverse homogeneous transformation matrix yields

$$\boldsymbol{F} = \varDelta\boldsymbol{T}\boldsymbol{T}^{-1}$$

which is nothing but an explicit definition of the full error matrix.

Left and Right Error

Since matrix product is not a commutative operation, in general, the order of multiplication is relevant; the perturbation of the homogeneous transformation matrix may be obtained, using the multiplicative error formalism, by either pre or post multiplication, as the alteration may occur in either the original or the transformed frame.

Specifically, the perturbed matrix $\boldsymbol{T} + \varDelta\boldsymbol{T}$ is obtained, from the unperturbed matrix $\boldsymbol{T}$, by means of a rigid transformation, consisting of a translation and a rotation, defined by two unit vectors, namely the displacement direction and the revolution axis: if they are represented in the original frame, the perturbed matrix is given by

$$\varDelta\boldsymbol{T} = {}^{\mathrm{L}}\boldsymbol{F}\boldsymbol{T} \tag{6.6}$$

where ${}^{\mathrm{L}}\boldsymbol{F}$ is termed **left error matrix**; conversely, if they are represented in the transformed frame, the perturbed matrix is given by

$$\varDelta\boldsymbol{T} = \boldsymbol{T}{}^{\mathrm{R}}\boldsymbol{F} \tag{6.7}$$

where ${}^{\mathrm{R}}\boldsymbol{F}$ is termed **right error matrix**.

On the other hand, the two expressions in eqs. (6.6) and (6.7) must be equal for the perturbed matrix $\boldsymbol{T} + \varDelta\boldsymbol{T}$ is unique irrespective of the representation: the condition

$${}^{\mathrm{L}}\boldsymbol{F}\boldsymbol{T} = \boldsymbol{T}{}^{\mathrm{R}}\boldsymbol{F}$$

gives the two fold relationship

$${}^{\mathrm{L}}\boldsymbol{F} = \boldsymbol{T}{}^{\mathrm{R}}\boldsymbol{F}\boldsymbol{T}^{-1} \tag{6.8a}$$

$${}^{\mathrm{R}}\boldsymbol{F} = \boldsymbol{T}^{-1}{}^{\mathrm{L}}\boldsymbol{F}\boldsymbol{T} \tag{6.8b}$$

which can be used to switch between left and right errors.

6.1.3 Infinitesimal Error

The structure of the full error matrix can be investigated expanding eq. (6.4) as

$$\boldsymbol{F} = \begin{pmatrix} \boldsymbol{R}(\boldsymbol{u}, \varDelta\theta) & \boldsymbol{v}\varDelta t \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} - \begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}(\boldsymbol{u}, \varDelta\theta) - \boldsymbol{I} & \boldsymbol{v}\varDelta t \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix} \tag{6.9}$$

that is partitioning the homogeneous transformation matrices into their corresponding blocks; first, the full error matrix is not a homogeneous transformation matrix anymore, as $f_{44} = 0$. If the rotation matrix is represented as in eq. (B.13)

$$\boldsymbol{R}(\boldsymbol{u}, \Delta\theta) = \boldsymbol{I} + \sin(\Delta\theta)\,\boldsymbol{S}(\boldsymbol{u}) + (1 - \cos(\Delta\theta))\,\boldsymbol{S}^2(\boldsymbol{u})$$

then the rotation block of the full error matrix is given by

$$\boldsymbol{R}(\boldsymbol{u}, \Delta\theta) - \boldsymbol{I} = \sin(\Delta\theta)\,\boldsymbol{S}(\boldsymbol{u}) + (1 - \cos(\Delta\theta))\,\boldsymbol{S}^2(\boldsymbol{u})$$

which is indeed the matrix giving only the modification, due to the rotation, of the vector it multiplies, therefore it does not represent an actual rotation.[1]

As already mentioned before, this discussion derives its meaning from the fact that the perturbation is typically moderate: in that case, the angular displacement $\Delta\theta$ is *relatively* small. The following fundamental trigonometric limits

$$\begin{aligned}\sin(\Delta\theta) &= \Delta\theta + \mathrm{o}(\Delta\theta) \;\; \Delta\theta \to 0\\ 1 - \cos(\Delta\theta) &= \frac{1}{2}\Delta\theta^2 + \mathrm{o}\left(\Delta\theta^2\right), \;\; \Delta\theta \to 0\end{aligned}$$

may be employed to write the asymptotic behaviour of the error rotation matrix

$$\boldsymbol{R}(\boldsymbol{u}\Delta\theta) - \boldsymbol{I} \sim \Delta\theta\boldsymbol{S}(\boldsymbol{u}) + \frac{1}{2}\Delta\theta^2\boldsymbol{S}^2(\boldsymbol{u}) = \boldsymbol{S}(\boldsymbol{u}\Delta\theta) + \frac{1}{2}\boldsymbol{S}^2(\boldsymbol{u}\Delta\theta), \quad \Delta\theta \to 0$$

as the rotation angle gets smaller and smaller. Embedding it within eq. (6.9) leads to the asymptotic error matrix $\boldsymbol{E}^{\mathrm{II}}$

$$\boldsymbol{F} \sim \boldsymbol{E}^{\mathrm{II}} = \begin{pmatrix} \boldsymbol{S}(\boldsymbol{u}\Delta\theta) + \frac{1}{2}\boldsymbol{S}^2(\boldsymbol{u}\Delta\theta) & \boldsymbol{v}\Delta t \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix}, \quad \Delta\theta \to 0$$

where $\boldsymbol{E}^{\mathrm{II}}$ represents the second order approximation of the full error matrix.

Neglecting also the second order infinitesimal in the rotation block

$$\boldsymbol{R}(\boldsymbol{u}\Delta\theta) - \boldsymbol{I} \sim \boldsymbol{S}(\boldsymbol{u}\Delta\theta), \quad \Delta\theta \to 0$$

gives the asymptotic error matrix $\boldsymbol{E}^{\mathrm{I}}$

$$\boldsymbol{F} \sim \boldsymbol{E}^{\mathrm{I}} = \begin{pmatrix} \boldsymbol{S}(\boldsymbol{u}\Delta\theta) & \boldsymbol{v}\Delta t \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix}, \quad \Delta\theta \to 0$$

where $\boldsymbol{E}^{\mathrm{I}}$ represents the first order approximation of the full error matrix and is called **infinitesimal error matrix**. Whenever the assumption of first order approximation is quite clear from the context, the roman superscript is dropped to streamline the notation.

[1] In general, the difference between two rotation matrices is *not* a rotation matrix anymore.

The rotation axis and the translation direction are unit vectors, therefore the vectors defined as follows

$$\begin{aligned}\boldsymbol{\Delta\theta} &\coloneqq \Delta\theta\boldsymbol{u}\\ \boldsymbol{\Delta t} &\coloneqq \Delta t\boldsymbol{v}\end{aligned}$$

have respectively the rotation angle and the translation distance as magnitude. The infinitesimal error matrix may be thus succinctly written as

$$\boldsymbol{E} = \begin{pmatrix}\boldsymbol{S}(\boldsymbol{\Delta\theta}) & \boldsymbol{\Delta t}\\ \boldsymbol{0}^{\mathsf{T}} & 0\end{pmatrix} \tag{6.10}$$

using only two infinitesimal vectors.

Single Axis Procedure

The nature of the multiplicative error might be also investigated using individual axis transformations, i.e. dealing with each coordinate axis *separately*. To this aim, eq. (6.2) can be written as

$$\boldsymbol{T} + \boldsymbol{\Delta T} = \left(\mathbf{Trans}\left(e_x, e_y, e_z\right)\mathbf{Rot}\left(\varepsilon_x, \varepsilon_y, \varepsilon_z\right)\right)\boldsymbol{T}$$

while eq. (6.4) can be written as

$$\boldsymbol{F} \coloneqq \mathbf{Trans}\left(e_x, e_y, e_z\right)\mathbf{Rot}\left(\varepsilon_x, \varepsilon_y, \varepsilon_z\right) - \boldsymbol{I} \tag{6.11}$$

where

$$\begin{aligned}\mathbf{Trans}\left(e_x, e_y, e_z\right) &\coloneqq \mathbf{Trans}\left(\boldsymbol{i}, e_x\right)\mathbf{Trans}\left(\boldsymbol{j}, e_y\right)\mathbf{Trans}\left(\boldsymbol{k}, e_z\right)\\ \mathbf{Rot}\left(\varepsilon_x, \varepsilon_y, \varepsilon_z\right) &\coloneqq \mathbf{Rot}\left(\boldsymbol{i}, \varepsilon_x\right)\mathbf{Rot}\left(\boldsymbol{j}, \varepsilon_y\right)\mathbf{Rot}\left(\boldsymbol{k}, \varepsilon_z\right)\end{aligned}$$

are two homogeneous transformation matrices based on a sequence of translations along and rotations about each coordinate axis one after the other.

The block structure of the multiplicative error matrix can be thus written as

$$\mathbf{Trans}\left(e_x, e_y, e_z\right)\mathbf{Rot}\left(\varepsilon_x, \varepsilon_y, \varepsilon_z\right) = \begin{pmatrix}\boldsymbol{R}(\boldsymbol{i}, \varepsilon_x)\,\boldsymbol{R}\big(\boldsymbol{j}, \varepsilon_y\big)\,\boldsymbol{R}(\boldsymbol{k}, \varepsilon_z) & e_x\boldsymbol{i} + e_y\boldsymbol{j} + e_z\boldsymbol{k}\\ \boldsymbol{0}^{\mathsf{T}} & 1\end{pmatrix}$$

remembering eq. (B.26). As before, the attention must be focused on the rotation block only: as the angles of rotation get smaller and smaller

$$\boldsymbol{R}(\boldsymbol{i}, \varepsilon_x) = \begin{pmatrix}1 & 0 & 0\\ 0 & +\mathrm{c}_{\varepsilon_x} & -\mathrm{s}_{\varepsilon_x}\\ 0 & +\mathrm{s}_{\varepsilon_x} & +\mathrm{c}_{\varepsilon_x}\end{pmatrix} \sim \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & -\varepsilon_x\\ 0 & +\varepsilon_x & 1\end{pmatrix} = \boldsymbol{I} + \boldsymbol{S}(\varepsilon_x\boldsymbol{i})\,, \quad \varepsilon_x \to 0$$

$$\boldsymbol{R}\big(\boldsymbol{j}, \varepsilon_y\big) = \begin{pmatrix}+\mathrm{c}_{\varepsilon_y} & 0 & +\mathrm{s}_{\varepsilon_y}\\ 0 & 1 & 0\\ -\mathrm{s}_{\varepsilon_y} & 0 & +\mathrm{c}_{\varepsilon_y}\end{pmatrix} \sim \begin{pmatrix}1 & 0 & +\varepsilon_y\\ 0 & 1 & 0\\ -\varepsilon_y & 0 & 1\end{pmatrix} = \boldsymbol{I} + \boldsymbol{S}\big(\varepsilon_y\boldsymbol{j}\big)\,, \quad \varepsilon_y \to 0$$

$$\boldsymbol{R}(\boldsymbol{k}, \varepsilon_z) = \begin{pmatrix}+\mathrm{c}_{\varepsilon_z} & -\mathrm{s}_{\varepsilon_z} & 0\\ +\mathrm{s}_{\varepsilon_z} & +\mathrm{c}_{\varepsilon_z} & 0\\ 0 & 0 & 1\end{pmatrix} \sim \begin{pmatrix}1 & -\varepsilon_z & 0\\ +\varepsilon_z & 1 & 0\\ 0 & 0 & 1\end{pmatrix} = \boldsymbol{I} + \boldsymbol{S}(\varepsilon_z\boldsymbol{k})\,, \quad \varepsilon_z \to 0$$

all elementary rotation matrices become infinitesimal. Performing the product, the asymptotic behaviour of the error matrix rotation block is given by

$$\begin{aligned}\boldsymbol{R}(\boldsymbol{i},\varepsilon_x)\,\boldsymbol{R}\big(\boldsymbol{j},\varepsilon_y\big)\,\boldsymbol{R}(\boldsymbol{k},\varepsilon_z)-\boldsymbol{I} &\sim \left(\boldsymbol{I}+\boldsymbol{S}(\varepsilon_x\boldsymbol{i})\right)\left(\boldsymbol{I}+\boldsymbol{S}\big(\varepsilon_y\boldsymbol{j}\big)\right)\left(\boldsymbol{I}+\boldsymbol{S}(\varepsilon_z\boldsymbol{k})\right)-\boldsymbol{I}\sim\\ &\sim \boldsymbol{I}+\boldsymbol{S}(\varepsilon_x\boldsymbol{i})+\boldsymbol{S}\big(\varepsilon_y\boldsymbol{j}\big)+\boldsymbol{S}(\varepsilon_z\boldsymbol{k})-\boldsymbol{I}=\\ &= \boldsymbol{S}\big(\varepsilon_x\boldsymbol{i}+\varepsilon_y\boldsymbol{j}+\varepsilon_z\boldsymbol{k}\big)\,,\quad \varepsilon_x\to 0,\varepsilon_y\to 0,\varepsilon_z\to 0\end{aligned}$$

because all higher order terms may be neglected. Packing the six error variables in two vectors

$$\boldsymbol{e}\coloneqq\begin{pmatrix}e_x\\e_y\\e_z\end{pmatrix},\quad \boldsymbol{\varepsilon}\coloneqq\begin{pmatrix}\varepsilon_x\\\varepsilon_y\\\varepsilon_z\end{pmatrix}$$

the error matrix first order asymptotic behaviour can be compactly written as

$$\boldsymbol{F}\sim\boldsymbol{E}^{\mathrm{I}}=\begin{pmatrix}\boldsymbol{S}(\boldsymbol{\varepsilon}) & \boldsymbol{e}\\ \boldsymbol{0}^{\mathsf{T}} & 0\end{pmatrix},\ \boldsymbol{\varepsilon}\to\boldsymbol{0}$$

where $\boldsymbol{E}^{\mathrm{I}}$ represents the first order approximation of the full error matrix.

It is worth noting that the infinitesimal error matrix

$$\boldsymbol{E}\coloneqq\begin{pmatrix}\boldsymbol{S}(\boldsymbol{\varepsilon}) & \boldsymbol{e}\\ \boldsymbol{0}^{\mathsf{T}} & 0\end{pmatrix}\tag{6.12}$$

can be obtained performing the substitution

$$\boldsymbol{\Delta t}=\boldsymbol{e},\ \boldsymbol{\Delta\theta}=\boldsymbol{\varepsilon}$$

into the expression of the infinitesimal error matrix given by eq. (6.10). As far as first order approximation is concerned, the sequence of elemental rotations is not relevant *whatsoever*, therefore both forms of the infinitesimal error matrix may be be used interchangeably. For this reason, the structure given by eq. (6.12) is universally employed, *regardless* of the actual attitude representation behind it. Whenever the kinematic error is small enough, eq. (6.5) can be approximated as

$$\boldsymbol{\Delta T}\approx\boldsymbol{ET}\tag{6.13}$$

using the infinitesimal error matrix in place of the full error matrix.

6.1.4 Rigid Body Kinematic Error

As mentioned at the beginning, homogeneous transformation matrices can be exploited to describe the pose of rigid bodies. Specifically, if a certain rigid body is kinematically described by means of the homogeneous transformation matrix

$$\boldsymbol{T}=\begin{pmatrix}\boldsymbol{R} & \boldsymbol{t}\\ \boldsymbol{0}^{\mathsf{T}} & 1\end{pmatrix}$$

then the rotation matrix $\boldsymbol{R}$ represents its orientation and the translation vector $\boldsymbol{t}$ represents its position. If at least one of them is subject to a change, the homogeneous transformation matrix changes accordingly, becoming

$$\boldsymbol{T} + \Delta\boldsymbol{T} = \begin{pmatrix} \boldsymbol{R} + \Delta\boldsymbol{R} & \boldsymbol{t} + \Delta\boldsymbol{t} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix}$$

where

$$\Delta\boldsymbol{T} \coloneqq \begin{pmatrix} \Delta\boldsymbol{R} & \Delta\boldsymbol{t} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix} \tag{6.14}$$

is the homogeneous transformation matrix variation due to an additive error in both orientation and position. Performing the product in eq. (6.13) blockwise

$$\boldsymbol{E}\boldsymbol{T} = \begin{pmatrix} \boldsymbol{S}(\boldsymbol{\varepsilon}) & \boldsymbol{e} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{S}(\boldsymbol{\varepsilon})\,\boldsymbol{R} & \boldsymbol{S}(\boldsymbol{\varepsilon})\,\boldsymbol{t} + \boldsymbol{e} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix}$$

and comparing it with eq. (6.14) gives the approximate orientation error

$$\Delta\boldsymbol{R} \approx \boldsymbol{S}(\boldsymbol{\varepsilon})\,\boldsymbol{R} \tag{6.15}$$

which is due to infinitesimal rotation *only*, and the approximate position error

$$\Delta\boldsymbol{t} \approx \boldsymbol{S}(\boldsymbol{\varepsilon})\,\boldsymbol{t} + \boldsymbol{e} \tag{6.16}$$

which is due to *both* infinitesimal rotation and translation.

6.1.5 Reference Frame Kinematic Error

On the other hand, as explained in subsection B.3.2, homogeneous transformation matrices can be used also to describe the change of reference frame. Specifically, considering a point P and two reference frames $\mathcal{R}_{\mathrm{A}}$ and $\mathcal{R}_{\mathrm{B}}$, since

$$\boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} = \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} \tag{6.17}$$

the matrix $\boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}}$ transforms the homogeneous coordinates of point P in $\mathcal{R}_{\mathrm{B}}$ into the homogeneous coordinates of point P in $\mathcal{R}_{\mathrm{A}}$. If the relationship between the two reference frames changes, the homogeneous transformation matrix reflects this change: the homogeneous coordinates in $\mathcal{R}_{\mathrm{A}}$ are then modified *accordingly*

$$\boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} + \Delta\boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} \coloneqq \left(\boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} + \Delta\boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}}\right) \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} = \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} + \Delta\boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} = \boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} + \Delta\boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}}$$

therefore the homogeneous coordinates variation

$$\Delta\boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} \coloneqq \Delta\boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} \tag{6.18}$$

is proportional to the homogeneous transformation matrix variation. Substituting

$$\Delta \boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} \coloneqq \begin{pmatrix} \Delta \boldsymbol{r}_{\mathrm{P}}^{\mathrm{A}} \\ 0 \end{pmatrix}, \quad \Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \coloneqq \begin{pmatrix} \Delta \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}} & \Delta \boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix}$$

into eq. (6.18), gives

$$\Delta \boldsymbol{r}_{\mathrm{P}}^{\mathrm{A}} = \Delta \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{r}_{\mathrm{P}}^{\mathrm{B}} + \Delta \boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \tag{6.19}$$

which is the **position error** of point P in $\mathcal{R}_{\mathrm{A}}$.

With regard to the multiplicative error representation, the error matrices are denoted using the label of the frame in which the modification occurs, as superscript

$$\Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} = {}^{\mathrm{A}}\boldsymbol{F} \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \tag{6.20a}$$

$$\Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} = \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} {}^{\mathrm{B}}\boldsymbol{F} \tag{6.20b}$$

giving then two alternate expressions

$$\Delta \boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} = \Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} = {}^{\mathrm{A}}\boldsymbol{F} \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} = {}^{\mathrm{A}}\boldsymbol{F} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} \tag{6.21a}$$

$$\Delta \boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} = \Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} = \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} {}^{\mathrm{B}}\boldsymbol{F} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} = \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \left({}^{\mathrm{B}}\boldsymbol{F} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} \right) \tag{6.21b}$$

of eq. (6.18). Using eq. (6.8), the two error matrices can be thus related by

$${}^{\mathrm{A}}\boldsymbol{F} = \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} {}^{\mathrm{B}}\boldsymbol{F} \boldsymbol{T}_{\mathrm{A}}^{\mathrm{B}}$$

$${}^{\mathrm{B}}\boldsymbol{F} = \boldsymbol{T}_{\mathrm{A}}^{\mathrm{B}} {}^{\mathrm{A}}\boldsymbol{F} \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}}$$

that is a special case of reference frame matrix transformation defined in eq. (B.35).

Infinitesimal Position Error

Using the same fashion of eq. (6.13), it is possible to approximate eq. (6.20) as

$$\Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \approx {}^{\mathrm{A}}\boldsymbol{E} \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \tag{6.22a}$$

$$\Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \approx \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} {}^{\mathrm{B}}\boldsymbol{E} \tag{6.22b}$$

and, consequently, also eq. (6.21) as

$$\Delta \boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} = \Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} \approx {}^{\mathrm{A}}\boldsymbol{E} \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} = {}^{\mathrm{A}}\boldsymbol{E} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} \tag{6.23a}$$

$$\Delta \boldsymbol{h}_{\mathrm{P}}^{\mathrm{A}} = \Delta \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} \approx \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} {}^{\mathrm{B}}\boldsymbol{E} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} = \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}} \left({}^{\mathrm{B}}\boldsymbol{E} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{B}} \right) \tag{6.23b}$$

where

$${}^{\mathrm{A}}\boldsymbol{E} \coloneqq \begin{pmatrix} \boldsymbol{S}\left({}^{\mathrm{A}}\boldsymbol{\varepsilon}\right) & {}^{\mathrm{A}}\boldsymbol{e} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix}, \; {}^{\mathrm{B}}\boldsymbol{E} \coloneqq \begin{pmatrix} \boldsymbol{S}\left({}^{\mathrm{B}}\boldsymbol{\varepsilon}\right) & {}^{\mathrm{B}}\boldsymbol{e} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix} \tag{6.24}$$

are respectively the infinitesimal error matrices in $\mathcal{R}_{\mathrm{A}}$ and $\mathcal{R}_{\mathrm{B}}$. Performing the products in eq. (6.23) blockwise finally gives the following two approximations

$$\Delta \boldsymbol{r}_{\mathrm{P}}^{\mathrm{A}} \approx \boldsymbol{S}\left({}^{\mathrm{A}}\boldsymbol{\varepsilon}\right) \boldsymbol{r}_{\mathrm{P}}^{\mathrm{A}} + {}^{\mathrm{A}}\boldsymbol{e} \tag{6.25a}$$

$$\Delta \boldsymbol{r}_{\mathrm{P}}^{\mathrm{A}} \approx \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}} \left(\boldsymbol{S}\left({}^{\mathrm{B}}\boldsymbol{\varepsilon}\right) \boldsymbol{r}_{\mathrm{P}}^{\mathrm{B}} + {}^{\mathrm{B}}\boldsymbol{e} \right) \tag{6.25b}$$

of eq. (6.19).

Sometimes the kinematic error is easily computed in a certain frame, but, for some reason, it must be expressed in another frame, thus a way to transform error vectors between frames is needed. To this aim, equating the right hand sides of eq. (6.22) gives

$$ {}^{\mathrm{A}}\boldsymbol{E} = \boldsymbol{T}_{\mathrm{B}}^{\mathrm{A}\,\mathrm{B}}\boldsymbol{E}\boldsymbol{T}_{\mathrm{A}}^{\mathrm{B}} \tag{6.26} $$

which is again a special case of eq. (B.35). Remembering, from eq. (B.34), that

$$ \boldsymbol{T}_{\mathrm{A}}^{\mathrm{B}} = \begin{pmatrix} \boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}} & -\boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}}\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} $$

eq. (6.26) may be expanded as

$$ \begin{pmatrix} \boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{e}\big) & {}^{\mathrm{A}}\boldsymbol{e} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}} & \boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{S}\big({}^{\mathrm{B}}\boldsymbol{e}\big) & {}^{\mathrm{B}}\boldsymbol{e} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}} & -\boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}}\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \\ \boldsymbol{0}^{\mathsf{T}} & 1 \end{pmatrix} = $$

$$ = \begin{pmatrix} \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\boldsymbol{S}\big({}^{\mathrm{B}}\boldsymbol{\varepsilon}\big)\boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}} & \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}\,\mathrm{B}}\boldsymbol{e} - \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\boldsymbol{S}\big({}^{\mathrm{B}}\boldsymbol{\varepsilon}\big)\boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}}\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix} = \begin{pmatrix} \boldsymbol{S}\big(\boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}\,\mathrm{B}}\boldsymbol{\varepsilon}\big) & \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}\,\mathrm{B}}\boldsymbol{e} - \boldsymbol{S}\big(\boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}\,\mathrm{B}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \\ \boldsymbol{0}^{\mathsf{T}} & 0 \end{pmatrix} $$

where eq. (A.40) has been used. Comparing the matrices corresponding blocks gives the transformations between error vectors from $\mathcal{R}_{\mathrm{B}}$ to $\mathcal{R}_{\mathrm{A}}$

$$ {}^{\mathrm{A}}\boldsymbol{\varepsilon} = \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}\,\mathrm{B}}\boldsymbol{\varepsilon} \tag{6.27a} $$

$$ {}^{\mathrm{A}}\boldsymbol{e} = \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}\,\mathrm{B}}\boldsymbol{e} - \boldsymbol{S}\big(\boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}\,\mathrm{B}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \tag{6.27b} $$

or, vice versa, from $\mathcal{R}_{\mathrm{A}}$ to $\mathcal{R}_{\mathrm{B}}$

$$ {}^{\mathrm{B}}\boldsymbol{\varepsilon} = \boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}\,\mathrm{A}}\boldsymbol{\varepsilon} \tag{6.28a} $$

$$ {}^{\mathrm{B}}\boldsymbol{e} = \boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}\,\mathrm{A}}\boldsymbol{e} + \boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}}\boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} \tag{6.28b} $$

applying few manipulations. Such a result is perfectly consistent with what had been already found before: substituting eq. (6.28) into the right hand side of eq. (6.25b)

$$ \begin{aligned} \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\left(\boldsymbol{S}\big({}^{\mathrm{B}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{r}_{\mathrm{P}}^{\mathrm{B}} + {}^{\mathrm{B}}\boldsymbol{e}\right) &= \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\Big(\boldsymbol{S}\big(\boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}\,\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{r}_{\mathrm{P}}^{\mathrm{B}} + \boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}\,\mathrm{A}}\boldsymbol{e} + \boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}}\boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}}\Big) = \\ &= \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}}\boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\boldsymbol{r}_{\mathrm{P}}^{\mathrm{B}} + \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}\,\mathrm{A}}\boldsymbol{e} + \boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\boldsymbol{R}_{\mathrm{A}}^{\mathrm{B}}\boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} = \\ &= \boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\boldsymbol{r}_{\mathrm{P}}^{\mathrm{B}} + {}^{\mathrm{A}}\boldsymbol{e} + \boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}} = \\ &= \boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\big(\boldsymbol{R}_{\mathrm{B}}^{\mathrm{A}}\boldsymbol{r}_{\mathrm{P}}^{\mathrm{B}} + \boldsymbol{t}_{\mathrm{B}}^{\mathrm{A}}\big) + {}^{\mathrm{A}}\boldsymbol{e} = \boldsymbol{S}\big({}^{\mathrm{A}}\boldsymbol{\varepsilon}\big)\,\boldsymbol{r}_{\mathrm{P}}^{\mathrm{A}} + {}^{\mathrm{A}}\boldsymbol{e} \end{aligned} $$

returns indeed the right hand side of eq. (6.25a).

6.1.6 Differential Error Model

The derivative of a matrix is the matrix having, as entries, the derivative of the corresponding matrix entries. The derivative of a homogeneous transformation matrix $\boldsymbol{T}$ with respect to a certain independent variable ξ is thus simply

$$\frac{\partial \boldsymbol{T}}{\partial \xi} \coloneqq \begin{pmatrix} \frac{\partial t_{11}}{\partial \xi} & \frac{\partial t_{12}}{\partial \xi} & \frac{\partial t_{13}}{\partial \xi} & \frac{\partial t_{14}}{\partial \xi} \\ \frac{\partial t_{21}}{\partial \xi} & \frac{\partial t_{22}}{\partial \xi} & \frac{\partial t_{23}}{\partial \xi} & \frac{\partial t_{24}}{\partial \xi} \\ \frac{\partial t_{31}}{\partial \xi} & \frac{\partial t_{32}}{\partial \xi} & \frac{\partial t_{33}}{\partial \xi} & \frac{\partial t_{34}}{\partial \xi} \\ \frac{\partial t_{41}}{\partial \xi} & \frac{\partial t_{42}}{\partial \xi} & \frac{\partial t_{43}}{\partial \xi} & \frac{\partial t_{44}}{\partial \xi} \end{pmatrix}$$

where t_{ij} is the entry of $\boldsymbol{T}$ in the i-th row and the j-th column. Despite its simplicity, this might not the best choice to differentiate homogeneous transformation matrices, for it requires the computation of 16 different derivatives all *at once*; in the scope of differential kinematic modelling, a different approach should be followed instead.

To this aim, the homogeneous transformation matrix derivative can be also written according to the definition of derivative, i.e. as limit of the difference quotient

$$\frac{\partial \boldsymbol{T}}{\partial \xi} = \lim_{\Delta\xi \to 0} \frac{\Delta \boldsymbol{T}}{\Delta \xi} = \lim_{\Delta\xi \to 0} \frac{\boldsymbol{T}(\xi + \Delta\xi) - \boldsymbol{T}(\xi)}{\Delta \xi} \tag{6.29}$$

where $\Delta\boldsymbol{T}$ is the matrix increment due to the variable increment $\Delta\xi$, already defined in eq. (6.1). Such an expression suggest to exploit the theory developed so far about homogeneous transformation matrix infinitesimal variations. Making use of the multiplicative error notation of eq. (6.5), the matrix change can be expressed as

$$\Delta \boldsymbol{T} = \boldsymbol{T}(\xi + \Delta\xi) - \boldsymbol{T}(\xi) = \boldsymbol{F}(\xi, \Delta\xi)\, \boldsymbol{T}(\xi)$$

the error matrix being the only term depending on the variable change. As the variable increment gets smaller and smaller, the matrix increment approaches

$$\boldsymbol{T}(\xi + \Delta\xi) - \boldsymbol{T}(\xi) \sim \boldsymbol{E}(\Delta\xi)\, \boldsymbol{T}(\xi)\,,\ \Delta\xi \to 0$$

because the error matrix asymptotically depends upon the variable change only. The infinitesimal error matrix may be written as a linear function of the variable change

$$\boldsymbol{E}(\Delta\xi) \coloneqq \boldsymbol{D}_{\xi} \Delta\xi \tag{6.30}$$

since only the first order behaviour is considered. Substituting this into eq. (6.29) gives

$$\lim_{\Delta\xi \to 0} \frac{\boldsymbol{T}(\xi + \Delta\xi) - \boldsymbol{T}(\xi)}{\Delta \xi} = \lim_{\Delta\xi \to 0} \frac{\boldsymbol{E}(\Delta\xi)\, \boldsymbol{T}(\xi)}{\Delta \xi} = \lim_{\Delta\xi \to 0} \frac{\boldsymbol{D}_{\xi} \Delta\xi}{\Delta \xi} \boldsymbol{T}(\xi) = \boldsymbol{D}_{\xi} \boldsymbol{T}(\xi)$$

hence the homogeneous transformation matrix derivative can be finally written as

$$\frac{\partial \boldsymbol{T}}{\partial \xi} = \boldsymbol{D}_{\xi} \boldsymbol{T}(\xi) \tag{6.31}$$

using the same compact fashion of eq. (6.13).

When the homogeneous transformation matrix change is due to an infinitesimal change of a given variable ξ, the infinitesimal error matrix is called differential error matrix and is denoted by $\boldsymbol{E}_\xi$, in order to stress the variable responsible for that error. It can be written, using the differentials notation, as

$$\boldsymbol{E}_\xi \coloneqq \boldsymbol{D}_\xi \mathrm{d}\xi \tag{6.32}$$

where $\boldsymbol{D}_\xi$ is termed **left differential matrix**: it is the matrix which yields the **left differential error matrix** $\boldsymbol{E}_\xi$ when multiplied by the differential variable change $\mathrm{d}\xi$. As put in evidence by the partial differential

$$\mathrm{d}\boldsymbol{T} = \frac{\partial \boldsymbol{T}}{\partial \xi}\mathrm{d}\xi = \boldsymbol{D}_\xi \boldsymbol{T}\mathrm{d}\xi = \boldsymbol{E}_\xi \boldsymbol{T} \tag{6.33}$$

the differential error matrix and the differential matrix play a similar role but the former is an absolute quantity while the latter is a relative quantity. Moreover, the two matrices clearly must share the same block structure: denoting the differential error matrix as

$$\boldsymbol{E}_\xi \coloneqq \begin{pmatrix} \boldsymbol{S}(\boldsymbol{\varepsilon}_\xi) & \boldsymbol{e}_\xi \\ \boldsymbol{0}^\mathsf{T} & 0 \end{pmatrix}$$

and the differential matrix as

$$\boldsymbol{D}_\xi \coloneqq \begin{pmatrix} \boldsymbol{S}(\boldsymbol{\delta}_\xi) & \boldsymbol{d}_\xi \\ \boldsymbol{0}^\mathsf{T} & 0 \end{pmatrix}$$

where

$$\boldsymbol{\varepsilon}_\xi = \boldsymbol{\delta}_\xi \mathrm{d}\xi \tag{6.34a}$$

$$\boldsymbol{e}_\xi = \boldsymbol{d}_\xi \mathrm{d}\xi \tag{6.34b}$$

because of eq. (6.32), gives the **differential orientation error**

$$\mathrm{d}\boldsymbol{R} = \boldsymbol{S}(\boldsymbol{\varepsilon}_\xi)\,\boldsymbol{R} = \boldsymbol{S}(\boldsymbol{\delta}_\xi)\,\boldsymbol{R}\mathrm{d}\xi \tag{6.35}$$

and the **differential position error**

$$\mathrm{d}\boldsymbol{t} = \boldsymbol{S}(\boldsymbol{\varepsilon}_\xi)\,\boldsymbol{t} + \boldsymbol{e}_\xi = \boldsymbol{S}(\boldsymbol{\delta}_\xi)\,\boldsymbol{t}\mathrm{d}\xi + \boldsymbol{d}_\xi \mathrm{d}\xi \tag{6.36}$$

representing the differential version of of eqs. (6.15) and (6.16), respectively.

6.2 Manipulator Pose Error Model

6.2.1 Kinematic Modelling Convention

The kinematic model of any manipulator requires the selection of some conventions to represent the transformation between each pair of adjacent link frames through a

suitable set of parameters. As thoroughly explained in section 4.2, the model chosen for the calibration of articulated manipulators is based upon the combination of two conventions, namely Denavit-Hartenberg, handling quasi-orthogonal joint motion axes and Hayati-Mirmirani, handling quasi-parallel joint motion axes.

The following augmented transformation with five kinematic parameters

$$\boldsymbol{T}_{\mathrm{VW}} = \begin{pmatrix} \mathrm{c}_\theta\mathrm{c}_\beta - \mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{s}_\beta & -\mathrm{s}_\theta\mathrm{c}_\alpha & \mathrm{c}_\theta\mathrm{s}_\beta + \mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{c}_\beta & a\mathrm{c}_\theta \\ \mathrm{s}_\theta\mathrm{c}_\beta + \mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{s}_\beta & \mathrm{c}_\theta\mathrm{c}_\alpha & \mathrm{s}_\theta\mathrm{s}_\beta - \mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{c}_\beta & a\mathrm{s}_\theta \\ -\mathrm{c}_\alpha\mathrm{s}_\beta & \mathrm{s}_\alpha & \mathrm{c}_\alpha\mathrm{c}_\beta & d \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.37}$$

based on Veitschegger-Wu convention[2], is chosen as basis to develop the **differential kinematic model**, for it embeds both target conventions: zeroing the parameter β

$$\boldsymbol{T}_{\mathrm{VW}}\big|_{\beta=0} = \begin{pmatrix} \mathrm{c}_\theta & -\mathrm{s}_\theta\mathrm{c}_\alpha & \mathrm{s}_\theta\mathrm{s}_\alpha & a\mathrm{c}_\theta \\ \mathrm{s}_\theta & \mathrm{c}_\theta\mathrm{c}_\alpha & -\mathrm{c}_\theta\mathrm{s}_\alpha & a\mathrm{s}_\theta \\ 0 & \mathrm{s}_\alpha & \mathrm{c}_\alpha & d \\ 0 & 0 & 0 & 1 \end{pmatrix} \equiv \boldsymbol{T}_{\mathrm{DH}}$$

Veitschegger-Wu becomes Denavit-Hartenberg, while zeroing the parameter d

$$\boldsymbol{T}_{\mathrm{VW}}\big|_{d=0} = \begin{pmatrix} \mathrm{c}_\theta\mathrm{c}_\beta - \mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{s}_\beta & -\mathrm{s}_\theta\mathrm{c}_\alpha & \mathrm{c}_\theta\mathrm{s}_\beta + \mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{c}_\beta & a\mathrm{c}_\theta \\ \mathrm{s}_\theta\mathrm{c}_\beta + \mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{s}_\beta & \mathrm{c}_\theta\mathrm{c}_\alpha & \mathrm{s}_\theta\mathrm{s}_\beta - \mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{c}_\beta & a\mathrm{s}_\theta \\ -\mathrm{c}_\alpha\mathrm{s}_\beta & \mathrm{s}_\alpha & \mathrm{c}_\alpha\mathrm{c}_\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \equiv \boldsymbol{T}_{\mathrm{HM}}$$

Veitschegger-Wu becomes Hayati-Mirmirani. For the sake of simplicity, such a homogeneous transformation matrix is hereinafter indicated merely as $\boldsymbol{T}$.

The adopted choice provides both flexibility, for the transformation adapts to each pair of links, and uniformity, for a unique transformation is needed for all pairs of links. Case by case, the kinematic parameter not involved in the transformation is set to zero and the corresponding derivative is simply discarded.

6.2.2 Individual Joint Kinematic Error Model

In principle, any of the five kinematic parameters involved in the generalised transformation defined by eq. (6.37) may vary, resulting in as many corresponding changes of that kinematic transformation; for this reason, in order to build a complete parametric kinematic error model, it is necessary to evaluate the derivative of the homogeneous transformation matrix with respect to each kinematic parameter.

To this aim, the approach based on the differential error described in subsection 6.1.6 is here followed. For each kinematic parameter, the differential orientation and position error vectors, due to a differential change of that parameter, are computed, performing, if

[2]The non-minimal five parameters Veitschegger-Wu convention, named after the authors of [11] and [19], is nothing but a mixture of Denavit-Hardenberg and Hayati-Mirmirani conventions.

necessary, the transformations provided by eq. (6.26) to express them in the right frame; the resulting differential matrix is then multiplied by the homogeneous transformation matrix itself to give the desired derivative.

Differentiation with respect to d parameter

The kinematic parameter d is the length of translation along the original z axis, therefore the differential increment $\mathrm{d}\theta$ induces the differential orientation error vector

$$\boldsymbol{\varepsilon}_d = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

and the differential position error vector

$$\boldsymbol{e}_d = \begin{pmatrix} 0 \\ 0 \\ \mathrm{d}d \end{pmatrix}$$

that give rise respectively to

$$\boldsymbol{\delta}_d = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad \boldsymbol{d}_d = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

using eq. (6.34). Embedding them in the differential matrix

$$\boldsymbol{D}_d = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.38}$$

the partial derivative with respect to d can be easily computed as

$$\frac{\partial \boldsymbol{T}}{\partial d} = \boldsymbol{D}_d \boldsymbol{T} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

thanks to eq. (6.31).

Differentiation with respect to θ parameter

The kinematic parameter θ is the angle of rotation about the original z axis, therefore the differential increment $\mathrm{d}\theta$ causes the differential orientation error vector

$$\boldsymbol{\varepsilon}_\theta = \begin{pmatrix} 0 \\ 0 \\ \mathrm{d}\theta \end{pmatrix}$$

and the differential position error vector

$$\boldsymbol{e}_\theta = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

that give rise respectively to

$$\boldsymbol{\delta}_\theta = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad \boldsymbol{d}_\theta = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

using eq. (6.34). Embedding them in the differential matrix

$$\boldsymbol{D}_\theta = \begin{pmatrix} 0 & -1 & 0 & 0 \\ +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.39}$$

the partial derivative with respect to θ can be easily computed as

$$\frac{\partial \boldsymbol{T}}{\partial \theta} = \boldsymbol{D}_\theta \boldsymbol{T} = \begin{pmatrix} -\mathrm{s}_\theta \mathrm{c}_\beta - \mathrm{c}_\theta \mathrm{s}_\alpha \mathrm{s}_\beta & -\mathrm{c}_\theta \mathrm{c}_\alpha & -\mathrm{s}_\theta \mathrm{s}_\beta + \mathrm{c}_\theta \mathrm{s}_\alpha \mathrm{c}_\beta & -a\mathrm{s}_\theta \\ \mathrm{c}_\theta \mathrm{c}_\beta - \mathrm{s}_\theta \mathrm{s}_\alpha \mathrm{s}_\beta & -\mathrm{s}_\theta \mathrm{c}_\alpha & \mathrm{c}_\theta \mathrm{s}_\beta + \mathrm{s}_\theta \mathrm{s}_\alpha \mathrm{c}_\beta & a\mathrm{c}_\theta \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

thanks to eq. (6.31).

Differentiation with respect to a parameter

The kinematic parameter a is the length of translation along the x axis modified by the first rotation about z, therefore the differential increment $\mathrm{d}a$ causes the differential orientation error vector

$$\boldsymbol{\varepsilon}_a = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

and the differential position error vector

$$\boldsymbol{e}_a = \begin{pmatrix} +\mathrm{c}_\theta & -\mathrm{s}_\theta & 0 \\ +\mathrm{s}_\theta & +\mathrm{c}_\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathrm{d}a \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \mathrm{c}_\theta \mathrm{d}a \\ \mathrm{s}_\theta \mathrm{d}a \\ 0 \end{pmatrix}$$

that give rise respectively to

$$\boldsymbol{\delta}_a = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad \boldsymbol{d}_a = \begin{pmatrix} \mathrm{c}_\theta \\ \mathrm{s}_\theta \\ 0 \end{pmatrix}$$

using eq. (6.34). Embedding them in the differential matrix

$$\boldsymbol{D}_a = \begin{pmatrix} 0 & 0 & 0 & \mathrm{c}_\theta \\ 0 & 0 & 0 & \mathrm{s}_\theta \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.40}$$

the partial derivative with respect to a can be easily computed as

$$\frac{\partial \boldsymbol{T}}{\partial a} = \boldsymbol{D}_a \boldsymbol{T} = \begin{pmatrix} 0 & 0 & 0 & \mathrm{c}_\theta \\ 0 & 0 & 0 & \mathrm{s}_\theta \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

thanks to eq. (6.31).

Differentiation with respect to α parameter

The kinematic parameter α is the angle of rotation about the x axis modified by the first rotation about z, therefore, the differential increment $\mathrm{d}\alpha$ causes the differential orientation error vector

$$\boldsymbol{\varepsilon}_\alpha = \begin{pmatrix} +\cos\theta & -\sin\theta & 0 \\ +\sin\theta & +\cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathrm{d}\alpha \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \mathrm{c}_\theta \mathrm{d}\alpha \\ \mathrm{s}_\theta \mathrm{d}\alpha \\ 0 \end{pmatrix}$$

and the differential position error vector

$$\boldsymbol{e}_\alpha = - \begin{pmatrix} 0 & 0 & +\mathrm{s}_\theta \mathrm{d}\alpha \\ 0 & 0 & -\mathrm{c}_\theta \mathrm{d}\alpha \\ -\mathrm{s}_\theta \mathrm{d}\alpha & +\mathrm{c}_\theta \mathrm{d}\alpha & 0 \end{pmatrix} \begin{pmatrix} a\mathrm{c}_\theta \\ a\mathrm{s}_\theta \\ d \end{pmatrix} = \begin{pmatrix} -d\mathrm{s}_\theta \mathrm{d}\alpha \\ +d\mathrm{c}_\theta \mathrm{d}\alpha \\ 0 \end{pmatrix}$$

that give rise respectively to

$$\boldsymbol{\delta}_\alpha = \begin{pmatrix} \mathrm{c}_\theta \\ \mathrm{s}_\theta \\ 0 \end{pmatrix}, \quad \boldsymbol{d}_\alpha = \begin{pmatrix} -d\mathrm{s}_\theta \\ +d\mathrm{c}_\theta \\ 0 \end{pmatrix}$$

using eq. (6.34). Embedding them in the differential matrix

$$\boldsymbol{D}_\alpha = \begin{pmatrix} 0 & 0 & +\mathrm{s}_\theta & -d\mathrm{s}_\theta \\ 0 & 0 & -\mathrm{c}_\theta & +d\mathrm{c}_\theta \\ -\mathrm{s}_\theta & +\mathrm{c}_\theta & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.41}$$

the partial derivative with respect to α can be easily computed as

$$\frac{\partial \boldsymbol{T}}{\partial \alpha} = \boldsymbol{D}_\alpha \boldsymbol{T} = \begin{pmatrix} -\mathrm{s}_\theta \mathrm{c}_\alpha \mathrm{s}_\beta & \mathrm{s}_\theta \mathrm{s}_\alpha & \mathrm{s}_\theta \mathrm{c}_\alpha \mathrm{c}_\beta & 0 \\ \mathrm{c}_\theta \mathrm{c}_\alpha \mathrm{s}_\beta & -\mathrm{c}_\theta \mathrm{s}_\alpha & -\mathrm{c}_\theta \mathrm{c}_\alpha \mathrm{c}_\beta & 0 \\ \mathrm{s}_\alpha \mathrm{s}_\beta & \mathrm{c}_\alpha & -\mathrm{s}_\alpha \mathrm{c}_\beta & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

thanks to eq. (6.31).

Differentiation with respect to β parameter

The kinematic parameter β is the angle of rotation about the y axis modified by the first rotation about z and by the second rotation about x, therefore the differential increment $\mathrm{d}\beta$ causes the differential orientation error vector

$$\boldsymbol{\varepsilon}_\beta = \begin{pmatrix} +\mathrm{c}_\theta & -\mathrm{s}_\theta & 0 \\ +\mathrm{s}_\theta & +\mathrm{c}_\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & +\mathrm{c}_\alpha & -\mathrm{s}_\alpha \\ 0 & +\mathrm{s}_\alpha & +\mathrm{c}_\alpha \end{pmatrix} \begin{pmatrix} 0 \\ \mathrm{d}\beta \\ 0 \end{pmatrix} = \begin{pmatrix} -\mathrm{s}_\theta\mathrm{c}_\alpha\mathrm{d}\beta \\ +\mathrm{c}_\theta\mathrm{c}_\alpha\mathrm{d}\beta \\ \mathrm{s}_\alpha\mathrm{d}\beta \end{pmatrix}$$

and the differential position error vector

$$\boldsymbol{e}_\beta = -\begin{pmatrix} 0 & -\mathrm{s}_\alpha\mathrm{d}\beta & +\mathrm{c}_\theta\mathrm{c}_\alpha\mathrm{d}\beta \\ +\mathrm{s}_\alpha\mathrm{d}\beta & 0 & +\mathrm{s}_\theta\mathrm{c}_\alpha\mathrm{d}\beta \\ -\mathrm{c}_\theta\mathrm{c}_\alpha\mathrm{d}\beta & -\mathrm{s}_\theta\mathrm{c}_\alpha\mathrm{d}\beta & 0 \end{pmatrix} \begin{pmatrix} a\mathrm{c}_\theta \\ a\mathrm{s}_\theta \\ d \end{pmatrix} = \begin{pmatrix} +a\mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{d}\beta - d\mathrm{c}_\theta\mathrm{c}_\alpha\mathrm{d}\beta \\ -a\mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{d}\beta - d\mathrm{s}_\theta\mathrm{c}_\alpha\mathrm{d}\beta \\ a\mathrm{c}_\alpha\mathrm{d}\beta \end{pmatrix}$$

that give rise respectively to

$$\boldsymbol{\delta}_\beta = \begin{pmatrix} -\mathrm{s}_\theta\mathrm{c}_\alpha \\ +\mathrm{c}_\theta\mathrm{c}_\alpha \\ \mathrm{s}_\alpha \end{pmatrix}, \quad \boldsymbol{d}_\beta = \begin{pmatrix} +a\mathrm{s}_\theta\mathrm{s}_\alpha - d\mathrm{c}_\theta\mathrm{c}_\alpha \\ -a\mathrm{c}_\theta\mathrm{s}_\alpha - d\mathrm{s}_\theta\mathrm{c}_\alpha \\ a\mathrm{c}_\alpha \end{pmatrix}$$

using eq. (6.34). Embedding them in the differential matrix

$$\boldsymbol{D}_\beta = \begin{pmatrix} 0 & -\mathrm{s}_\alpha & +\mathrm{c}_\theta\mathrm{c}_\alpha & +a\mathrm{s}_\theta\mathrm{s}_\alpha - d\mathrm{c}_\theta\mathrm{c}_\alpha \\ +\mathrm{s}_\alpha & 0 & +\mathrm{s}_\theta\mathrm{c}_\alpha & -a\mathrm{c}_\theta\mathrm{s}_\alpha - d\mathrm{s}_\theta\mathrm{c}_\alpha \\ -\mathrm{c}_\theta\mathrm{c}_\alpha & -\mathrm{s}_\theta\mathrm{c}_\alpha & 0 & a\mathrm{c}_\alpha \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.42}$$

the partial derivative with respect to β can be easily computed as

$$\frac{\partial \boldsymbol{T}}{\partial \beta} = \boldsymbol{D}_\beta \boldsymbol{T} = \begin{pmatrix} -\mathrm{c}_\theta\mathrm{s}_\beta - \mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{c}_\beta & 0 & \mathrm{c}_\theta\mathrm{c}_\beta - \mathrm{s}_\theta\mathrm{s}_\alpha\mathrm{s}_\beta & 0 \\ -\mathrm{s}_\theta\mathrm{s}_\beta + \mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{c}_\beta & 0 & \mathrm{s}_\theta\mathrm{c}_\beta + \mathrm{c}_\theta\mathrm{s}_\alpha\mathrm{s}_\beta & 0 \\ -\mathrm{c}_\alpha\mathrm{c}_\beta & 0 & -\mathrm{c}_\alpha\mathrm{s}_\beta & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

thanks to eq. (6.31).

Full Parametric Differential Model

The partial derivatives of the homogeneous transformation matrix with respect to the kinematic parameters give the total differential

$$\mathrm{d}\boldsymbol{T} = \frac{\partial \boldsymbol{T}}{\partial d}\mathrm{d}d + \frac{\partial \boldsymbol{T}}{\partial \theta}\mathrm{d}\theta + \frac{\partial \boldsymbol{T}}{\partial a}\mathrm{d}a + \frac{\partial \boldsymbol{T}}{\partial \alpha}\mathrm{d}\alpha + \frac{\partial \boldsymbol{T}}{\partial \beta}\mathrm{d}\beta \tag{6.43}$$

therefore they measure the sensitivity of the reference frame transformation to the variation of the corresponding parameters. If these parametric variations are due the variation

of another generic quantity ζ, e.g., the temperature of a link, the total differential can be converted into the total derivative of the homogeneous transformation matrix with respect to that quantity

$$\frac{\mathrm{d}\boldsymbol{T}}{\mathrm{d}\zeta} = \frac{\partial \boldsymbol{T}}{\partial d}\frac{\partial d}{\partial \zeta} + \frac{\partial \boldsymbol{T}}{\partial \theta}\frac{\partial \theta}{\partial \zeta} + \frac{\partial \boldsymbol{T}}{\partial a}\frac{\partial a}{\partial \zeta} + \frac{\partial \boldsymbol{T}}{\partial \alpha}\frac{\partial \alpha}{\partial \zeta} + \frac{\partial \boldsymbol{T}}{\partial \beta}\frac{\partial \beta}{\partial \zeta}$$

measuring the sensitivity of the reference frame transformation to the variation of the target quantity.

If all derivatives in eq. (6.43) are represented using the corresponding differential matrices, according to eq. (6.31)

$$\begin{aligned}\mathrm{d}\boldsymbol{T} &= \boldsymbol{D}_d\boldsymbol{T}\mathrm{d}d + \boldsymbol{D}_\theta\boldsymbol{T}\mathrm{d}\theta + \boldsymbol{D}_a\boldsymbol{T}\mathrm{d}a + \boldsymbol{D}_\alpha\boldsymbol{T}\mathrm{d}\alpha + \boldsymbol{D}_\beta\boldsymbol{T}\mathrm{d}\beta = \\ &= \left(\boldsymbol{D}_d\mathrm{d}d + \boldsymbol{D}_\theta\mathrm{d}\theta + \boldsymbol{D}_a\mathrm{d}a + \boldsymbol{D}_\alpha\mathrm{d}\alpha + \boldsymbol{D}_\beta\mathrm{d}\beta\right)\boldsymbol{T} = (\boldsymbol{E}_d + \boldsymbol{E}_\theta + \boldsymbol{E}_a + \boldsymbol{E}_\alpha + \boldsymbol{E}_\beta)\boldsymbol{T}\end{aligned}$$

the homogeneous transformation matrix $\boldsymbol{T}$ can be factored out as common term. Hence, by embedding all differential terms into a cumulative differential error matrix

$$\boldsymbol{E} \coloneqq \boldsymbol{E}_d + \boldsymbol{E}_\theta + \boldsymbol{E}_a + \boldsymbol{E}_\alpha + \boldsymbol{E}_\beta = \boldsymbol{D}_d\mathrm{d}d + \boldsymbol{D}_\theta\mathrm{d}\theta + \boldsymbol{D}_a\mathrm{d}a + \boldsymbol{D}_\alpha\mathrm{d}\alpha + \boldsymbol{D}_\beta\mathrm{d}\beta \tag{6.44}$$

the differential in eq. (6.43) can be implicitly written as

$$\mathrm{d}\boldsymbol{T} = \boldsymbol{E}\boldsymbol{T} \tag{6.45}$$

with the same fashion of eq. (6.33).

6.2.3 Entire Manipulator Error Model

As widely clarified in subsection 2.3.4, the kinematic model of a manipulator *essentially* consists in a description of the kinematic relation between the base reference frame and the end effector reference frame. In the case of 6DOF manipulators, the homogeneous transformation matrix $\boldsymbol{T}_6^0$, describing the total kinematic transformation between the base frame $\mathcal{R}_0$ and the end frame $\mathcal{R}_6$, is simply given by the product of each homogeneous transformation matrix $\boldsymbol{T}_i^{i-1}$, with $i = 1, \ldots, 6$, describing the partial kinematic transformation between the $(i-1)$-th link frame and the i-th link frame, function of the i-th kinematic parameters *only*.

By virtue of such a representation of the manipulator kinematic chain, the differential kinematic error model developed before characterises *only* the dependence of the differential error of the relative kinematic relation between two adjacent manipulator link frames from the differential error of the kinematic parameters involved in that relation. In order to formulate a *complete* differential kinematic error model of the entire manipulator, it is thus necessary to assess the differential error for each pair of adjacent links frames and properly transpose it into a *common* absolute reference frame, which may be, in the most straightforward case, the base frame.

Single Joint Differential Error Matrix in Base Frame

The first step consists in representing the differential change of a relative frame transformation in base frame. The i-th kinematic error has to be propagated along the entire kinematic chain, down to frame 0: since

$$\boldsymbol{T}_6^0 + \mathrm{d}\boldsymbol{T}_6^0 = \prod_{j=1}^{i-1} \boldsymbol{T}_j^{j-1} \left(\boldsymbol{T}_i^{i-1} + \mathrm{d}\boldsymbol{T}_i^{i-1}\right) \prod_{j=i+1}^{6} \boldsymbol{T}_j^{j-1} = \boldsymbol{T}_{i-1}^0 \left(\boldsymbol{T}_i^{i-1} + \mathrm{d}\boldsymbol{T}_i^{i-1}\right) \boldsymbol{T}_6^i =$$
$$= \boldsymbol{T}_{i-1}^0 \boldsymbol{T}_i^{i-1} \boldsymbol{T}_6^i + \boldsymbol{T}_{i-1}^0 \mathrm{d}\boldsymbol{T}_i^{i-1} \boldsymbol{T}_6^i = \boldsymbol{T}_6^0 + \boldsymbol{T}_{i-1}^0 \mathrm{d}\boldsymbol{T}_i^{i-1} \boldsymbol{T}_6^i$$

then

$$\mathrm{d}\boldsymbol{T}_6^0 = \boldsymbol{T}_{i-1}^0 \mathrm{d}\boldsymbol{T}_i^{i-1} \boldsymbol{T}_6^i \tag{6.46}$$

gives the differential change of $\boldsymbol{T}_6^0$, due to a differential change of $\boldsymbol{T}_i^{i-1}$. On the other hand, these differential changes can be respectively written as

$$\mathrm{d}\boldsymbol{T}_6^0 = {}^0\boldsymbol{E}\boldsymbol{T}_6^0 \tag{6.47}$$

and

$$\mathrm{d}\boldsymbol{T}_i^{i-1} = {}^{i-1}\boldsymbol{E}\boldsymbol{T}_i^{i-1} \tag{6.48}$$

adding the proper labels to eq. (6.32). Substituting them into eq. (6.46) gives

$${}^0\boldsymbol{E}\boldsymbol{T}_6^0 = \boldsymbol{T}_{i-1}^0 {}^{\mathrm{i}-1}\boldsymbol{E}\boldsymbol{T}_i^{i-1} \boldsymbol{T}_6^i = \boldsymbol{T}_{i-1}^0 {}^{\mathrm{i}-1}\boldsymbol{E}\boldsymbol{T}_6^{i-1}$$

hence the differential error matrix is transformed into base frame through

$${}^0\boldsymbol{E} = \boldsymbol{T}_{i-1}^0 {}^{\mathrm{i}-1}\boldsymbol{E}\boldsymbol{T}_0^{i-1} \tag{6.49}$$

which is a special case of eq. (6.26) applied to the manipulator kinematic chain.

All Joints Differential Error Matrices in Base Frame

The second step consists in evaluating the differential change of all relative frame transformations. All kinematic errors have to be propagated along the entire kinematic chain, down to frame 0: since

$$\boldsymbol{T}_6^0 + \mathrm{d}\boldsymbol{T}_6^0 = \prod_{i=1}^{6} \left(\boldsymbol{T}_i^{i-1} + \mathrm{d}\boldsymbol{T}_i^{i-1}\right) = \prod_{i=1}^{6} \boldsymbol{T}_i^{i-1} + \sum_{i=1}^{6} \left(\prod_{j=1}^{i-1} \boldsymbol{T}_j^{j-1} \right) \mathrm{d}\boldsymbol{T}_i^{i-1} \left(\prod_{j=i}^{6} \boldsymbol{T}_j^{j-1} \right) =$$
$$= \boldsymbol{T}_6^0 + \sum_{i=1}^{6} \boldsymbol{T}_{i-1}^0 \mathrm{d}\boldsymbol{T}_i^{i-1} \boldsymbol{T}_6^i$$

neglecting all higher order terms, then

$$\mathrm{d}\boldsymbol{T}_6^0 = \sum_{i=1}^{6} \boldsymbol{T}_{i-1}^0 \mathrm{d}\boldsymbol{T}_i^{i-1} \boldsymbol{T}_6^i \tag{6.50}$$

gives the differential change of $\boldsymbol{T}_6^0$, due to the differential change of all $\boldsymbol{T}_i^{i-1}$, which is just the summation of all terms defined in eq. (6.46). In the case of multiple errors, eqs. (6.48) and (6.49) must be respectively rewritten as

$$\mathrm{d}\boldsymbol{T}_i^{i-1} = {}^{i-1}\boldsymbol{E}_i \boldsymbol{T}_i^{i-1} \tag{6.51}$$

and

$${}^0\boldsymbol{E}_i = \boldsymbol{T}_{i-1}^0 {}^{i-1}\boldsymbol{E}_i \boldsymbol{T}_0^{i-1} \tag{6.52}$$

where the right subscript denotes the source of perturbation, whereas the left superscript denotes the frame of representation. Substituting them into eq. (6.50) gives

$${}^0\boldsymbol{E}\boldsymbol{T}_6^0 = \sum_{i=1}^{6} \boldsymbol{T}_{i-1}^0 {}^{i-1}\boldsymbol{E}_i \boldsymbol{T}_i^{i-1} \boldsymbol{T}_6^i = \sum_{i=1}^{6} \boldsymbol{T}_{i-1}^0 {}^{i-1}\boldsymbol{E}_i \boldsymbol{T}_6^{i-1} = \sum_{i=1}^{6} {}^0\boldsymbol{E}_i \boldsymbol{T}_6^0$$

hence the differential error matrix in base frame, due to the differential error of all parameters, is given by

$${}^0\boldsymbol{E} = \sum_{i=1}^{6} {}^0\boldsymbol{E}_i = \sum_{i=1}^{6} \boldsymbol{T}_{i-1}^0 {}^{i-1}\boldsymbol{E}_i \boldsymbol{T}_0^{i-1} \tag{6.53}$$

which is just the summation of all differential error matrices defined in eq. (6.52).

Differential Error Vectors in Base Frame

The total differential error matrix in global frame is expanded as

$${}^0\boldsymbol{E} = \begin{pmatrix} \boldsymbol{S}\left({}^0\boldsymbol{\varepsilon}\right) & {}^0\boldsymbol{e} \\ \boldsymbol{0}^\mathsf{T} & 0 \end{pmatrix} \tag{6.54}$$

and each partial differential error matrix in local frame is expanded as

$${}^{i-1}\boldsymbol{E}_i = \begin{pmatrix} \boldsymbol{S}\left({}^{i-1}\boldsymbol{\varepsilon}_i\right) & {}^{i-1}\boldsymbol{e}_i \\ \boldsymbol{0}^\mathsf{T} & 0 \end{pmatrix}$$

using the same standard block structure based on differential error vectors. Using them to perform the products in eq. (6.53) blockwise, and applying eq. (6.27), it is possible to relate the orientation total differential error vector

$${}^0\boldsymbol{\varepsilon} = \sum_{i=1}^{6} {}^0\boldsymbol{\varepsilon}_i = \sum_{i=1}^{6} \boldsymbol{R}_{i-1}^0 {}^{i-1}\boldsymbol{\varepsilon}_i \tag{6.55}$$

to the orientation partial differential error vectors *only*, and the position total differential error vector

$${}^0\boldsymbol{e} = \sum_{i=1}^{6} {}^0\boldsymbol{e}_i = \sum_{i=1}^{6} \left(\boldsymbol{R}_{i-1}^0 {}^{i-1}\boldsymbol{e}_i - \boldsymbol{S}\left({}^{i-1}\boldsymbol{\varepsilon}_i\right) \boldsymbol{t}_{i-1}^0 \right) \tag{6.56}$$

to *both* the position and the orientation partial differential error vectors.

End Effector Pose Differential Error

As said repeatedly, the goal of manipulator kinematics is the description of the pose of the end effector with respect to the base. Whenever the reference frame 6 is attached to the end effector and the reference frame 0 is attached to the base, the homogeneous transformation matrix

$$\boldsymbol{T}_6^0 = \begin{pmatrix} \boldsymbol{R}_6^0 & \boldsymbol{t}_6^0 \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}$$

describes the end effector pose, as the translation vector $\boldsymbol{t}_6^0$ represents the end effector position, while the rotation matrix $\boldsymbol{R}_6^0$ represents the end effector orientation. For this reason, the homogeneous transformation matrix $\boldsymbol{T}_6^0$ is often considered as the kinematic model itself and its differential change

$$\mathrm{d}\boldsymbol{T}_6^0 = \begin{pmatrix} \mathrm{d}\boldsymbol{R}_6^0 & \mathrm{d}\boldsymbol{t}_6^0 \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}$$

compactly describes the manipulator differential kinematic error. The error matrix in eq. (6.54) yields the differential end effector orientation error

$$\mathrm{d}\boldsymbol{R}_6^0 = \boldsymbol{S}\big({}^0\boldsymbol{\varepsilon}\big)\, \boldsymbol{R}_6^0$$

and the differential end effector position error

$$\mathrm{d}\boldsymbol{t}_6^0 = \boldsymbol{S}\big({}^0\boldsymbol{\varepsilon}\big)\, \boldsymbol{t}_6^0 + {}^0\boldsymbol{e}$$

applying respectively eqs. (6.35) and (6.36).

On the other hand, when the end effector equips a specific tool, the goal of manipulator kinematics is the description of the tool pose in the work-space. Since, typically, the tool is firmly mounted to the manipulator flange by means of suitable mechanical couplings, in principle, there is no relative motion of the tool with respect to the flange, then the tool pose is related to the flange pose by means of constant rigid transformation. Besides, the tool orientation is usually the same as the flange orientation, or, sometimes, it differs by an elemental rotation only, thus difference concerns only the tool position, which, by the way, is not just an offset flange position, as it depends on the flange orientation *too*. If $\boldsymbol{r}_\mathrm{T}^6$ is the position of the tool in the flange reference frame, then

$$\boldsymbol{r}_\mathrm{T}^0 = \boldsymbol{R}_6^0 \boldsymbol{r}_\mathrm{T}^6 + \boldsymbol{t}_6^0$$

gives the position of the tool in the base reference frame; the same relation may be also written succinctly as

$$\boldsymbol{h}_\mathrm{T}^0 = \boldsymbol{T}_6^0 \boldsymbol{h}_\mathrm{T}^6$$

using homogeneous coordinates representation. The differential variation of the tool homogeneous position in base frame is given by

$$\mathrm{d}\boldsymbol{h}_\mathrm{T}^0 = \mathrm{d}\big(\boldsymbol{T}_6^0 \boldsymbol{h}_\mathrm{T}^6\big) = \mathrm{d}\boldsymbol{T}_6^0 \boldsymbol{h}_\mathrm{T}^6 = {}^0\boldsymbol{E}\boldsymbol{T}_6^0 \boldsymbol{h}_\mathrm{T}^6 = {}^0\boldsymbol{E}\boldsymbol{h}_\mathrm{T}^0 \tag{6.57}$$

since the tool position in flange frame is fixed; the differential tool position error is eventually computed as

$$\mathrm{d}\boldsymbol{r}_{\mathrm{T}}^{0} = \boldsymbol{S}\left({}^{0}\boldsymbol{\varepsilon}\right)\boldsymbol{r}_{\mathrm{T}}^{0} + {}^{0}\boldsymbol{e} \tag{6.58}$$

using again eq. (6.54) into eq. (6.36).

Kinematic Function Jacobian

Adding the proper labels to eq. (6.44)

$${}^{i-1}\boldsymbol{E}_{i} = {}^{i-1}\boldsymbol{D}_{d_i}\mathrm{d}d_i + {}^{i-1}\boldsymbol{D}_{\theta_i}\mathrm{d}\theta_i + {}^{i-1}\boldsymbol{D}_{a_i}\mathrm{d}a_i + {}^{i-1}\boldsymbol{D}_{\alpha_i}\mathrm{d}\alpha_i + {}^{i-1}\boldsymbol{D}_{\beta_i}\mathrm{d}\beta_i$$

it is possible to expand eq. (6.52) as

$$\begin{aligned}{}^{0}\boldsymbol{E}_{i} &= \boldsymbol{T}_{i-1}^{0}\left({}^{i-1}\boldsymbol{D}_{d_i}\mathrm{d}d_i + {}^{i-1}\boldsymbol{D}_{\theta_i}\mathrm{d}\theta_i + {}^{i-1}\boldsymbol{D}_{a_i}\mathrm{d}a_i + {}^{i-1}\boldsymbol{D}_{\alpha_i}\mathrm{d}\alpha_i + {}^{i-1}\boldsymbol{D}_{\beta_i}\mathrm{d}\beta_i\right)\boldsymbol{T}_{0}^{i-1} = \\ &= {}^{0}\boldsymbol{D}_{d_i}\mathrm{d}d_i + {}^{0}\boldsymbol{D}_{\theta_i}\mathrm{d}\theta_i + {}^{0}\boldsymbol{D}_{a_i}\mathrm{d}a_i + {}^{0}\boldsymbol{D}_{\alpha_i}\mathrm{d}\alpha_i + {}^{0}\boldsymbol{D}_{\beta_i}\mathrm{d}\beta_i\end{aligned}$$

using only differential matrices. Substituting this indexed term into eq. (6.47) gives

$${}^{0}\boldsymbol{E} = \sum_{i=1}^{6}\left({}^{0}\boldsymbol{D}_{d_i}\mathrm{d}d_i + {}^{0}\boldsymbol{D}_{\theta_i}\mathrm{d}\theta_i + {}^{0}\boldsymbol{D}_{a_i}\mathrm{d}a_i + {}^{0}\boldsymbol{D}_{\alpha_i}\mathrm{d}\alpha_i + {}^{0}\boldsymbol{D}_{\beta_i}\mathrm{d}\beta_i\right)$$

that allows to compute the total differential of the tool homogeneous coordinates as

$$\mathrm{d}\boldsymbol{h}_{\mathrm{T}}^{0} = \sum_{i=1}^{6}\left({}^{0}\boldsymbol{D}_{d_i}\mathrm{d}d_i + {}^{0}\boldsymbol{D}_{\theta_i}\mathrm{d}\theta_i + {}^{0}\boldsymbol{D}_{a_i}\mathrm{d}a_i + {}^{0}\boldsymbol{D}_{\alpha_i}\mathrm{d}\alpha_i + {}^{0}\boldsymbol{D}_{\beta_i}\mathrm{d}\beta_i\right)\boldsymbol{h}_{\mathrm{T}}^{0} \tag{6.59}$$

applying eq. (6.57). If all kinematic parameters are gathered into a large vector

$$\boldsymbol{k} \coloneqq \begin{pmatrix}\boldsymbol{k}_1 \\ \vdots \\ \boldsymbol{k}_6\end{pmatrix} \in \mathbb{R}^{30}$$

with

$$\boldsymbol{k}_i \coloneqq \begin{pmatrix}d_i \\ \theta_i \\ a_i \\ \alpha_i \\ \beta_i\end{pmatrix}$$

the above total differential might be compactly written as

$$\mathrm{d}\boldsymbol{h}_{\mathrm{T}}^{0} = \boldsymbol{J}_{\boldsymbol{k}}\boldsymbol{h}_{\mathrm{T}}^{0}\mathrm{d}\boldsymbol{k}$$

using the Jacobian matrix

$$\boldsymbol{J}_{\boldsymbol{k}}\boldsymbol{h}_{\mathrm{T}}^{0} = \begin{pmatrix}\boldsymbol{J}_{\boldsymbol{k}_1}\boldsymbol{h}_{\mathrm{T}}^{0} & \cdots & \boldsymbol{J}_{\boldsymbol{k}_6}\boldsymbol{h}_{\mathrm{T}}^{0}\end{pmatrix}$$

where each

$$\boldsymbol{J}_{\boldsymbol{k}_i}\boldsymbol{h}_{\mathrm{T}}^{0} \coloneqq \begin{pmatrix} {}^{0}\boldsymbol{D}_{d_i}\boldsymbol{h}_{\mathrm{T}}^{0} & {}^{0}\boldsymbol{D}_{\theta_i}\boldsymbol{h}_{\mathrm{T}}^{0} & {}^{0}\boldsymbol{D}_{a_i}\boldsymbol{h}_{\mathrm{T}}^{0} & {}^{0}\boldsymbol{D}_{\alpha_i}\boldsymbol{h}_{\mathrm{T}}^{0} & {}^{0}\boldsymbol{D}_{\beta_i}\boldsymbol{h}_{\mathrm{T}}^{0} \end{pmatrix} \tag{6.60}$$

is the Jacobian matrix of the tool homogeneous coordinates in base frame with respect to the i-th kinematic parameters. The corresponding i-th Jacobian matrix of the tool position in base frame

$$\boldsymbol{J}_{\boldsymbol{k}_i}\boldsymbol{r}_{\mathrm{T}}^{0} = \begin{pmatrix} \dfrac{\partial \boldsymbol{r}_{\mathrm{T}}^{0}}{\partial d_i} & \dfrac{\partial \boldsymbol{r}_{\mathrm{T}}^{0}}{\partial \theta_i} & \dfrac{\partial \boldsymbol{r}_{\mathrm{T}}^{0}}{\partial a_i} & \dfrac{\partial \boldsymbol{r}_{\mathrm{T}}^{0}}{\partial \alpha_i} & \dfrac{\partial \boldsymbol{r}_{\mathrm{T}}^{0}}{\partial \beta_i} \end{pmatrix} =$$

$$= \begin{pmatrix} \boldsymbol{S}\big({}^{0}\boldsymbol{\delta}_{d_i}\big)\boldsymbol{r}_{\mathrm{T}}^{0}+{}^{0}\boldsymbol{d}_{d_i} & \boldsymbol{S}\big({}^{0}\boldsymbol{\delta}_{\theta_i}\big)\boldsymbol{r}_{\mathrm{T}}^{0}+{}^{0}\boldsymbol{d}_{\theta_i} & \boldsymbol{S}\big({}^{0}\boldsymbol{\delta}_{a_i}\big)\boldsymbol{r}_{\mathrm{T}}^{0}+{}^{0}\boldsymbol{d}_{a_i} & \boldsymbol{S}\big({}^{0}\boldsymbol{\delta}_{\alpha_i}\big)\boldsymbol{r}_{\mathrm{T}}^{0}+{}^{0}\boldsymbol{d}_{\alpha_i} & \boldsymbol{S}\big({}^{0}\boldsymbol{\delta}_{\beta_i}\big)\boldsymbol{r}_{\mathrm{T}}^{0}+{}^{0}\boldsymbol{d}_{\beta_i} \end{pmatrix}$$

easily comes out by exploiting the peculiar structure of differential matrices.

As mentioned in subsection 6.2.1, the kinematic transformation chosen for deriving the differential error model is generic enough to suit both kinematic conventions required by the kinematic model especially designed for calibration purposes. For this reason, the Jacobian matrix required by the numerical identification procedure may be easily extracted from the full Jacobian matrix above computed, by deleting all the columns corresponding to the kinematic parameters not involved in that kinematic model.

Chapter 7

Conclusion

7.1 Objectives Fulfilment

The prime goal of this study was the investigation and the compensation of the geometric and thermal errors in the kinematic model of a 6DOF articulated robot able to affect its volumetric accuracy. In light of the achieved experimental outcomes and results, it is rather reasonable to claim that the objectives set have been satisfactorily fulfilled. Nevertheless, the obtained achievements have not come without some difficulty and obstacle, some of which are here briefly reported.

Special attention has been devoted to planning and developing the numerical tools on MATLAB®. In this regard, a blind approach involving a generic and redundant kinematic model was abandoned in favour of a more approach solution involving a custom revised kinematic model in order to avoid heavy numeric processing and employ only very simple computational tools such as matrix product and inversion; for this reason, the proposed solution required high research effort at design time but very low computational effort at run time. In the end, the algorithmic software procedure implementing the iterative solution of the numerical optimisation problem proved extremely fast and stable. Such an achievement has a two fold advantage in practical terms, for it gives the possibility to integrate effectively the implemented procedure within even bigger complex projects without introducing a performance bottleneck and to port entirely the implemented procedure into a different target software framework, embedding very basic matrices libraries, without any special precaution or difficulty by the programmers.

Great care has been taken also in the mechanical aspects regarding the design of the robot tool employed for calibration purposes. Since only the geometric characteristics of the robot sheer kinematic chain can be observed and then trimmed after numerical identification, the spatial relationship between the robot hand and the tool targets comes into play as a constant and certain rigid transformation. For this reason, designing a reliable tailored mechanical interface between the tool plate and the robot flange represented immediately an issue of paramount importance, in order to allow placement and replacement avoiding the introduction of undetectable sources of error.

Another important issue has been definitely the cost and the ease of the temperature

measurement chain to employ as real-time robot temperatures monitor. The original project, involving generic data acquisition systems and standard analog thermal probes requiring compensation, extremely expensive and cumbersome, was deemed infeasible and thus subsequently set aside. A long inquiry was then necessary in order to pinpoint all elements that make up the modular, convenient and low-cost measurement solution eventually adopted, namely the compact transducers endowed with built in digital-to-analog converter and serial interface and the cheap microcontroller board equipped with serial interface and wireless module.

The working experience connected to the research activity has also represented an opportunity to handle a large and complex all-round project from start to finish and a pretext to acquire priceless scientific knowledge and valuable technological expertise. The study and the work carried out while addressing a matter not yet sufficiently investigated in the current landscape of research advancements, might open the way for several stimulating future works.

7.2 Further Developments

Despite the quite promising results, there still remains room for further improvement.

The first interesting idea could be to set up a temperature monitor and perform a thermal calibration of another industrial robot constructed in a completely different way. For instance, robots made of aluminium could exhibit larger thermal strain, because aluminium features higher thermal expansion and higher thermal conductivity than standard cast iron. Moreover, robots constructed with a full outer shell that envelops the mechanical framework and covers all joint motors, could experience larger thermal strain too, for the heat produced in the joints is trapped within the robot body and cannot be dissipated directly by the air flow.

It might be useful to identify and single out the most relevant spots in the robot links that give an approximate picture of the robot thermal gradient, in order to either keep only the necessary thermal transducers or to select the minimum number of thermal transducers in view of performing the thermal calibration of another articulated robot unit possibly different from the one already studied.

Investigating the thermal error due to self heating of single joints, that is to say performing repetitive simple joint motions and checking periodically the induced tool position error, could represent a valid though very time consuming alternative strategy for probing the effect of temperature variations, due to local sources separately, on the geometry of the manipulator kinematic chain.

Finally, the implementation stage of robot calibration, in which the results are routinely applied to correct the robot direct kinematic function, may be considered perfect whenever the results of calibration are exploited to refine the robot inverse kinematic function too. In this regard, for instance, it might be reasonable to adopt an iterative optimisation scheme similar to the one designed for identification of the kinematic parameters, after the due modifications.

Appendix A

Linear Algebra

A.1 Vectors

A.1.1 Fundamentals

Real **vectors** are arrays of real numbers: a vector $\boldsymbol{x}$ of dimension n is an ordered sequence of n scalars

$$\boldsymbol{x} = (x_i)_{1 \leq i \leq n} = (x_1, \ldots, x_n) \in \mathbb{R}^n$$

that is an element of the n-dimensional Euclidean space. The entries of a vector $\boldsymbol{x} \in \mathbb{R}^n$ are actually its components with respect to the **standard basis** $E = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n)$ of $\mathbb{R}^n$

$$\boldsymbol{x} = \sum_{i=1}^{n} x_i \boldsymbol{e}_i$$

briefly[1] defined as $\boldsymbol{e}_i = \left(\delta_{ij}\right)_{1 \leq j \leq n}$, $1 \leq i \leq n$.

A set of p vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p \in \mathbb{R}^n$, with $p \leq n$, is said to be **linearly independent** if a linear combination of the vectors is zero only when all coefficients are zero

$$\lambda_1 \boldsymbol{x}_1 + \ldots + \lambda_p \boldsymbol{x}_p = \boldsymbol{0} \iff \lambda_1, \ldots, \lambda_p = 0$$

or, equivalently, none of them may be written as a linear combination of the others. The set is said to be **linearly dependent**, otherwise.

Dot Product

The scalar defined by

$$\boldsymbol{x} \cdot \boldsymbol{y} \coloneqq x_1 y_1 + \ldots + x_n y_n = \sum_{i=1}^{n} x_i y_i.$$

is called **scalar product** or **dot product** between $\boldsymbol{x}$ and $\boldsymbol{y}$. It is commutative, as verifiable from the definition. Two vectors giving zero dot product are said **orthogonal**.

[1] The symbol $\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$ denotes the **Kronecker Delta**.

Euclidean Norm

The dot product *naturally* induces the **norm**

$$\|\boldsymbol{x}\| \coloneqq \sqrt{\boldsymbol{x} \cdot \boldsymbol{x}} = \sqrt{\sum_{i=1}^{n} x_i^2}$$

termed **euclidean norm**, or **euclidean distance**, as it yields a measure of the length, or magnitude, of the vector $\boldsymbol{x}$. As any norm, it satisfies

$$\|\boldsymbol{x}\| \geq 0,\ \forall \boldsymbol{x} \in \mathbb{R}^n,\ \|\boldsymbol{x}\| = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{0}$$

called positivity property, and

$$\|\lambda \boldsymbol{x}\| = |\lambda| \, \|\boldsymbol{x}\|,\ \forall \boldsymbol{x} \in \mathbb{R}^n, \forall \lambda \in \mathbb{R}$$

called homogeneity property.

Vectors with unit norm are termed **unit vectors** and typically represent spatial directions, that is to say vectors with unimportant length. Generic vectors can be transformed into *corresponding* unit vectors by means of normalisation: if $\boldsymbol{x} \neq \boldsymbol{0}$, then $\hat{\boldsymbol{x}} = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|}$ is a unit vector.

Projections

Any vector can be decomposed into the sum of two vectors, one parallel and one perpendicular to another vector. The **orthogonal projection** of $\boldsymbol{y}$ onto $\boldsymbol{x} \neq 0$ gives

$$\boldsymbol{y}_{\parallel} \coloneqq \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\|^2} \boldsymbol{x} \tag{A.1}$$

called **axial component** of $\boldsymbol{y}$ with respect to $\boldsymbol{x}$, while the **orthogonal projection** of $\boldsymbol{y}$ onto a plane orthogonal to $\boldsymbol{x}$ gives

$$\boldsymbol{y}_{\perp} \coloneqq \boldsymbol{y} - \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\|^2} \boldsymbol{x} \tag{A.2}$$

called **normal component** of $\boldsymbol{y}$ with respect to $\boldsymbol{x}$.

Norm Inequalities

If $\boldsymbol{x} = \boldsymbol{0}$, then $\boldsymbol{x} \cdot \boldsymbol{y} = 0$, whereas, if $\boldsymbol{x} \neq \boldsymbol{0}$, the squared norm of eq. (A.2)

$$0 \leq \left\| \boldsymbol{y} - \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\|^2} \boldsymbol{x} \right\|^2 = \|\boldsymbol{y}\|^2 - 2\frac{(\boldsymbol{x} \cdot \boldsymbol{y})^2}{\|\boldsymbol{x}\|^2} + \frac{(\boldsymbol{x} \cdot \boldsymbol{y})^2}{\|\boldsymbol{x}\|^2} = \|\boldsymbol{y}\|^2 - \frac{(\boldsymbol{x} \cdot \boldsymbol{y})^2}{\|\boldsymbol{x}\|^2}$$

leads to the property

$$\|\boldsymbol{x}\| \, \|\boldsymbol{y}\| \leq |\boldsymbol{x} \cdot \boldsymbol{y}|,\ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n \tag{A.3}$$

known as **Cauchy-Schwarz inequality**. Since

$$\boldsymbol{y} - \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{\|\boldsymbol{x}\|^2}\boldsymbol{x} = \boldsymbol{0} \iff \boldsymbol{y} = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{\|\boldsymbol{x}\|^2}\boldsymbol{x}$$

and, conversely, $|\boldsymbol{x}\cdot\lambda\boldsymbol{x}| = |\lambda|\,\boldsymbol{x}\cdot\boldsymbol{x} = |\lambda|\,\|\boldsymbol{x}\|^2 = \|\lambda\boldsymbol{x}\|\,\|\boldsymbol{x}\|$, then

$$\|\boldsymbol{x}\|\,\|\boldsymbol{y}\| = |\boldsymbol{x}\cdot\boldsymbol{y}|\,, \boldsymbol{x}\neq\boldsymbol{0} \iff \exists\lambda\in\mathbb{R}\colon \boldsymbol{y}=\lambda\boldsymbol{x}, \tag{A.4}$$

that is to say eq. (A.3) becomes an equality only if the vectors are linearly dependent. The inequality also justifies the alternative expression of the dot product

$$\boldsymbol{x}\cdot\boldsymbol{y} = \|\boldsymbol{x}\|\,\|\boldsymbol{y}\|\cos\theta \tag{A.5}$$

where $\cos\theta$ is termed **direction cosine** and $\theta\in[0,\pi]$ is the angle between $\boldsymbol{x}$ and $\boldsymbol{y}$.

Expanding, through eq. (A.3), the squared norm of the vector sum between $\boldsymbol{x}$ and $\boldsymbol{y}$

$$\|\boldsymbol{x}+\boldsymbol{y}\|^2 = \|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2 + 2\boldsymbol{x}\cdot\boldsymbol{y} \leq \|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2 + 2\,\|\boldsymbol{x}\|\,\|\boldsymbol{y}\| = (\|\boldsymbol{x}\| + \|\boldsymbol{y}\|)^2$$

leads to the additional property

$$\|\boldsymbol{x}+\boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|\,,\ \forall\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^n \tag{A.6}$$

known as **triangle inequality**.

A.1.2 3D Vectors

Spatial geometry studies mainly **3D vectors**, i.e. vectors of the 3-dimensional Euclidean space $\mathbb{R}^3$. The vectors of the **standard basis** of $\mathbb{R}^3$ are usually denoted as $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$.

Cross Product

The vector defined by

$$\boldsymbol{x}\times\boldsymbol{y} = (x_2y_3 - x_3y_2, x_3y_1 - x_1y_3, x_1y_2 - x_2y_1)$$

is called **vector product** or **cross product** between $\boldsymbol{x}$ and $\boldsymbol{y}$. It is linear

$$\boldsymbol{x}\times(\lambda\boldsymbol{y}+\mu\boldsymbol{z}) = \lambda\,(\boldsymbol{x}\times\boldsymbol{y}) + \mu\,(\boldsymbol{x}\times\boldsymbol{z})\,,\ \forall\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\in\mathbb{R}^3 \tag{A.7}$$

but not commutative, in fact it is anti-commutative

$$\boldsymbol{y}\times\boldsymbol{x} = -\boldsymbol{x}\times\boldsymbol{y},\ \forall\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^3 \tag{A.8}$$

as verified from the definition. The cross product of two parallel vectors is zero

$$\lambda\boldsymbol{x}\times\boldsymbol{x} = \boldsymbol{0},\ \forall\boldsymbol{x}\in\mathbb{R}^3, \forall\lambda\in\mathbb{R} \tag{A.9}$$

because of the above properties.

Mixed Product

The dot product of the cross product between two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ with another vector $\boldsymbol{z}$ is called **scalar triple product** or **mixed product**. Since

$$\begin{aligned}(\boldsymbol{x}\times\boldsymbol{y})\cdot\boldsymbol{z} &= z_1x_2y_3 - z_1x_3y_2 + z_2x_3y_1 - z_2x_1y_3 + z_3x_1y_2 - z_3x_2y_1 = \\ &= x_1\left(y_2z_3 - y_3z_2\right) + x_2\left(y_3z_1 - y_1z_3\right) + x_3\left(y_1z_2 - y_2z_1\right) = (\boldsymbol{y}\times\boldsymbol{z})\cdot\boldsymbol{x}\end{aligned}$$

and, from eq. (A.8), $(\boldsymbol{x}\times\boldsymbol{y})\cdot\boldsymbol{z} = -(\boldsymbol{y}\times\boldsymbol{x})\cdot\boldsymbol{z} = -(\boldsymbol{x}\times\boldsymbol{z})\cdot\boldsymbol{y} = (\boldsymbol{z}\times\boldsymbol{x})\cdot\boldsymbol{y}$, then the scalar triple product satisfies the property

$$(\boldsymbol{x}\times\boldsymbol{y})\cdot\boldsymbol{z} = (\boldsymbol{y}\times\boldsymbol{z})\cdot\boldsymbol{x} = (\boldsymbol{z}\times\boldsymbol{x})\cdot\boldsymbol{y}, \quad \forall\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\in\mathbb{R}^3 \tag{A.10}$$

called invariance under cyclic permutations of the vectors. Putting $\boldsymbol{z} = \boldsymbol{x}\vee\boldsymbol{y}$ gives

$$(\boldsymbol{x}\times\boldsymbol{y})\cdot\boldsymbol{x} = (\boldsymbol{x}\times\boldsymbol{x})\cdot\boldsymbol{y} = 0, \quad \forall\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^3 \tag{A.11a}$$

$$(\boldsymbol{x}\times\boldsymbol{y})\cdot\boldsymbol{y} = (\boldsymbol{y}\times\boldsymbol{y})\cdot\boldsymbol{z} = 0, \quad \forall\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^3 \tag{A.11b}$$

so the cross product between two vectors is a vector perpendicular to *both*. The mixed product is used to characterise triples of linearly independent vectors: the triple $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ is said **right handed** if $(\boldsymbol{x}\times\boldsymbol{y})\cdot\boldsymbol{z} > 0$, or **left handed** if $(\boldsymbol{x}\times\boldsymbol{y})\cdot\boldsymbol{z} < 0$.

Triple Product

The cross product of a vector with the cross product between other two vectors is called **vector triple product**. Considering that

$$\begin{aligned}\boldsymbol{x}\times(\boldsymbol{y}\times\boldsymbol{z}) &= \left(x_2\left(y_1z_2 - y_2z_1\right) - x_3\left(y_3z_1 - y_1z_3\right)\right)\boldsymbol{i} + \\ &+ \left(x_3\left(y_2z_3 - y_3z_2\right) - x_1\left(y_1z_2 - y_2z_1\right)\right)\boldsymbol{j} + \\ &+ \left(x_1\left(y_3z_1 - y_1z_3\right) - x_2\left(y_2z_3 - y_3z_2\right)\right)\boldsymbol{k} = \\ &= \left(y_1\left(x_2z_2 + x_3z_3\right) - z_1\left(x_2y_2 + x_3y_3\right) + x_1y_1z_1 - x_1y_1z_1\right)\boldsymbol{i} + \\ &+ \left(y_2\left(x_3z_3 + x_1z_1\right) - z_2\left(x_3y_3 + x_1y_1\right) + x_2y_2z_2 - x_2y_2z_2\right)\boldsymbol{j} + \\ &+ \left(y_3\left(x_1z_1 + x_2z_2\right) - z_3\left(x_1y_1 + x_2y_2\right) + x_3y_3z_3 - x_3y_3z_3\right)\boldsymbol{k}\end{aligned}$$

the vector triple product satisfies the property

$$\boldsymbol{x}\times(\boldsymbol{y}\times\boldsymbol{z}) = (\boldsymbol{z}\cdot\boldsymbol{x})\,\boldsymbol{y} - (\boldsymbol{x}\cdot\boldsymbol{y})\,\boldsymbol{z}, \quad \forall\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\in\mathbb{R}^3 \tag{A.12}$$

also known as Lagrange's formula.

Properties

Using concurrently eqs. (A.10) and (A.12) yields

$$(\boldsymbol{x}\times\boldsymbol{y})\cdot(\boldsymbol{x}\times\boldsymbol{y}) = (\boldsymbol{y}\times(\boldsymbol{x}\times\boldsymbol{y}))\cdot\boldsymbol{x} = ((\boldsymbol{y}\cdot\boldsymbol{y})\,\boldsymbol{x} - (\boldsymbol{y}\cdot\boldsymbol{x})\,\boldsymbol{y})\cdot\boldsymbol{x}$$

thus the squared norm of the cross product is given by

$$\|\boldsymbol{x} \times \boldsymbol{y}\|^2 = \|\boldsymbol{x}\|^2 \|\boldsymbol{y}\|^2 - (\boldsymbol{x} \cdot \boldsymbol{y})^2 \tag{A.13}$$

or, alternatively, applying eq. (A.5), by $\|\boldsymbol{x} \times \boldsymbol{y}\|^2 = \|\boldsymbol{x}\|^2 \|\boldsymbol{y}\|^2 \sin^2 \theta$.

As $\boldsymbol{x} \times \boldsymbol{y} = \boldsymbol{0} \Leftrightarrow \|\boldsymbol{x} \times \boldsymbol{y}\| = 0$, eqs. (A.13) and (A.4) give, combined with eq. (A.9)

$$\boldsymbol{x} \times \boldsymbol{y} = \boldsymbol{0}, \boldsymbol{x} \neq \boldsymbol{0} \iff \exists \lambda \in \mathbb{R} \colon \boldsymbol{y} = \lambda \boldsymbol{x} \tag{A.14}$$

that is the cross product of two vectors is zero only if they are linearly dependent.

A.2 Matrices

A.2.1 Fundamentals

Real **matrices** are 2-dimensional arrays of real numbers: a $m \times n$ matrix $\boldsymbol{A}$ is collection of mn real scalars

$$\boldsymbol{A} = \left(a_{ij}\right)_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} \in \mathbb{R}^{mn}$$

arranged in a rectangular fashion with m **rows** and n **columns**

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

where a_{ij} is the entry in the i-th row and j-th column. A $m \times n$ matrix is thus made up of m n-dimensional vectors, i.e. its rows, or n m-dimensional vectors, i.e. its columns; when $m > n$, the matrix is said to be **tall**, when $m < n$, the matrix is said to be **fat** and when $m = n$, the matrix is said to be **square**. A $n \times n$ matrix is said to be **square** of order n. The **set** of $m \times n$ matrices is a vector space, denoted by $\mathbb{R}^{m \times n}$ or $\mathbb{R}^{m,n}$, isomorphic to the Euclidean space $\mathbb{R}^{mn}$.

Vectors can be seen as special matrices: a n-dimensional vector $\boldsymbol{b} \in \mathbb{R}^n$ can be represented either as a column

$$\boldsymbol{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \in \mathbb{R}^{n \times 1}$$

or as a row

$$\boldsymbol{b} = \begin{pmatrix} b_1 & \cdots & b_n \end{pmatrix} \in \mathbb{R}^{1 \times n}$$

in order to define operations involving *both* matrices and vectors.

Linearity

Since matrices with same size make up a vector space, the operations of **sum** and **product by scalar** on matrices, over the field of real numbers, are naturally defined. If $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times n}$ and $\lambda, \mu \in \mathbb{R}$, then $\boldsymbol{C} = \lambda \boldsymbol{A} + \mu \boldsymbol{B} \in \mathbb{R}^{m \times n}$ is the matrix with

$$c_{ij} \coloneqq \lambda a_{ij} + \mu b_{ij}, \quad 1 \leq i \leq m, 1 \leq j \leq n$$

as entry in the i-th row and the j-th column.

Matrix Product

A **matrix product** can be defined for **conformable matrices**, i.e. if the number of columns of a matrix equals the number of rows of the other matrix. If $\boldsymbol{A} \in \mathbb{R}^{l \times m}$ and $\boldsymbol{B} \in \mathbb{R}^{m \times n}$, the product $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{B} \in \mathbb{R}^{l \times n}$ is the matrix with

$$c_{ik} \coloneqq \sum_{j=1}^{m} a_{ij} b_{jk}, \quad 1 \leq i \leq l, 1 \leq k \leq n$$

as entry in the i-th row and j-th column. It is not commutative, in general.

Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^{n \times 1}$ be two vectors. The scalar given by

$$\boldsymbol{x}^{\mathsf{T}} \boldsymbol{y} \coloneqq x_1 y_1 + \ldots + x_n y_n \in \mathbb{R}$$

is called **inner product** between $\boldsymbol{x}$ and $\boldsymbol{y}$ and corresponds to the dot product $\boldsymbol{x} \cdot \boldsymbol{y}$ written using matrix product. The matrix given by

$$\boldsymbol{x}\boldsymbol{y}^{\mathsf{T}} \coloneqq \begin{pmatrix} y_1 x_1 & \cdots & y_n x_1 \\ \vdots & \ddots & \vdots \\ y_1 x_n & \cdots & y_n x_n \end{pmatrix} = \begin{pmatrix} y_1 \boldsymbol{x} & \cdots & y_n \boldsymbol{x} \end{pmatrix} = \begin{pmatrix} x_1 \boldsymbol{y}^{\mathsf{T}} \\ \vdots \\ x_n \boldsymbol{y}^{\mathsf{T}} \end{pmatrix} \in \mathbb{R}^{n \times n}$$

is called **outer product** between $\boldsymbol{x}$ and $\boldsymbol{y}$ and coincides with the dyadic product $\boldsymbol{x} \otimes \boldsymbol{y}$ written using matrix product.

Block Matrices

A **partitioned matrix** or **block matrix** $\boldsymbol{A}$ is a matrix made up of smaller sub-matrices

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \cdots & \boldsymbol{A}_{1n} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \cdots & \boldsymbol{A}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}_{m1} & \boldsymbol{A}_{m2} & \cdots & \boldsymbol{A}_{mn} \end{pmatrix}$$

arranged in m row partitions and n column partitions. Let $\boldsymbol{A}$ be a block matrix with l row and m column partitions, and $\boldsymbol{B}$ be a block matrix with m row and n column partitions: if they are conformable and **conformably partitioned**, i.e. $\boldsymbol{A}_{ij}$ and $\boldsymbol{B}_{jk}$

are conformable $\forall 1 \leq i \leq l, 1 \leq j \leq m, 1 \leq k \leq n$, the product $\boldsymbol{C} = \boldsymbol{AB}$ has l row partitions and n column partitions, with the sub-matrix

$$\boldsymbol{C}_{ik} = \sum_{j=1}^{m} \boldsymbol{A}_{ij} \boldsymbol{B}_{jk}$$

as block in the i-th row partition and j-th column partition.

Transposition

The matrix $\boldsymbol{A}^\mathsf{T} \in \mathbb{R}^{n \times m}$ obtained switching the indices of rows and columns of $\boldsymbol{A} \in \mathbb{R}^{m \times n}$

$$\boldsymbol{A}^\mathsf{T} \coloneqq \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{pmatrix}$$

is called **transpose** of $\boldsymbol{A}$; the transposition trivially satisfies $(\boldsymbol{A}^\mathsf{T})^\mathsf{T} = \boldsymbol{A}$.

Let $\boldsymbol{C} = \boldsymbol{A}^\mathsf{T}$, $\boldsymbol{D} = \boldsymbol{B}^\mathsf{T}$ be the transposes of $\boldsymbol{A} \in \mathbb{R}^{l \times m}$, $\boldsymbol{B} \in \mathbb{R}^{m \times n}$: since

$$\sum_{j=1}^{m} a_{kj} b_{ji} = \sum_{j=1}^{m} b_{ji} a_{kj} = \sum_{j=1}^{m} d_{ij} c_{jk}$$

then the transposition satisfies

$$(\boldsymbol{AB})^\mathsf{T} = \boldsymbol{B}^\mathsf{T} \boldsymbol{A}^\mathsf{T}, \ \forall \boldsymbol{A} \in \mathbb{R}^{l \times m}, \boldsymbol{B} \in \mathbb{R}^{m \times n} \tag{A.15}$$

that is to say the transpose of the product of two matrices is the product of their transposes with reverse order.

A.2.2 Linear Map

Image and Kernel

Any matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is closely connected to a linear map between euclidean spaces, namely the vector-valued vector function $\boldsymbol{f} \colon \mathbb{R}^n \to \mathbb{R}^m$ defined as $\boldsymbol{f}(\boldsymbol{x}) \coloneqq \boldsymbol{Ax}$.

The **Image** $\operatorname{im} \boldsymbol{A}$ or **Range Space** $\mathcal{R}(\boldsymbol{A})$ of $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is the subspace of $\mathbb{R}^m$

$$\mathcal{R}(\boldsymbol{A}) \coloneqq \{\boldsymbol{y} \in \mathbb{R}^m \colon \boldsymbol{y} = \boldsymbol{Ax}, \boldsymbol{x} \in \mathbb{R}^n\}$$

that is the set of values of $\boldsymbol{Ax}$. Its dimension is denoted as $\rho(\boldsymbol{A}) \coloneqq \dim \mathcal{R}(\boldsymbol{A})$.

The **Kernel** $\ker \boldsymbol{A}$ or **Null Space** $\mathcal{N}(\boldsymbol{A})$ of $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is the subspace of $\mathbb{R}^n$

$$\mathcal{N}(\boldsymbol{A}) \coloneqq \{\boldsymbol{x} \in \mathbb{R}^n \colon \boldsymbol{Ax} = \boldsymbol{0}\}$$

that is the set of zeros of $\boldsymbol{Ax}$. Its dimension is denoted as $\nu(\boldsymbol{A}) \coloneqq \dim \mathcal{R}(\boldsymbol{A})$.

As $\mathcal{R}(\boldsymbol{A})$ is a subspace of $\mathbb{R}^m$, using the direct sum[2] of complementary subspaces, the m-dimensional Euclidean space may be decomposed as $\mathbb{R}^m = \mathcal{R}(\boldsymbol{A}) \oplus \mathcal{R}^\perp(\boldsymbol{A})$, where $\mathcal{R}^\perp(\boldsymbol{A})$ is the orthogonal complement[3] of $\mathcal{R}(\boldsymbol{A})$. Every $\boldsymbol{y} \in \mathbb{R}^m$ can be thus *uniquely* written as $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$, with $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{b} \in \mathcal{R}^\perp(\boldsymbol{A})$. By definition, $\boldsymbol{b}$ is orthogonal to $\mathcal{R}(\boldsymbol{A})$, therefore it satisfies $\boldsymbol{0} = (\boldsymbol{A}\boldsymbol{x})^\mathsf{T}\boldsymbol{b} = \boldsymbol{x}^\mathsf{T}\boldsymbol{A}^\mathsf{T}\boldsymbol{b}$, $\forall \boldsymbol{x} \in \mathbb{R}^n \Leftrightarrow \boldsymbol{A}^\mathsf{T}\boldsymbol{b} = \boldsymbol{0}$, implying that $\boldsymbol{b} \in \mathcal{N}(\boldsymbol{A}^\mathsf{T})$ and $\mathcal{R}^\perp(\boldsymbol{A}) = \mathcal{N}(\boldsymbol{A}^\mathsf{T})$. By applying the same reasoning to $\boldsymbol{A}^\mathsf{T}$, the n-dimensional Euclidean space may be decomposed as $\mathbb{R}^n = \mathcal{R}(\boldsymbol{A}^\mathsf{T}) \oplus \mathcal{R}^\perp(\boldsymbol{A}^\mathsf{T})$, where $\mathcal{R}^\perp(\boldsymbol{A}^\mathsf{T}) = \mathcal{N}((\boldsymbol{A}^\mathsf{T})^\mathsf{T}) = \mathcal{N}(\boldsymbol{A})$. The four subspaces thus satisfy

$$\mathbb{R}^n = \mathcal{R}(\boldsymbol{A}^\mathsf{T}) \oplus \mathcal{N}(\boldsymbol{A}) \tag{A.16a}$$

$$\mathbb{R}^m = \mathcal{R}(\boldsymbol{A}) \oplus \mathcal{N}(\boldsymbol{A}^\mathsf{T}) \tag{A.16b}$$

which is known as **fundamental theorem of linear algebra**.

If $\nu = \nu(\boldsymbol{A})$, there exist ν linearly independent vectors $\boldsymbol{\zeta}_1, \ldots, \boldsymbol{\zeta}_\nu \in \mathbb{R}^n$ spanning $\mathcal{N}(\boldsymbol{A})$ and $n-\nu$ linearly independent vectors $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_{n-\nu} \in \mathbb{R}^n$ spanning $\mathcal{N}^\perp(\boldsymbol{A})$, hence $\boldsymbol{\zeta}_1, \ldots, \boldsymbol{\zeta}_\nu, \boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_{n-\nu}$ make up a basis of $\mathbb{R}^n$; every $\boldsymbol{x} \in \mathbb{R}^n$ is *uniquely* written as a linear combination of these vectors $\boldsymbol{x} = \lambda_1\boldsymbol{\zeta}_1 + \ldots + \lambda_\nu\boldsymbol{\zeta}_\nu + \mu_1\boldsymbol{\xi}_1 + \ldots + \mu_{n-\nu}\boldsymbol{\xi}_{n-\nu}$ that gives image $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}(\lambda_1\boldsymbol{\zeta}_1 + \ldots + \lambda_\nu\boldsymbol{\zeta}_\nu + \mu_1\boldsymbol{\xi}_1 + \ldots + \mu_{n-\nu}\boldsymbol{\xi}_{n-\nu}) = \mu_1\boldsymbol{A}\boldsymbol{\xi}_1 + \ldots + \mu_{n-\nu}\boldsymbol{A}\boldsymbol{\xi}_{n-\nu}$. The condition $\mu_1\boldsymbol{A}\boldsymbol{\xi}_1 + \ldots + \mu_{n-\nu}\boldsymbol{A}\boldsymbol{\xi}_{n-\nu} = \boldsymbol{A}(\mu_1\boldsymbol{\xi}_1 + \ldots + \mu_{n-\nu}\boldsymbol{\xi}_{n-\nu}) = \boldsymbol{0}$ requires that $\mu_1\boldsymbol{\xi}_1 + \ldots + \mu_{n-\nu}\boldsymbol{\xi}_{n-\nu} \in \mathcal{N}(\boldsymbol{A})$, but, since $\mu_1\boldsymbol{\xi}_1 + \ldots + \mu_{n-\nu}\boldsymbol{\xi}_{n-\nu} \in \mathcal{N}^\perp(\boldsymbol{A})$, then it automatically requires also that $\mu_1\boldsymbol{\xi}_1 + \ldots + \mu_{n-\nu}\boldsymbol{\xi}_{n-\nu} = \boldsymbol{0}$. As $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_{n-\nu}$ are linearly independent, such a condition can be met only when $\mu_i = 0$, $\forall i = 1, \ldots, n-\nu$, proving that $\boldsymbol{A}\boldsymbol{\xi}_1, \ldots, \boldsymbol{A}\boldsymbol{\xi}_{n-\nu} \in \mathbb{R}^m$ are $n-\nu$ linearly independent vectors spanning $\mathcal{R}(\boldsymbol{A})$, which has thus dimension $n-\nu$. The dimension of the range and of the kernel therefore satisfy

$$\rho(\boldsymbol{A}) + \nu(\boldsymbol{A}) = n \tag{A.17}$$

also known as **dimension theorem**.

Rank

The column rank of a matrix is the dimension of its column space, i.e. the subspace spanned by its columns, or, equivalently, its greatest number of linearly independent columns. For the column space of $\boldsymbol{A}$ is the range space of $\boldsymbol{A}$, the column rank is $\rho(\boldsymbol{A})$. The row rank of a matrix is the dimension of its row space, i.e. the subspace spanned by its rows, or, equivalently, its greatest number of linearly independent rows. For the row space of $\boldsymbol{A}$ is the range space of $\boldsymbol{A}^\mathsf{T}$, the row rank is $\rho(\boldsymbol{A}^\mathsf{T})$. Remembering that $\rho(\boldsymbol{A}^\mathsf{T}) + \nu(\boldsymbol{A}) = n$ and $\rho(\boldsymbol{A}) + \nu(\boldsymbol{A}) = n$ because of eqs. (A.16a) and (A.17) respectively, the row rank and the column rank actually coincide and are referred simply as rank. The **rank** of a matrix $\boldsymbol{A}$ is then the non negative integer $\operatorname{rk}\boldsymbol{A} \in \mathbb{N}$ equal to the maximum number of linearly independent columns or rows in the matrix.

[2]The **direct sum** between two subsets of the same set $X \subseteq Z$ and $Y \subseteq Z$, is the subset defined by $X \oplus Y \coloneqq \{z \in Z \colon z = x + y, x \in X \wedge y \in Y\}$.

[3]The **orthogonal complement** of a subspace $V \subseteq W$ of vector space equipped with a scalar product, is the subspace $V^\perp \subseteq W$ defined as $V^\perp \coloneqq \{w \in W \colon \langle w, v\rangle = 0, \forall v \in V\}$.

The rank of a matrix cannot exceed the number of either its rows or its columns, thus, in general, $\operatorname{rk} \boldsymbol{A} \leq \min\{m, n\}, \forall \boldsymbol{A} \in \mathbb{R}^{m \times n}$: in particular, if the equality holds, $\boldsymbol{A}$ is said to be full rank, otherwise $\boldsymbol{A}$ is said to be not full rank or rank deficient. Transposing means swapping column space and row space, thus $\operatorname{rk} \boldsymbol{A}^{\mathsf{T}} = \operatorname{rk} \boldsymbol{A}$, $\forall \boldsymbol{A} \in \mathbb{R}^{m \times n}$, that is the rank is invariant under transposition. Since each column of $\boldsymbol{AB}$ is a linear combination of the columns of $\boldsymbol{A}$ while each row of $\boldsymbol{AB}$ is a linear combination of the rows of $\boldsymbol{B}$, then $\operatorname{rk}(\boldsymbol{AB}) \leq \min\{\operatorname{rk}(\boldsymbol{A}), \operatorname{rk}(\boldsymbol{B})\}$, $\forall \boldsymbol{A} \in \mathbb{R}^{m \times n}, \boldsymbol{B} \in \mathbb{R}^{n \times p}$.

A.2.3 Square Matrices

The **main diagonal** of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is the set of the entries with equal row and column indices, i.e. $a_{11}, \ldots, a_{nn}$. Among square matrices, there are some of them with a particular structure with respect to the main diagonal.

A square matrix is said to be **triangular** if its entries either below or above the main diagonal are all zero: a square matrix $\boldsymbol{U} \in \mathbb{R}^{n \times n}$ is said to be **upper triangular** if its entries below the main diagonal are all zero, i.e. $u_{ij} = 0, \forall i, j = 1, \ldots, n \colon i > j$; a square matrix $\boldsymbol{L} \in \mathbb{R}^{n \times n}$ is said to be **lower triangular** if its entries above the main diagonal are all zero, i.e. $l_{ij} = 0, \forall i, j = 1, \ldots, n \colon i < j$.

A square matrix $\boldsymbol{S} \in \mathbb{R}^{n \times n}$ is said to be **symmetric** if its entries symmetric about the main diagonal are equal, i.e. $s_{ji} = s_{ij}, \forall i, j = 1, \ldots, n$, or $\boldsymbol{S}^{\mathsf{T}} = \boldsymbol{S}$. There are

$$\frac{n^2 - n}{2} + n = \frac{n^2 + n}{2} = \frac{n(n+1)}{2} \tag{A.18}$$

free entries in a symmetric matrix of order n. A symmetric matrix $\boldsymbol{S} \in \mathbb{R}^{n \times n}$ is said to be **positive definite** if $\boldsymbol{x}^{\mathsf{T}} \boldsymbol{S} \boldsymbol{x} > 0, \forall \boldsymbol{x} \in \mathbb{R}^n \backslash \{\boldsymbol{0}\}$, **positive semi-definite** if $\boldsymbol{x}^{\mathsf{T}} \boldsymbol{S} \boldsymbol{x} \geq 0, \forall \boldsymbol{x} \in \mathbb{R}^n$, **negative definite** if $\boldsymbol{x}^{\mathsf{T}} \boldsymbol{S} \boldsymbol{x} < 0, \forall \boldsymbol{x} \in \mathbb{R}^n \backslash \{\boldsymbol{0}\}$, **negative semi-definite** if $\boldsymbol{x}^{\mathsf{T}} \boldsymbol{S} \boldsymbol{x} \leq 0, \forall \boldsymbol{x} \in \mathbb{R}^n$ and **indefinite** otherwise. The product $\boldsymbol{A}^{\mathsf{T}} \boldsymbol{A}$, where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, is always positive semi-definite, as $\boldsymbol{x}^{\mathsf{T}} \boldsymbol{A}^{\mathsf{T}} \boldsymbol{A} \boldsymbol{x} = \|\boldsymbol{A} \boldsymbol{x}\|^2$.

A square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is said to be **anti-symmetric** or **skew-symmetric**, if its entries symmetric about the main diagonal are opposite, i.e. $a_{ji} = -a_{ij}, \forall i, j = 1, \ldots, n$, or $\boldsymbol{A}^{\mathsf{T}} = -\boldsymbol{A}$. There are

$$\frac{n^2 - n}{2} = \frac{n(n-1)}{2} \tag{A.19}$$

free entries in a skew-symmetric matrix of order n.

A square matrix $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ is said to be **diagonal** if its entries outside the main diagonal are all zero, i.e. $d_{ij} = 0, \forall i, j = 1, \ldots, n \colon i \neq j$. A diagonal matrix can be denoted also as $\boldsymbol{D} = \operatorname{\mathbf{diag}}(\boldsymbol{d})$, where $\boldsymbol{d} \in \mathbb{R}^n$ is a vector made up of the diagonal entries.

The most important diagonal matrix is the **identity matrix** $\boldsymbol{I} \in \mathbb{R}^{n \times n}$

$$\boldsymbol{I} \coloneqq \left(\delta_{ij}\right)_{\substack{1 \leq i \leq n \\ 1 \leq j \leq n}}$$

that is the matrix with 1 on the main diagonal and 0 elsewhere. The notation $\boldsymbol{I}_n$ is used when the order n has to be specified. The identity matrix is the identity element of matrix product as trivially satisfies $\boldsymbol{I}_m \boldsymbol{A} = \boldsymbol{A}$, $\boldsymbol{A} \boldsymbol{I}_n = \boldsymbol{A}$, $\forall \boldsymbol{A} \in \mathbb{R}^{m \times n}$.

Inversion

If there exists a square matrix $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ such that $\boldsymbol{AB} = \boldsymbol{BA} = \boldsymbol{I}$, then the square matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is said to be invertible and the matrix $\boldsymbol{B}$ is said the inverse matrix of $\boldsymbol{A}$. If $\boldsymbol{A}$ is **invertible**, its **inverse** $\boldsymbol{A}^{-1}$ is the matrix satisfying

$$\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I}$$

that is multiplying matrix and its inverse, in either order, gives the identity matrix.

If $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is full rank, all vectors in $\mathbb{R}^n$, and in particular the columns of $\boldsymbol{I}$, may be expressed as unique linear combinations of the columns of $\boldsymbol{A}$, hence full rank matrices are invertible. If $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is invertible, $n = \operatorname{rk} \boldsymbol{I} = \operatorname{rk} \boldsymbol{A}\boldsymbol{A}^{-1} \leq \min\left\{\operatorname{rk} \boldsymbol{A}, \operatorname{rk} \boldsymbol{A}^{-1}\right\}$ and $\operatorname{rk} \boldsymbol{A} \leq n$ give $\operatorname{rk} \boldsymbol{A} = n$, hence invertible matrices are full rank.

If $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is invertible, since $(\boldsymbol{A}^{-1})^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}} = (\boldsymbol{A}\boldsymbol{A}^{-1})^{\mathsf{T}}$ because of eq. (A.15), then

$$(\boldsymbol{A}^{\mathsf{T}})^{-1} = (\boldsymbol{A}^{-1})^{\mathsf{T}}$$

that is inverse of the transpose is the transpose of the inverse.

If $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times n}$ are both invertible, since $\boldsymbol{B}^{-1}\boldsymbol{A}^{-1}\boldsymbol{AB} = \boldsymbol{I}$, then

$$(\boldsymbol{AB})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1}$$

that is the inverse of a product is the product of the inverses with reversed order.

A square matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ is said to be **orthogonal** if its columns and rows are orthonormal vectors, that is $\boldsymbol{Q}\boldsymbol{Q}^{\mathsf{T}} = \boldsymbol{Q}^{\mathsf{T}}\boldsymbol{Q} = \boldsymbol{I}$, or, equivalently $\boldsymbol{Q}^{-1} = \boldsymbol{Q}^{\mathsf{T}}$.

Trace

The **trace** of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is the number $\operatorname{tr} \boldsymbol{A} \in \mathbb{R}$ defined as

$$\operatorname{tr} \boldsymbol{A} \coloneqq \sum_{i=1}^{n} a_{ii} \tag{A.20}$$

that is the sum of all the elements on the matrix main diagonal. The trace is linear

$$\operatorname{tr}(\lambda\boldsymbol{A} + \mu\boldsymbol{B}) = \lambda \operatorname{tr}(\boldsymbol{A}) + \mu \operatorname{tr}(\boldsymbol{B}), \forall \boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times n}, \forall \lambda, \mu \in \mathbb{R}$$

and is invariant to transposition

$$\operatorname{tr} \boldsymbol{A}^{\mathsf{T}} = \operatorname{tr} \boldsymbol{A},\ \forall \boldsymbol{A} \in \mathbb{R}^{n\times n}$$

as verifiable from the definition. Considering $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{n\times m}$, since

$$\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}b_{ji} = \sum_{j=1}^{n}\sum_{i=1}^{m} b_{ji}a_{ij}$$

then the trace satisfies the property

$$\operatorname{tr}(\boldsymbol{AB}) = \operatorname{tr}(\boldsymbol{BA}),\ \forall \boldsymbol{A} \in \mathbb{R}^{m\times n}, \boldsymbol{B} \in \mathbb{R}^{n\times m}$$

of trace invariance under cyclic permutations. Applied to $\boldsymbol{M}^{-1}\boldsymbol{AM}$ gives the property

$$\operatorname{tr}\left(\boldsymbol{M}^{-1}\boldsymbol{AM}\right) = \operatorname{tr}(\boldsymbol{A}), \forall \boldsymbol{A}, \boldsymbol{M} \in \mathbb{R}^{n\times n} \colon \operatorname{rk} \boldsymbol{M} = n$$

of trace invariance under similarity transformations.

Determinant

The **determinant** of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is the number $\det \boldsymbol{A} \in \mathbb{R}$ recursively defined through the Laplace's rule with respect to the i-th row or the j-th column

$$\det \boldsymbol{A} \coloneqq \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det \boldsymbol{A}_{ij} = \sum_{i=1}^{n} (-1)^{i+j} a_{ij} \det \boldsymbol{A}_{ij} \tag{A.21}$$

where $\boldsymbol{A}_{ij}$ is the $(n-1) \times (n-1)$ matrix obtained removing the i-th row and the j-th column from $\boldsymbol{A}$ and $\det a_{ij} \coloneqq a_{ij}$. The term $\det \boldsymbol{A}_{ij}$ is called ij **first minor**, while the term $(-1)^{i+j} \det \boldsymbol{A}_{ij}$ is called ij **cofactor**. The determinant is multi-linear

$$\det (\lambda \boldsymbol{A}) = \lambda^n \det \boldsymbol{A}, \ \forall \boldsymbol{A} \in \mathbb{R}^{n \times n}, \forall \lambda \in \mathbb{R}$$

and is invariant to transposition

$$\det \boldsymbol{A}^{\mathsf{T}} = \det \boldsymbol{A}, \ \forall \boldsymbol{A} \in \mathbb{R}^{n \times n}$$

as verifiable from the definition.

The basic properties of the determinant can be inferred directly from its definition: if two rows/columns are swapped, the determinant changes sign; if two rows/columns are equal, the determinant is zero, as swapping them does not alter the matrix but the determinant changes sign. If a row/column is multiplied by a scalar λ and added to another row/column, the determinant does not change, as the expansion with respect to the modified row/column yields the sum between the original determinant and λ times the determinant of a matrix with two equal rows or columns, which is zero.

The determinant of block triangular matrices comes from the product of the determinants of all diagonal blocks; in particular

$$\begin{gathered} \det \begin{pmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{O} & \boldsymbol{A}_{22} \end{pmatrix} = \det \begin{pmatrix} \boldsymbol{A}_{11} & \boldsymbol{O} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{pmatrix} = \det \boldsymbol{A}_{11} \det \boldsymbol{A}_{22} \\ \forall \boldsymbol{A}_{11} \in \mathbb{R}^{m \times m}, \boldsymbol{A}_{12} \in \mathbb{R}^{m \times n}, \boldsymbol{A}_{21} \in \mathbb{R}^{n \times m}, \boldsymbol{A}_{22} \in \mathbb{R}^{n \times n} \end{gathered} \tag{A.22}$$

and

$$\begin{gathered} \det \begin{pmatrix} \boldsymbol{O} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{pmatrix} = \det \begin{pmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{O} \end{pmatrix} = (-1)^{mn} \det \boldsymbol{A}_{12} \det \boldsymbol{A}_{21} \\ \forall \boldsymbol{A}_{11} \in \mathbb{R}^{n \times m}, \boldsymbol{A}_{12} \in \mathbb{R}^{m \times m}, \boldsymbol{A}_{21} \in \mathbb{R}^{n \times n}, \boldsymbol{A}_{22} \in \mathbb{R}^{m \times n} \end{gathered} \tag{A.23}$$

as verifiable from eq. (A.21). Considering that in the following product between conformably partitioned block matrices

$$\begin{pmatrix} \boldsymbol{I} & \boldsymbol{L} \\ \boldsymbol{O} & \boldsymbol{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{M} & \boldsymbol{N} \\ \boldsymbol{P} & \boldsymbol{Q} \end{pmatrix} = \begin{pmatrix} \boldsymbol{M} + \boldsymbol{L}\boldsymbol{P} & \boldsymbol{N} + \boldsymbol{L}\boldsymbol{Q} \\ \boldsymbol{P} & \boldsymbol{Q} \end{pmatrix}$$

the rows of $\begin{pmatrix} \boldsymbol{M} + \boldsymbol{L}\boldsymbol{P} & \boldsymbol{N} + \boldsymbol{L}\boldsymbol{Q} \end{pmatrix}$ are the sum between rows of $\begin{pmatrix} \boldsymbol{M} & \boldsymbol{N} \end{pmatrix}$ and a linear combination of the rows of $\begin{pmatrix} \boldsymbol{P} & \boldsymbol{Q} \end{pmatrix}$, then taking the determinant of both sides of

$$\begin{pmatrix} \boldsymbol{I} & \boldsymbol{A} \\ \boldsymbol{O} & \boldsymbol{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{A} & \boldsymbol{O} \\ -\boldsymbol{I} & \boldsymbol{B} \end{pmatrix} = \begin{pmatrix} \boldsymbol{O} & \boldsymbol{A}\boldsymbol{B} \\ -\boldsymbol{I} & \boldsymbol{B} \end{pmatrix}$$

gives the fundamental property, known as **Binet formula**

$$\det \boldsymbol{AB} = \det \boldsymbol{A} \det \boldsymbol{B}, \quad \forall \boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times n} \tag{A.24}$$

that is the determinant of a product is the product of the determinants.

If $\det \boldsymbol{A} = 0$, then $\boldsymbol{A}$ is said to be **singular**, whereas if $\det \boldsymbol{A} \neq 0$, then $\boldsymbol{A}$ is said to be **non singular**. A rank deficient matrix is always singular, because at least one of its rows/columns can be written as a linear combination of its other rows/columns.

The adjugate matrix $\mathbf{adj}\, \boldsymbol{A}$ of a $n \times n$ matrix $\boldsymbol{A}$ is the $n \times n$ matrix

$$\mathbf{adj}\, \boldsymbol{A} = \begin{pmatrix} \det \boldsymbol{A}_{11} & -\det \boldsymbol{A}_{21} & \cdots & (-1)^{n+1} \det \boldsymbol{A}_{n1} \\ -\det \boldsymbol{A}_{12} & \det \boldsymbol{A}_{22} & \cdots & (-1)^{n+2} \det \boldsymbol{A}_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ (-1)^{1+n} \det \boldsymbol{A}_{1n} & (-1)^{2+n} \det \boldsymbol{A}_{2n} & \cdots & \det \boldsymbol{A}_{nn} \end{pmatrix}$$

that is the transpose of the matrix with the cofactors of $\boldsymbol{A}$ as entries. Since

$$\sum_{k=1}^{n} a_{ik} (-1)^{j+k} \det \boldsymbol{A}_{jk} = \sum_{k=1}^{n} (-1)^{i+k} \det \boldsymbol{A}_{ki} a_{kj} = \delta_{ij} \det \boldsymbol{A}$$

then

$$\boldsymbol{A}\, \mathbf{adj}\, \boldsymbol{A} = \mathbf{adj}\, \boldsymbol{A} \boldsymbol{A} = (\det \boldsymbol{A}) \boldsymbol{I} \tag{A.25}$$

which gives an expression of the inverse of $\boldsymbol{A}$

$$\boldsymbol{A}^{-1} = \frac{1}{\det \boldsymbol{A}} \mathbf{adj}\, \boldsymbol{A}$$

whenever $\det \boldsymbol{A} \neq 0$.

A.2.4 Spectral Theory

Complex vectors

The following analysis cannot be restrained to real numbers only. Complex vectors are arrays of complex numbers: a n-dimensional complex vector $\boldsymbol{z} = (z_i)_{1 \leq i \leq n} \in \mathbb{C}^n$, is defined as $\boldsymbol{z} \coloneqq \boldsymbol{x} + i\boldsymbol{y}$, where $\boldsymbol{x} = (x_i)_{1 \leq i \leq n} \in \mathbb{R}^n$ and $\boldsymbol{y} = (y_i)_{1 \leq i \leq n} \in \mathbb{R}^n$ are termed respectively real part and imaginary part.

A scalar product between two complex vectors $\boldsymbol{z}, \boldsymbol{w} \in \mathbb{C}^n$ can be defined as

$$\boldsymbol{z} \cdot \boldsymbol{w} \coloneqq \sum_{i=1}^{n} z_i {w_i}^{\star}$$

that is the standard dot product between $\boldsymbol{z}$ and the **conjugate** of $\boldsymbol{w}$. It is *not* commutative, but satisfies the conjugate symmetry property $(\boldsymbol{z} \cdot \boldsymbol{w})^{\star} = \boldsymbol{z} \cdot \boldsymbol{w}$. This scalar product induces the euclidean norm

$$\|\boldsymbol{z}\| \coloneqq \sqrt{\boldsymbol{z} \cdot \boldsymbol{z}} = \sqrt{\sum_{i=1}^{n} z_i {z_i}^{\star}} = \sqrt{\sum_{i=1}^{n} |z_i|^2} = \sqrt{\sum_{i=1}^{n} |x_i + i y_i|^2} = \sqrt{\sum_{i=1}^{n} x_i^2 + y_i^2}$$

the square of which satisfies the condition

$$\|\boldsymbol{z}\|^2 = \boldsymbol{z} \cdot \boldsymbol{z} = (\boldsymbol{x} + i\boldsymbol{y}) \cdot (\boldsymbol{x} + i\boldsymbol{y}) = \boldsymbol{x} \cdot \boldsymbol{x} + i\boldsymbol{y} \cdot \boldsymbol{x} - i\boldsymbol{x} \cdot \boldsymbol{y} - i^2 \boldsymbol{y} \cdot \boldsymbol{y} = \|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2$$

thus the norm of complex vectors come from the norm of their real and imaginary part.

Complex matrices are defined similarly; in this regard, it is useful to introduce the **hermitian transpose** of a matrix $\boldsymbol{A}$, that is the transpose of the conjugate $\boldsymbol{A}^{\mathsf{H}} \coloneqq \boldsymbol{A}^{\star\mathsf{T}}$. Writing complex vectors as columns $\boldsymbol{z}, \boldsymbol{w} \in \mathbb{C}^{n \times 1}$, the scalar product becomes $\boldsymbol{w}^{\mathsf{H}} \boldsymbol{z}$.

Eigenvalues and Eigenvectors

The scalar $\lambda \in \mathbb{C}$ is called **eigenvalue** or **characteristic value** of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ if there exists a non zero vector $\boldsymbol{v} \in \mathbb{C}^n$ satisfying the condition

$$\boldsymbol{A}\boldsymbol{v} = \lambda \boldsymbol{v} \tag{A.26}$$

called **eigenequation** of $\boldsymbol{A}$: pre-multiplying $\boldsymbol{v}$ by $\boldsymbol{A}$ returns still $\boldsymbol{v}$ scaled by λ. The zero vector is not a case of interest, as it trivially satisfies this equation for any scalar. Any vector $\boldsymbol{v}$ satisfying eq. (A.26) is called **eigenvector** or **characteristic vector** of the matrix $\boldsymbol{A}$ associated to the **eigenvalue** λ; the pair $(\lambda, \boldsymbol{v})$ is called **eigenpair** of $\boldsymbol{A}$.

The eigenequation can be rewritten as the homogeneous system of linear equations

$$\boldsymbol{A}\boldsymbol{v} - \lambda \boldsymbol{v} = (\boldsymbol{A} - \lambda \boldsymbol{I})\, \boldsymbol{v} = \boldsymbol{0} \tag{A.27}$$

which is consistent, as it always admits at least the trivial solution $\boldsymbol{v} = \boldsymbol{0}$ while other solutions $\boldsymbol{v} \neq \boldsymbol{0}$ may exist only if the columns of $\boldsymbol{A} - \lambda \boldsymbol{I}$ are linearly dependent, or in other words, if λ satisfies

$$\det(\lambda \boldsymbol{I} - \boldsymbol{A}) = 0 \tag{A.28}$$

called **characteristic equation** of $\boldsymbol{A}$. Once an eigenvalue λ_i is determined solving $\det(\lambda_i \boldsymbol{I} - \boldsymbol{A}) = 0$, the related eigenvectors $\boldsymbol{v}_i$ are determined solving $(\boldsymbol{A} - \lambda_i \boldsymbol{I})\, \boldsymbol{v}_i = \boldsymbol{0}$.

Spectrum

The left hand side of eq. (A.28) is a polynomial of degree n in λ with real coefficients

$$p(\lambda) = \det(\lambda \boldsymbol{I} - \boldsymbol{A}) = \lambda^n + a_{n-1}\lambda^{n-1} + \ldots + a_1 \lambda + a_0 \tag{A.29}$$

known as **characteristic polynomial** of $\boldsymbol{A}$. Its r roots are the eigenvalues of $\boldsymbol{A}$ hence

$$p(\lambda) = (\lambda - \lambda_1)^{\mu_1} \cdots (\lambda - \lambda_r)^{\mu_r} = \prod_{i=1}^{r} (\lambda - \lambda_i)^{\mu_i}, \quad \sum_{i=1}^{r} \mu_i = n$$

where, of course, $p(\lambda_i) = 0, \forall i = 1, \ldots, r$. The exponent μ_i is called **algebraic multiplicity** of λ_i, as it is the number of times it appears in the polynomial. The eigenvalue λ_i is called simple, if $\mu_i = 1$, or multiple, if $\mu_i > 1$.

According to the principle of polynomials identity, comparing

$$p(\lambda) = \det \begin{pmatrix} \lambda - a_{11} & -a_{1n} & \cdots & -a_{1n} \\ -a_{21} & \lambda - a_{22} & \cdots & -a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & \lambda - a_{nn} \end{pmatrix} = \lambda^n - \operatorname{tr}(\boldsymbol{A})\,\lambda^{n-1} + \ldots + (-1)^n \det(\boldsymbol{A})$$

with

$$p(\lambda) = \prod_{j=1}^{n} (\lambda - \lambda_j) = \lambda^n - \left(\sum_{j=1}^{n} \lambda_j \right) \lambda^{n-1} + \ldots + (-1)^n \prod_{j=1}^{n} \lambda_j$$

where $\lambda_j, j = 1, \ldots, n$ are the n eigenvalues, single or repeated, of $\boldsymbol{A}$, gives an alternative expression for the trace and the determinant of $\boldsymbol{A}$

$$\operatorname{tr} \boldsymbol{A} = \sum_{j=1}^{n} \lambda_j \tag{A.30a}$$

$$\det \boldsymbol{A} = \prod_{j=1}^{n} \lambda_j \tag{A.30b}$$

in terms of its eigenvalues.

The set $\lambda(\boldsymbol{A}) = \{\lambda_1, \ldots, \lambda_r\}$ containing all eigenvalues of $\boldsymbol{A}$ is called **spectrum** of $\boldsymbol{A}$. Because the characteristic polynomial has only real coefficients, from the fundamental theorem of algebra, if the complex value $\lambda = \sigma + i\omega$ is a root, its complex conjugate $\lambda^\star = \sigma - i\omega$ must also be a root: the spectrum contains either real eigenvalues or pairs of complex conjugate eigenvalues. Besides, if $(\lambda, \boldsymbol{v})$ is an eigenpair of a real matrix $\boldsymbol{A}$, then $(\lambda^\star, \boldsymbol{v}^\star)$ is also an eigenpair, since $\boldsymbol{A}\boldsymbol{v}^\star = (\boldsymbol{A}\boldsymbol{v})^\star = (\lambda \boldsymbol{v})^\star = \lambda^\star \boldsymbol{v}^\star$.

Transformations

The determinant is invariant to transposition

$$\det(\lambda \boldsymbol{I} - \boldsymbol{A}^\mathsf{T}) = \det(\lambda \boldsymbol{I} - \boldsymbol{A})^\mathsf{T} = \det(\lambda \boldsymbol{I} - \boldsymbol{A})$$

thus a matrix and its transpose have equal eigenvalues.

If $\boldsymbol{A}$ is non singular, pre-multiplying eq. (A.26) by $\boldsymbol{A}^{-1}$ gives

$$\boldsymbol{A}^{-1}\boldsymbol{v} = \frac{1}{\lambda}\boldsymbol{v}$$

thus a matrix and its inverse have equal eigenvectors and reciprocal eigenvalues.

If $\det \boldsymbol{M} \neq 0$, another square matrix $\boldsymbol{B}$ can be obtained from $\boldsymbol{A}$ via the similarity transformation $\boldsymbol{B} = \boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{M}$. As $\boldsymbol{A} = \boldsymbol{M}\boldsymbol{B}\boldsymbol{M}^{-1}$, eq. (A.26) becomes

$$\boldsymbol{B}\boldsymbol{M}^{-1}\boldsymbol{v} = \lambda \boldsymbol{M}^{-1}\boldsymbol{v}$$

thus, if $(\lambda, \boldsymbol{v})$ is an eigenpair of $\boldsymbol{A}$, $(\lambda, \boldsymbol{M}^{-1}\boldsymbol{v})$ is an eigenpair of $\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{M}$.

Eigenspace

An eigenvalue may give rise to more eigenvectors. The subspace $V_{\lambda_i} \coloneqq \mathcal{N}(\lambda_i \boldsymbol{I} - \boldsymbol{A})$ spanned by the eigenvectors associated to the eigenvalue λ_i is termed **eigenspace** associated to λ_i; ts dimension $\nu_i \coloneqq \nu(\lambda \boldsymbol{I} - \boldsymbol{A})$, i.e. the maximum number of linearly independent eigenvectors associated to λ_i, is called **geometric multiplicity** of λ_i.

The matrix $\boldsymbol{V}_i \coloneqq \begin{pmatrix} \boldsymbol{v}_{i1} & \cdots & \boldsymbol{v}_{i\nu_i} \end{pmatrix} \in \mathbb{R}^{n \times \nu_i}$ made up of ν_i linearly independent eigenvectors $\boldsymbol{v}_{i1}, \ldots, \boldsymbol{v}_{i\nu_i}$ of $\boldsymbol{A}$, associated to the same eigenvalue λ_i, is full rank, hence the square matrix $\boldsymbol{P}_i = \begin{pmatrix} \boldsymbol{V}_i & \boldsymbol{N}_i \end{pmatrix}$, where the matrix $\boldsymbol{N}_i \coloneqq \begin{pmatrix} \boldsymbol{n}_{i1} & \cdots & \boldsymbol{n}_{in-\nu_i} \end{pmatrix} \in \mathbb{R}^{n \times n-\nu_i}$ is made up of $n - \nu_i$ linearly independent vectors orthogonal to V_{λ_i}, is non singular. If its inverse $\boldsymbol{P}_i^{-1} = \boldsymbol{Q}_i^{\mathsf{T}}$ is written through a similar block structure, that is $\boldsymbol{Q}_i = \begin{pmatrix} \boldsymbol{U}_i & \boldsymbol{M}_i \end{pmatrix}$ with $\boldsymbol{U}_i \in \mathbb{R}^{n \times \nu_i}, \boldsymbol{M}_i \in \mathbb{R}^{n \times n-\nu_i}$, the similarity transformation via $\boldsymbol{P}_i$ gives

$$\boldsymbol{P}_i^{-1} \boldsymbol{A} \boldsymbol{P}_i = \begin{pmatrix} \boldsymbol{U}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{V}_i & \boldsymbol{U}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{N}_i \\ \boldsymbol{M}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{V}_i & \boldsymbol{M}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{N}_i \end{pmatrix} = \begin{pmatrix} \lambda_i \boldsymbol{I}_{\nu_i} & \boldsymbol{U}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{N}_i \\ \boldsymbol{O}_{n-\nu_i, \nu_i} & \boldsymbol{M}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{N}_i \end{pmatrix}$$

because $\boldsymbol{U}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{V}_i = \boldsymbol{U}_i^{\mathsf{T}} \lambda_i \boldsymbol{I}_n \boldsymbol{V}_i = \lambda_i \boldsymbol{U}_i \boldsymbol{V}_i$ and $\boldsymbol{M}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{V}_i = \boldsymbol{M}_i^{\mathsf{T}} \lambda_i \boldsymbol{I}_n \boldsymbol{V}_i = \boldsymbol{M}_i^{\mathsf{T}} \boldsymbol{V}_i$. Thanks to spectrum invariance under similarity transformation, eq. (A.29) is written as

$$\begin{aligned} p(\boldsymbol{A}) &= \det(\lambda \boldsymbol{I} - \boldsymbol{A}) = \det\left(\lambda \boldsymbol{I}_{\nu_i} - \lambda_i \boldsymbol{I}_{\nu_i}\right) \det(\lambda \boldsymbol{I}_{n-\nu_i} - \boldsymbol{M}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{N}_i) = \\ &= \det\left((\lambda - \lambda_i) \boldsymbol{I}_{\nu_i}\right) \det(\lambda \boldsymbol{I}_{n-\nu_i} - \boldsymbol{M}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{N}_i) = (\lambda - \lambda_i)^{\nu_i} p(\boldsymbol{M}_i^{\mathsf{T}} \boldsymbol{A} \boldsymbol{N}_i) \end{aligned}$$

using eq. (A.22). This expansion shows that λ_i appears at least ν_i times in $p(\boldsymbol{A})$, proving the **multiplicity inequality** $\mu_i \geq \nu_i$. If $\mu_i = \nu_i$, the eigenvalue λ_i is called semi-simple.

Diagonalisation

There are as many eigenspaces as distinct eigenvalues. Let $\boldsymbol{v} = \alpha_1 \boldsymbol{v}_1 + \alpha_2 \boldsymbol{v}_2$ be a linear combination of two eigenvectors $\boldsymbol{v}_1 \in V_{\lambda_1}, \boldsymbol{v}_2 \in V_{\lambda_2}$ associated to two distinct eigenvalues λ_1, λ_2; if $\boldsymbol{v} = \boldsymbol{0}$, then $(\boldsymbol{A} - \lambda_1 \boldsymbol{I})\boldsymbol{v} = \alpha_2(\lambda_2 - \lambda_1)\boldsymbol{v}_2 = \boldsymbol{0}$, implying $\alpha_2 = 0$, as $\lambda_1 \neq \lambda_2$; consequently $\boldsymbol{v} = \alpha_1 \boldsymbol{v}_1 = \boldsymbol{0}$, implying also $\alpha_1 = 0$: $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are linearly independent. Let $\boldsymbol{v} = \alpha_1 \boldsymbol{v}_1 + \ldots + \alpha_p \boldsymbol{v}_p + \alpha_{p+1} \boldsymbol{v}_{p+1} = \boldsymbol{0}$ be a linear combination of p linearly independent eigenvectors $\boldsymbol{v}_1 \in V_{\lambda_1}, \ldots, \boldsymbol{v}_p \in V_{\lambda_p}$ associated to p distinct eigenvalues $\lambda_1, \ldots, \lambda_p$, and the eigenvector $\boldsymbol{v}_{p+1} \in V_{\lambda_{p+1}}$ associated to another eigenvalue λ_{p+1}; if $\boldsymbol{v} = \boldsymbol{0}$, then $(\boldsymbol{A} - \lambda_{p+1} \boldsymbol{I})\boldsymbol{v} = \alpha_1(\lambda_1 - \lambda_{p+1})\boldsymbol{v}_1 + \ldots + \alpha_p(\lambda_p - \lambda_{p+1})\boldsymbol{v}_p = \boldsymbol{0}$, implying $\alpha_1, \ldots, \alpha_p = 0$, as $\lambda_i \neq \lambda_{p+1}, \forall i = 1, \ldots, p$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_p$ are linearly independent; consequently $\boldsymbol{v} = \alpha_{p+1} \boldsymbol{v}_{p+1} = \boldsymbol{0}$, implying also $\alpha_{p+1} = 0$. This proves by induction that eigenvectors associated to distinct eigenvalues are linearly independent.

A matrix with r distinct eigenvalues has at least r linearly independent eigenvectors. Besides, there exist ν_i linearly independent eigenvectors per *each* eigenvalue λ_i. Since $\mu_1 + \ldots + \mu_r = n$, if $\nu_i = \mu_i$, $\forall i = 1, \ldots, r$, then there are exactly n linearly independent eigenvectors. The n eigenequations $\boldsymbol{A} \boldsymbol{v}_j = \lambda_j \boldsymbol{v}_j$ can be written in matrix form

$$\boldsymbol{A} \boldsymbol{V} = \boldsymbol{V} \boldsymbol{\Lambda} \tag{A.31}$$

where $\boldsymbol{V} \coloneqq \begin{pmatrix} \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_n \end{pmatrix}$ and $\boldsymbol{\Lambda} \coloneqq \mathbf{diag}\left(\lambda_1, \ldots, \lambda_n\right)$. Because $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ are linearly independent, $\boldsymbol{V}$, termed **modal matrix**, is full rank and thus non singular: premultiplying eq. (A.31) by $\boldsymbol{V}^{-1}$ gives

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1} \tag{A.32}$$

therefore $\boldsymbol{A}$ is similar to a diagonal matrix via $\boldsymbol{V}$.

Symmetric Matrices

If a symmetric matrix $\boldsymbol{A}$ has the conjugate eigenpairs $(\lambda, \boldsymbol{v})$ and $(\lambda^\star, \boldsymbol{v}^\star)$, then

$$\lambda \left\|\boldsymbol{v}\right\|^2 = \lambda \boldsymbol{v}^{\mathsf{H}}\boldsymbol{v} = \boldsymbol{v}^{\mathsf{H}}\boldsymbol{A}\boldsymbol{v} = \boldsymbol{v}^{\mathsf{H}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{v} = \boldsymbol{v}^{\mathsf{H}}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v} = (\boldsymbol{A}\boldsymbol{v})^{\mathsf{H}}\boldsymbol{v} = \lambda^\star \boldsymbol{v}^{\mathsf{H}}\boldsymbol{v} = \lambda^\star \left\|\boldsymbol{v}\right\|^2$$

implies $\lambda^\star = \lambda, \Leftrightarrow \lambda \in \mathbb{R}$: eigenvalues of symmetric matrices are *all* real. If $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ are two eigenvectors associated to a couple of distinct eigenvalues λ_i and λ_j, then

$$\lambda_i \boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{v}_j = (\boldsymbol{A}\boldsymbol{v}_i)^{\mathsf{T}}\boldsymbol{v}_j = \boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{v}_j = \boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v}_j = \boldsymbol{v}_i^{\mathsf{T}}(\lambda_j \boldsymbol{v}_j) = \lambda_j \boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{v}_j$$

implies $\boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{v}_j = 0$ if $i \neq j$: eigenspaces of symmetric matrices are *mutually* orthogonal.

There are ν_i orthogonal eigenvectors associated to the eigenvalue λ_i. The matrix $\boldsymbol{P}_i = \begin{pmatrix} \boldsymbol{U}_i & \boldsymbol{N}_i \end{pmatrix}$, where $\boldsymbol{U}_i \coloneqq \begin{pmatrix} \boldsymbol{u}_{i1} & \cdots & \boldsymbol{u}_{i\nu_i} \end{pmatrix} \in \mathbb{R}^{n \times \nu_i}$ is made up of ν_i orthonormal eigenvectors and $\boldsymbol{N}_i \coloneqq \begin{pmatrix} \boldsymbol{n}_{i1} & \cdots & \boldsymbol{n}_{in-\nu_i} \end{pmatrix} \in \mathbb{R}^{n \times n-\nu_i}$ is made up of $n - \nu_i$ orthonormal vectors orthogonal to V_{λ_i}, is orthogonal. The similarity transformation of $\boldsymbol{A}$ via $\boldsymbol{P}_i$ gives

$$\boldsymbol{P}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{P}_i = \begin{pmatrix} \boldsymbol{U}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{U}_i & \boldsymbol{U}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i \\ \boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{U}_i & \boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i \end{pmatrix} = \begin{pmatrix} \lambda_i \boldsymbol{I}_{\nu_i} & \boldsymbol{O}_{\nu_i, n-\nu_i} \\ \boldsymbol{O}_{n-\nu_i, \nu_i} & \boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i \end{pmatrix}$$

because $\boldsymbol{U}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{U}_i = \boldsymbol{U}_i^{\mathsf{T}}\lambda_i \boldsymbol{I}_n \boldsymbol{U}_i = \lambda_i \boldsymbol{U}_i^{\mathsf{T}}\boldsymbol{U}_i$, $\boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{U}_i = \boldsymbol{N}_i^{\mathsf{T}}\lambda_i \boldsymbol{I}_n \boldsymbol{U}_i = \lambda_i \boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{U}_i$ and $\boldsymbol{U}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i = \boldsymbol{U}_i^{\mathsf{T}}\boldsymbol{I}_n \lambda_i \boldsymbol{N}_i = \lambda_i \boldsymbol{U}_i^{\mathsf{T}}\boldsymbol{N}_i$. Again, eq. (A.29) can be written as

$$\begin{aligned} p(\boldsymbol{A}) = \det\left(\lambda \boldsymbol{I} - \boldsymbol{A}\right) &= \det\left(\lambda \boldsymbol{I}_{\nu_i} - \lambda_i \boldsymbol{I}_{\nu_i}\right) \det(\lambda \boldsymbol{I}_{n-\nu_i} - \boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i) = \\ &= \det\left((\lambda - \lambda_i)\boldsymbol{I}_{\nu_i}\right) \det(\lambda \boldsymbol{I}_{n-\nu_i} - \boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i) = (\lambda - \lambda_i)^{\nu_i} p(\boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i) \end{aligned}$$

using eq. (A.22). This expansion shows that $\mu_i > \nu_i$ could occur only if λ_i were a root of $p(\boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i)$ too, i.e. if there were at least one non zero vector $\boldsymbol{z} \in \mathbb{R}^{n-\nu_i}$ such that $\left(\boldsymbol{N}_i^{\mathsf{T}}\boldsymbol{A}\boldsymbol{N}_i - \lambda_i \boldsymbol{I}_{n-\nu_i}\right)\boldsymbol{z} = \boldsymbol{N}_i^{\mathsf{T}}(\boldsymbol{A} - \lambda_i \boldsymbol{I}_n)\boldsymbol{N}_i \boldsymbol{z} = \boldsymbol{0}$. Since $\mathcal{R}(\boldsymbol{N}_i) = \mathcal{N}^\perp(\boldsymbol{A} - \lambda_i \boldsymbol{I})$ and $\rho(\boldsymbol{A} - \lambda_i \boldsymbol{I}) = n - \nu_i$, then $\mathcal{R}(\boldsymbol{N}_i^{\mathsf{T}}(\boldsymbol{A} - \lambda_i \boldsymbol{I})\boldsymbol{N}_i) = \mathcal{R}(\boldsymbol{N}_i^{\mathsf{T}}(\boldsymbol{A} - \lambda_i \boldsymbol{I})) = \mathcal{R}(\boldsymbol{N}_i^{\mathsf{T}}) = \mathbb{R}^{n-\nu_i}$, so the eigenequation is satisfied by $\boldsymbol{z} = \boldsymbol{0}$ only, and consequently $\mu_i = \nu_i, \forall i = 1, \ldots, r$.

Symmetric matrices have only semi-simple eigenvalues, thus they always admit **diagonalisation**; moreover, the matrix $\boldsymbol{U} \coloneqq \begin{pmatrix} \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_n \end{pmatrix}$ made up of n normalised eigenvectors, is orthogonal: using it into eq. (A.31) gives

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathsf{T}} \tag{A.33}$$

therefore a symmetric matrix is similar to a diagonal matrix via an orthogonal matrix. Performing the product block-wise gives

$$\boldsymbol{A} = \sum_{j=1}^{n} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^{\mathsf{T}}$$

hence symmetric matrices can be decomposed into the sum of rank one matrices (**dyads**).

A.2.5 Orthogonal Matrices

If vectors are transformed by an orthogonal matrix $\boldsymbol{Q}$, applying the orthogonality condition $\boldsymbol{Q}^{\mathsf{T}}\boldsymbol{Q} = \boldsymbol{I}$ to the dot product and the norm give respectively

$$(\boldsymbol{Q}\boldsymbol{x})^{\mathsf{T}}(\boldsymbol{Q}\boldsymbol{y}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{Q}^{\mathsf{T}}\boldsymbol{Q}\boldsymbol{y} = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{y},\ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n \tag{A.34}$$

that is dot product invariance under product by orthogonal matrices, and

$$\|(\boldsymbol{Q}\boldsymbol{x})\|^2 = (\boldsymbol{Q}\boldsymbol{x})^{\mathsf{T}}(\boldsymbol{Q}\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{x} = \|\boldsymbol{x}\|^2,\ \forall \boldsymbol{x} \in \mathbb{R}^n \tag{A.35}$$

that is norm invariance under product by orthogonal matrices.

The set of all orthogonal matrices of order n is a group called **orthogonal group** of degree n and is denoted with $\mathrm{O}(n)$. Applying eq. (A.24) to $\boldsymbol{Q}^{\mathsf{T}}\boldsymbol{Q} = \boldsymbol{I}$ returns $1 = \det \boldsymbol{I} = \det \boldsymbol{Q}^{\mathsf{T}}\boldsymbol{Q} = \det \boldsymbol{Q}^{\mathsf{T}} \det \boldsymbol{Q} = (\det \boldsymbol{Q})^2 \Leftrightarrow \det \boldsymbol{Q} = \pm 1$, thus orthogonal matrices have unitary determinant, in general. Writing eq. (A.25) for $\boldsymbol{Q}$ yields $\mathbf{adj}\, \boldsymbol{Q} = \pm \boldsymbol{Q}^{\mathsf{T}}$. If $\det \boldsymbol{Q} = +1$, $\boldsymbol{Q}$ is said **special orthogonal**; the set of special orthogonal matrices is also a group called **special orthogonal group** of degree n and is denoted with $\mathrm{SO}(n)$.

Let $(\lambda, \boldsymbol{v})$ be an eigenpair of an orthogonal matrix $\boldsymbol{Q}$: since

$$\|\boldsymbol{v}\|^2 = \boldsymbol{v}^{\mathsf{H}}\boldsymbol{v} = \boldsymbol{v}^{\mathsf{H}}\boldsymbol{Q}^{\mathsf{H}}\boldsymbol{Q}\boldsymbol{v} = (\boldsymbol{Q}\boldsymbol{v})^{\mathsf{H}}(\boldsymbol{Q}\boldsymbol{v}) = (\lambda\boldsymbol{v})^{\mathsf{H}}(\lambda\boldsymbol{v}) = \lambda^{\star}\lambda\boldsymbol{v}^{\mathsf{H}}\boldsymbol{v} = |\lambda|^2 \|\boldsymbol{v}\|^2$$

then $|\lambda| = 1$, that is the eigenvalues of orthogonal matrices lie on the unitary circle.

A.3 Matrices and Vectors in Three Dimensions

A.3.1 Cross Product Matrix

Definition

Using column vectors to write the vector product in full

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \times \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} x_2 y_3 - x_3 y_2 \\ x_3 y_1 - x_1 y_3 \\ x_1 y_2 - x_2 y_1 \end{pmatrix}$$

suggests to define the 3×3 **skew symmetric matrix** $\boldsymbol{S}(\boldsymbol{x})$

$$\boldsymbol{S}(\boldsymbol{x}) \coloneqq \begin{pmatrix} 0 & -x_3 & +x_2 \\ +x_3 & 0 & -x_1 \\ -x_2 & +x_1 & 0 \end{pmatrix}$$

in order to express the cross product as $\boldsymbol{x} \times \boldsymbol{y} = \boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y}$, by means a sheer matrix product, which is possible indeed for the cross product is a linear operator; $\boldsymbol{S}(\boldsymbol{x})$ is also called **cross product matrix**. As $\boldsymbol{S}^{\mathsf{T}}(\boldsymbol{x}) = -\boldsymbol{S}(\boldsymbol{x}) = \boldsymbol{S}(-\boldsymbol{x})$, transposing the matrix corresponds to changing the sign of the associated vector.

Properties

All the cross product properties must be mirrored also by the skew symmetric matrix product operator. Applying it to a linear combination of vectors gives

$$\boldsymbol{S}(\lambda\boldsymbol{x} + \mu\boldsymbol{y}) = \lambda\boldsymbol{S}(\boldsymbol{x}) + \mu\boldsymbol{S}(\boldsymbol{y})$$

which reflects the linearity property in eq. (A.7). Swapping the role of the vectors gives

$$\boldsymbol{S}(\boldsymbol{y})\,\boldsymbol{x} = -\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y}$$

which reflects the anti-commutativity property in eq. (A.8). Multiplying the skew symmetric by the vector it is associated to gives

$$\boldsymbol{S}(\lambda\boldsymbol{x})\,\boldsymbol{x} = \lambda\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{x} = \boldsymbol{0}$$

which reflects the linear dependence property in eq. (A.9). The dot product between $\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y}$ and another vector $\boldsymbol{z}$ gives

$$\boldsymbol{z}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y} = \boldsymbol{y}^{\mathsf{T}}(\boldsymbol{S}^{\mathsf{T}}(\boldsymbol{x})\,\boldsymbol{z}) = \boldsymbol{y}^{\mathsf{T}}\left(-\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{z}\right) = \boldsymbol{y}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{z})\,\boldsymbol{x}$$

which is equivalent to the triple product property in eq. (A.10). The dot product between $\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y}$ and $\boldsymbol{y}$ itself gives

$$\boldsymbol{y}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y} = \boldsymbol{y}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{y})\,\boldsymbol{x} = 0$$

which reflects the orthogonality property in eq. (A.11). The product between two skew symmetric matrices has the particular structure

$$\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{S}(\boldsymbol{y}) = \begin{pmatrix} -x_2y_2 - x_3y_3 & x_2y_1 & x_3y_1 \\ x_1y_2 & -x_3y_3 - x_1y_1 & x_3y_2 \\ x_1y_3 & x_2y_3 & -x_1y_1 - x_2y_2 \end{pmatrix} = \boldsymbol{y}\boldsymbol{x}^{\mathsf{T}} - (\boldsymbol{x}^{\mathsf{T}}\boldsymbol{y})\boldsymbol{I} \quad \text{(A.36)}$$

that can be used to compute the square of a skew symmetric matrix

$$\boldsymbol{S}^2(\boldsymbol{x}) = \boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{S}(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{x}^{\mathsf{T}} - (\boldsymbol{x}^{\mathsf{T}}\boldsymbol{x})\boldsymbol{I} = \boldsymbol{x}\boldsymbol{x}^{\mathsf{T}} - \|\boldsymbol{x}\|^2\,\boldsymbol{I} \quad \text{(A.37)}$$

which is actually a symmetric matrix. Multiplying the double matrix $\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{S}(\boldsymbol{y})$ by another vector $\boldsymbol{z}$ and exploiting eq. (A.36) gives

$$\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{S}(\boldsymbol{y})\,\boldsymbol{z} = \left(\boldsymbol{y}\boldsymbol{x}^{\mathsf{T}} - (\boldsymbol{x}^{\mathsf{T}}\boldsymbol{y})\boldsymbol{I}\right)\boldsymbol{z} = \boldsymbol{y}\boldsymbol{x}^{\mathsf{T}}\boldsymbol{z} - (\boldsymbol{x}^{\mathsf{T}}\boldsymbol{y})\boldsymbol{z} = (\boldsymbol{z}^{\mathsf{T}}\boldsymbol{x})\boldsymbol{y} - (\boldsymbol{x}^{\mathsf{T}}\boldsymbol{y})\boldsymbol{z}$$

which reflects the triple product property in eq. (A.12). Substituting eq. (A.37) into the squared norm of $\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y}$ gives

$$\begin{aligned} \|\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y}\|^2 = (\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y})^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y} = \boldsymbol{y}^{\mathsf{T}}\boldsymbol{S}^{\mathsf{T}}(\boldsymbol{x})\,\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y} = -\boldsymbol{y}^{\mathsf{T}}\boldsymbol{S}^2(\boldsymbol{x})\,\boldsymbol{y} = \\ = \boldsymbol{y}^{\mathsf{T}}\left((\boldsymbol{x}^{\mathsf{T}}\boldsymbol{x})\boldsymbol{I} - \boldsymbol{x}\boldsymbol{x}^{\mathsf{T}}\right)\boldsymbol{y} = (\boldsymbol{x}^{\mathsf{T}}\boldsymbol{x})(\boldsymbol{y}^{\mathsf{T}}\boldsymbol{y}) - (\boldsymbol{x}^{\mathsf{T}}\boldsymbol{y})(\boldsymbol{x}^{\mathsf{T}}\boldsymbol{y}) \end{aligned}$$

which reflects the norm property in eq. (A.13).

Features

Since $\mathbf{adj}\,(\boldsymbol{S}(\boldsymbol{x})) = \boldsymbol{x}\boldsymbol{x}^{\mathsf{T}}$, as verifiable from the definition, then $\det(\boldsymbol{S}(\boldsymbol{x})) = 0,\ \forall \boldsymbol{x} \in \mathbb{R}^3$ because $\det(\boldsymbol{S}(\boldsymbol{x}))\,\boldsymbol{I} = \boldsymbol{S}(\boldsymbol{x})\,\mathbf{adj}\,(\boldsymbol{S}(\boldsymbol{x})) = \boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{x}\boldsymbol{x}^{\mathsf{T}} = \mathbf{0}\boldsymbol{x}^{\mathsf{T}} = \boldsymbol{O}$.

$\boldsymbol{S}(\boldsymbol{x})$ has a zero eigenvalue associated to the eigenvector $\boldsymbol{x}$, because $\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{x} = \mathbf{0} = 0\boldsymbol{x}$. Moreover, its **characteristic polynomial** is

$$\det(\lambda\boldsymbol{I} - \boldsymbol{S}(\boldsymbol{x})) = \lambda^3 - x_1x_2x_3 + x_1x_2x_3 + x_2^2\lambda + x_3^2\lambda + x_1^2\lambda = \lambda^3 + \|\boldsymbol{x}\|^2\,\lambda = \lambda(\lambda^2 + \|\boldsymbol{x}\|^2)$$

therefore $\boldsymbol{S}(\boldsymbol{x})$ has also a couple conjugate imaginary eigenvalues $\pm i\,\|\boldsymbol{x}\| \in \mathbb{C}$, associated to the couple of complex conjugate eigenvectors $\boldsymbol{y} \mp i\boldsymbol{z} \in \mathbb{C}^3$, where $\boldsymbol{y}, \boldsymbol{z} \in \mathbb{R}^3$ are any vectors such that $\|\boldsymbol{y}\| = \|\boldsymbol{z}\|$, $\boldsymbol{x}\cdot\boldsymbol{y} = \boldsymbol{y}\cdot\boldsymbol{z} = \boldsymbol{z}\cdot\boldsymbol{x} = 0$ and $(\boldsymbol{x}\times\boldsymbol{y})\cdot\boldsymbol{z} > 0$, because $\boldsymbol{S}(\boldsymbol{x})\,(\boldsymbol{y} \mp i\boldsymbol{z}) = \|\boldsymbol{x}\|\,\boldsymbol{z} \pm i\,\|\boldsymbol{x}\|\,\boldsymbol{y} = \pm i\,\|\boldsymbol{x}\|\,(\boldsymbol{y} \mp i\boldsymbol{z})$.

A.3.2 Square Matrices of order 3

Matrix row vectors

A matrix $\boldsymbol{M} \in \mathbb{R}^{3\times3}$ can be written as

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{a}^{\mathsf{T}} \\ \boldsymbol{b}^{\mathsf{T}} \\ \boldsymbol{c}^{\mathsf{T}} \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix}$$

putting in evidence its rows $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} \in \mathbb{R}^{3\times1}$ as separate vectors. In this way, vector algebra may be used to give alternative expressions of some matrix quantities: its adjugate

$$\mathbf{adj}\,(\boldsymbol{M}) = \begin{pmatrix} \boldsymbol{S}(\boldsymbol{b})\,\boldsymbol{c} & \boldsymbol{S}(\boldsymbol{c})\,\boldsymbol{a} & \boldsymbol{S}(\boldsymbol{a})\,\boldsymbol{b} \end{pmatrix} \tag{A.38}$$

has the cross products of all rows cyclic permutations as columns, thus its determinant

$$\det(\boldsymbol{M}) = \boldsymbol{a}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{b})\,\boldsymbol{c} = \boldsymbol{b}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{c})\,\boldsymbol{a} = \boldsymbol{c}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{a})\,\boldsymbol{b} \tag{A.39}$$

is the mixed product of the three rows.

Cross Product Transformation

Considering that, in general

$$\boldsymbol{M}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{M}^{\mathsf{T}} = \begin{pmatrix} \boldsymbol{a}^{\mathsf{T}} \\ \boldsymbol{b}^{\mathsf{T}} \\ \boldsymbol{c}^{\mathsf{T}} \end{pmatrix} \boldsymbol{S}(\boldsymbol{x}) \begin{pmatrix} \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} \end{pmatrix} = \begin{pmatrix} \boldsymbol{a}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{a} & \boldsymbol{a}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{b} & \boldsymbol{a}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{c} \\ \boldsymbol{b}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{a} & \boldsymbol{b}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{b} & \boldsymbol{b}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{c} \\ \boldsymbol{c}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{a} & \boldsymbol{c}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{b} & \boldsymbol{c}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{c} \end{pmatrix} =$$

$$= \begin{pmatrix} 0 & -\boldsymbol{x}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{a})\,\boldsymbol{b} & +\boldsymbol{x}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{c})\,\boldsymbol{a} \\ +\boldsymbol{x}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{a})\,\boldsymbol{b} & 0 & -\boldsymbol{x}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{b})\,\boldsymbol{c} \\ -\boldsymbol{x}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{c})\,\boldsymbol{a} & +\boldsymbol{x}^{\mathsf{T}}\boldsymbol{S}(\boldsymbol{b})\,\boldsymbol{c} & 0 \end{pmatrix} = \boldsymbol{S}\Big(\mathbf{adj}\,\boldsymbol{M}^{\mathsf{T}}\boldsymbol{x}\Big)$$

thus, when $\boldsymbol{Q} \in \mathrm{O}(3)$, $\mathbf{adj}\,\boldsymbol{Q}^{\mathsf{T}} = \pm\boldsymbol{Q}$ gives

$$\boldsymbol{Q}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{Q}^{\mathsf{T}} = \pm\boldsymbol{S}(\boldsymbol{Q}\boldsymbol{x}) \tag{A.40}$$

or equivalently

$$\boldsymbol{Q}\boldsymbol{S}(\boldsymbol{x}) = \pm\boldsymbol{S}(\boldsymbol{Q}\boldsymbol{x})\,\boldsymbol{Q} \tag{A.41}$$

which can be used to prove the fundamental property

$$\boldsymbol{Q}\boldsymbol{S}(\boldsymbol{x})\,\boldsymbol{y} = \pm\boldsymbol{S}(\boldsymbol{Q}\boldsymbol{x})\,\boldsymbol{Q}\boldsymbol{y} \tag{A.42}$$

of the cross product between vectors transformed by special orthogonal matrices.

Spectrum

Rearranging the terms in the characteristic polynomial of $\boldsymbol{M}$ as

$$\begin{aligned}\det(\lambda\boldsymbol{I}-\boldsymbol{M}) &= (\lambda-a_1)(\lambda-b_2)(\lambda-c_3) - a_3b_1c_2 - a_2b_3c_1 + \\ &- a_3(\lambda-b_2)c_1 - a_2b_1(\lambda-c_3) - (\lambda-a_1)b_3c_1 = \\ &= \lambda^3 - (a_1+b_2+c_3)\lambda^2 + (a_1b_2 - a_2b_1 + b_2c_3 - b_3c_2 + c_3a_1 - c_1a_3)\lambda + \\ &+ a_1b_2c_3 + a_3b_1c_2 + a_2b_3c_1 - a_3b_2c_1 - a_2b_1c_3 - a_1b_3c_2\end{aligned}$$

gives $p(\boldsymbol{M}) = \lambda^3 - \operatorname{tr}\boldsymbol{M}\lambda^2 + \operatorname{tr}(\mathbf{adj}\,\boldsymbol{M})\,\lambda - \det\boldsymbol{M}$. In particular, if $\boldsymbol{Q}\in\mathrm{O}(3)$

$$p(\boldsymbol{Q}) = \lambda^3 - \operatorname{tr}\boldsymbol{Q}\lambda^2 \pm \operatorname{tr}\boldsymbol{Q}\lambda \mp 1 = (\lambda\mp1)\left(\lambda^2 - (\operatorname{tr}\boldsymbol{Q}\mp1)\lambda + 1\right) \tag{A.43}$$

because $\operatorname{tr}\mathbf{adj}\,\boldsymbol{Q} = \pm\operatorname{tr}\boldsymbol{Q}^{\mathsf{T}} = \pm\operatorname{tr}\boldsymbol{Q}$ and $\det\boldsymbol{Q} = \pm1$.

In general, 3×3 matrices have either three *distinct* real eigenvalues or one real eigenvalue and two complex conjugate eigenvalues. As the eigenvalues of orthogonal matrices lie on the unit circle and there are not three distinct real numbers with unit modulus, 3×3 orthogonal matrices have only one real eigenvalue and two complex conjugate eigenvalues. $\boldsymbol{Q}$ has real eigenvalue $\lambda_1 = \pm1$, depending on the sign of the determinant, since $\det\boldsymbol{Q} = \lambda_1\lambda_2\lambda_3 = \lambda_1\lambda_2\lambda_2^\star = \lambda_1|\lambda_2|^2 = \lambda_1$. Let $\boldsymbol{x}$ be the unit eigenvector associated to the eigenvalue λ_1, i.e. $\boldsymbol{Q}\boldsymbol{x} = \pm\boldsymbol{x}$. Let $\boldsymbol{y},\boldsymbol{z}$ be any two unit vectors such that $\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$ are mutually orthogonal and make up a right handed triple; since orthogonal matrices preserve the dot product, then $\boldsymbol{Q}\boldsymbol{y}$ has to be orthogonal to $\boldsymbol{Q}\boldsymbol{x}$ and can be thus expressed as a linear combination of $\boldsymbol{y}$ and $\boldsymbol{z}$. If $\boldsymbol{Q}\boldsymbol{y} = \alpha\boldsymbol{y} + \beta\boldsymbol{z}$, applying eq. (A.42) gives $\boldsymbol{Q}\boldsymbol{z} = \boldsymbol{Q}(\boldsymbol{x}\times\boldsymbol{y}) = \pm\boldsymbol{Q}\boldsymbol{x}\times\boldsymbol{Q}\boldsymbol{y} = \boldsymbol{x}\times(\alpha\boldsymbol{y}+\beta\boldsymbol{z}) = \alpha\boldsymbol{z} - \beta\boldsymbol{y}$, thus

$$\begin{aligned}\boldsymbol{Q}(\boldsymbol{y}\mp i\boldsymbol{z}) &= (\alpha\boldsymbol{y}+\beta\boldsymbol{z}) \mp i(\alpha\boldsymbol{z}-\beta\boldsymbol{y}) = (\alpha\pm i\beta)\boldsymbol{y} + (\beta\mp i\alpha)\boldsymbol{z} = \\ &= (\alpha\pm i\beta)\boldsymbol{y} \mp i(\alpha\pm i\beta)\boldsymbol{z} = (\alpha\pm i\beta)(\boldsymbol{y}\mp i\boldsymbol{z})\end{aligned}$$

eventually shows that $\lambda_{2,3} = \alpha\pm i\beta$ is a couple of complex conjugate eigenvalues associated to a couple of complex conjugate eigenvectors $\boldsymbol{y}\mp i\boldsymbol{z}$.

A.4 Bases

A.4.1 Basis Vectors

A **basis** of a vector space with dimension n is a list of n linearly independent vectors that belong to that vector space. By virtue of the linear independence, any other element of

the vector space can be represented as a *unique* linear combination of the elements of the **basis**, termed **basis vectors**.

The concept of **basis** concerns whichever abstract vector space, although it becomes definitely more intuitive when applied to Euclidean spaces: a basis $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ of $\mathbb{R}^n$ is an ordered collection of n-dimensional linearly independent vectors, therefore any n-dimensional vector $\boldsymbol{v} \in \mathbb{R}^n$ can be uniquely expressed as

$$\boldsymbol{v} = x_1\boldsymbol{b}_1 + \ldots + x_n\boldsymbol{b}_n = \sum_{i=1}^{n} x_i\boldsymbol{b}_i$$

where the n coefficients $x_1, \ldots, x_n \in \mathbb{R}$ of the linear combination are called the **components** of $\boldsymbol{v}$ with respect to the **basis** B.

In any case, the n components of a vector with respect to a certain basis make up a n-dimensional vector themselves, i.e. $\boldsymbol{x} \coloneqq (x_1, \ldots, x_n) \in \mathbb{R}^n$. The vector of the components is often unambiguously used in place of the *actual* vector, whenever the basis is tacitly implied.

A.4.2 Change of Basis

Any vector space admits *infinite* bases, therefore the same vector can be expressed with respect to different bases, that is to say by means of different suitable linear combinations of their respective basis vectors. For instance, a vector $\boldsymbol{v}$ in a vector space with dimension n may have representation

$$\boldsymbol{v} = x_1\boldsymbol{e}_1 + \ldots + x_n\boldsymbol{e}_n = \sum_{i=1}^{n} x_i\boldsymbol{e}_i \tag{A.44}$$

in the base $E = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n)$, and representation

$$\boldsymbol{v} = y_1\boldsymbol{f}_1 + \ldots + y_n\boldsymbol{f}_n = \sum_{i=1}^{n} y_i\boldsymbol{f}_i \tag{A.45}$$

in the base $F = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_n)$.

On the other hand, the elements of bases must be vectors too, therefore $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_n$ might be expressed with respect to $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$ as

$$\boldsymbol{f}_j = p_{1j}\boldsymbol{e}_1 + \ldots + p_{nj}\boldsymbol{e}_n,\ j = 1, \ldots, n$$

where p_{ij} is the component of the j-th element of the F basis with respect to the i-th element of the E basis. Substituting these expressions into eq. (A.45)

$$\boldsymbol{v} = \sum_{j=1}^{n} y_j\boldsymbol{f}_j = \sum_{j=1}^{n} y_j \sum_{i=1}^{n} p_{ij}\boldsymbol{e}_i = \sum_{i=1}^{n} \left(\sum_{j=1}^{n} p_{ij}y_j \right) \boldsymbol{e}_i$$

and comparing the result with eq. (A.44) gives the relation

$$x_i = \sum_{j=1}^{n} p_{ij} y_j \; i = 1, \ldots, n$$

between the components of $\boldsymbol{v}$ with respect to the F basis and the components of $\boldsymbol{v}$ with respect to the E basis; gathering all these coefficients into a $n \times n$ matrix, this relation can be rewritten succinctly as

$$\boldsymbol{x} = \boldsymbol{P}\boldsymbol{y} \tag{A.46}$$

where $\boldsymbol{P}$ is the **change of basis matrix** between the F and E bases or **transition matrix** from the F basis to the E basis.

Conversely, repeating the same reasoning in reverse, $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$ might be expressed with respect to $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_n$ as

$$\boldsymbol{e}_j = q_{1j}\boldsymbol{f}_1 + \ldots + q_{nj}\boldsymbol{f}_n, \; j = 1, \ldots, n$$

where q_{ij} is the component of the j-th element of the E basis with respect to the i-th element of the F basis; gathering all these coefficients into a $n \times n$ matrix, this relation can be rewritten succinctly as

$$\boldsymbol{y} = \boldsymbol{Q}\boldsymbol{x} \tag{A.47}$$

where $\boldsymbol{Q}$ is the **change of basis matrix** between the E and F bases or **transition matrix** from the E basis to the F basis.

Since $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_n$ are linearly independent by definition, then the vectors made up of their components with respect to $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$ must be linearly independent as well: these vectors are exactly the rows of the change of basis matrix $\boldsymbol{P}$, which is thus full rank and invertible. Multiplying both sides of eq. (A.46) by $\boldsymbol{P}^{-1}$ provides the inverse relation $\boldsymbol{P}^{-1}\boldsymbol{x} = \boldsymbol{y}$, which, matched with eq. (A.47), eventually proves that $\boldsymbol{Q} = \boldsymbol{P}^{-1}$.

A.4.3 Orthonormal Bases

Bases that consist of orthogonal vectors only are said to be **orthogonal**. Among orthogonal bases, those made up of unit vectors only are said to be **orthonormal**. The basis vectors of an orthonormal base $N = (\boldsymbol{n}_1, \ldots, \boldsymbol{n}_n)$ are subject to the orthonormality constraint $\boldsymbol{n}_i \cdot \boldsymbol{n}_j = \delta_{ij}, \; 1 \leq i, j \leq n$.

Let $E = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n)$ and $F = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_n)$ be two orthonormal bases. By virtue of orthonormality, the dot product of $\boldsymbol{v}$ with the elements of E basis

$$\boldsymbol{v} \cdot \boldsymbol{e}_i = \sum_{j=1}^{n} x_j \boldsymbol{e}_j \cdot \boldsymbol{e}_i = x_i$$

gives the components of $\boldsymbol{v}$ with respect to the E basis. If $\boldsymbol{v}$ is already represented in the F basis, performing the dot product as before

$$\boldsymbol{v} \cdot \boldsymbol{e}_i = \sum_{j=1}^{n} y_j \boldsymbol{f}_j \cdot \boldsymbol{e}_i$$

and comparing this result with the one found before gives the relation

$$x_i = \sum_{j=1}^{n} \left(\boldsymbol{f}_j \cdot \boldsymbol{e}_i \right) y_j$$

which can be rewritten, defining the transition matrix from the F basis to the E basis

$$\boldsymbol{P} \coloneqq \begin{pmatrix} \boldsymbol{f}_1 \cdot \boldsymbol{e}_1 & \dots & \boldsymbol{f}_n \cdot \boldsymbol{e}_1 \\ \vdots & \ddots & \vdots \\ \boldsymbol{f}_1 \cdot \boldsymbol{e}_n & \dots & \boldsymbol{f}_n \cdot \boldsymbol{e}_n \end{pmatrix} \tag{A.48}$$

in the compact form $\boldsymbol{x} = \boldsymbol{P}\boldsymbol{y}$.

Swapping the role of the E and F bases leads to

$$y_i = \sum_{j=1}^{n} \left(\boldsymbol{e}_j \cdot \boldsymbol{f}_i \right) x_j$$

which can be rewritten, defining the transition matrix from the E basis to the F basis

$$\boldsymbol{Q} \coloneqq \begin{pmatrix} \boldsymbol{e}_1 \cdot \boldsymbol{f}_1 & \dots & \boldsymbol{e}_n \cdot \boldsymbol{f}_1 \\ \vdots & \ddots & \vdots \\ \boldsymbol{e}_1 \cdot \boldsymbol{f}_n & \dots & \boldsymbol{e}_n \cdot \boldsymbol{f}_n \end{pmatrix} \tag{A.49}$$

in the compact form $\boldsymbol{y} = \boldsymbol{Q}\boldsymbol{x}$.

It is worth noting that the transition matrix from E basis to F basis is nothing but the transition matrix from the F basis to the E basis only with rows and columns exchanged, therefore $\boldsymbol{Q} = \boldsymbol{P}^{\mathsf{T}}$. On the other hand, as shown before, the transition matrix from the E basis to the F basis is the inverse of the transition matrix from the F basis to the E basis, therefore also $\boldsymbol{Q} = \boldsymbol{P}^{-1}$. These two considerations together imply that $\boldsymbol{P}^{-1} = \boldsymbol{P}^{\mathsf{T}}$, that is to say transition matrices between orthonormal bases are orthogonal. The orthogonality of transition matrices is a very important property and justifies the use of orthonormal bases: for instance, observing that

$$\|\boldsymbol{x}\|^2 = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{x} = (\boldsymbol{P}\boldsymbol{y})^{\mathsf{T}} (\boldsymbol{P}\boldsymbol{y}) = \boldsymbol{y}^{\mathsf{T}}\boldsymbol{P}^{\mathsf{T}}\boldsymbol{P}\boldsymbol{y} = \boldsymbol{y}^{\mathsf{T}}\boldsymbol{y} = \|\boldsymbol{y}\|^2$$

it ensures norm invariance under change of representation between orthonormal bases.

A.5 Euclidean Geometry

A.5.1 Planes

Planes in three dimensions are hyperplanes in $\mathbb{R}^3$, therefore they are affine subsets of $\mathbb{R}^3$ with dimension 2. Two alternative analytical descriptions of a 3D plane, namely the cartesian equation and the parametric equation, may be given.

Cartesian Equation

A plane in 3D space may be defined as the locus of points such that the vector connecting them to a fixed point is orthogonal to a certain direction. Such a description gives rise to the plane cartesian equation

$$(\boldsymbol{r} - \boldsymbol{r}_0) \cdot \boldsymbol{n} = 0 \tag{A.50}$$

where $\boldsymbol{r}_0 \in \mathbb{R}^3$ is any point that belongs to the plane and $\boldsymbol{n} \in \mathbb{R}^3$ is a non zero vector perpendicular to the plane, termed normal vector. The vector $\boldsymbol{n}$ specifies only the spatial inclination of the plane, therefore its magnitude is actually not relevant: for this reason, without loss of generality, a unit vector is usually chosen. Labelling each component of the normal vector as $\boldsymbol{n} \coloneqq (a, b, c)$ the plane equation can be rewritten in the more familiar form

$$ax + by + cz + d = 0$$

where $d = -\boldsymbol{r}_0 \cdot \boldsymbol{n} = -ax_0 - by_0 - cz_0$.

The **orthogonal projection** of the origin onto the plane is the point $\boldsymbol{p} \in \mathbb{R}^3$ on the plane closest to the origin. The vector joining the origin with its orthogonal projection on the plane must be orthogonal to the plane, or, equivalently parallel to the normal vector, i.e. $\exists \lambda \in \mathbb{R} \colon \boldsymbol{p} = \lambda \boldsymbol{n}$. Moreover, the point $\boldsymbol{p}$ must also lie on the plane, i.e. $\boldsymbol{p} \cdot \boldsymbol{n} + d = 0$. Putting together such conditions gives $\boldsymbol{p} \cdot \boldsymbol{n} + d = \lambda \boldsymbol{n} \cdot \boldsymbol{n} + d = \lambda \left\| \boldsymbol{n} \right\|^2 + d = 0$, and, if $\boldsymbol{n}$ is a unit vector, then $\lambda = -d$ and $\boldsymbol{p} = -d\boldsymbol{n}$. The length $\left\| \boldsymbol{p} \right\| = |\lambda| = d$ of this vector is the distance of the plane from the origin.

Parametric Equation

A plane in 3D space may be defined as the locus of points given by the sum between a fixed point and the linear combination of two linearly independent vectors orthogonal to a certain constant orientation. Such a description gives rise to the plane parametric equation

$$\boldsymbol{r} = \boldsymbol{r}_0 + s\boldsymbol{u} + t\boldsymbol{v}$$

where $\boldsymbol{r}_0 \in \mathbb{R}^3$ is any point that belongs to the plane, $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^3$ are two non zero and non aligned vectors parallel to the plane and $s, t \in \mathbb{R}$ are two free scalar parameters: only two free parameters are required because the plane is an affine set of dimension 2. The vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ define only the spatial inclination of the plane, therefore their magnitude and their relative orientation are actually not relevant: for this reason, without loss of generality, two orthonormal vectors are usually chosen.

Relationship between Planes Representations

Different representations of the same plane have to be equivalent, of course.

The conversion from a parametric representation to a cartesian representation may be carried out by observing that the normal vector has to be orthogonal to all vectors lying on the plane. A unit vector perpendicular to the plane might be given by

$$\boldsymbol{n} = \pm \frac{\boldsymbol{u} \times \boldsymbol{v}}{\left\| \boldsymbol{u} \times \boldsymbol{v} \right\|}$$

where the double sign indicates that both choices are equally valid. The same point $\boldsymbol{r}_0$ might be used as reference or any other point might be selected by fixing some values for the couple of parameters (s, t).

The conversion from a cartesian representation to a parametric representation may be carried out by observing that all vectors lying on the plane have to be orthogonal the normal vector. Two vectors parallel to the plane might be given by $\boldsymbol{u} = \boldsymbol{r}_1 - \boldsymbol{r}_0$ and $\boldsymbol{v} = \boldsymbol{r}_2 - \boldsymbol{r}_0$, where $\boldsymbol{r}_1, \boldsymbol{r}_2$ are two points of the plane such that $(\boldsymbol{r}_1 - \boldsymbol{r}_0) \times (\boldsymbol{r}_2 - \boldsymbol{r}_0) \neq \boldsymbol{0}$. If two orthonormal vectors are desired, they might be given instead by

$$\boldsymbol{u} = \frac{\overline{\boldsymbol{r}} - \boldsymbol{r}_0}{\|\overline{\boldsymbol{r}} - \boldsymbol{r}_0\|}, \; \boldsymbol{v} = \frac{\boldsymbol{n} \times \boldsymbol{u}}{\|\boldsymbol{n}\|}$$

where $\overline{\boldsymbol{r}}$ is a point of the plane.

A.5.2 Lines

Lines in three dimension are affine subsets of $\mathbb{R}^3$ with dimension 1. Two alternative analytical descriptions of a 3D line, namely the cartesian equation and the parametric equation, may be given.

Cartesian Equation

A line in 3D space may be defined as the locus of points given by the intersection between two non parallel planes; such a description gives rise to the line **cartesian equation**, that is the system of two planes cartesian equations

$$\begin{cases} (\boldsymbol{r} - \boldsymbol{r}_1) \cdot \boldsymbol{n}_1 = a_1 x + b_1 y + c_1 z + d_1 = 0 \\ (\boldsymbol{r} - \boldsymbol{r}_2) \cdot \boldsymbol{n}_2 = a_2 x + b_2 y + c_2 z + d_2 = 0 \end{cases}$$

with the straightforward condition $\boldsymbol{n}_1 \times \boldsymbol{n}_2 \neq \boldsymbol{0}$, otherwise the intersection would be either empty, whether $d_1 \neq d_2$ (parallel planes) or the entire plane, whether $d_1 = d_2$ (coincident planes).

Parametric Equation

A line in 3D space may be defined as the locus of points given by the sum between a fixed point and a vector with constant orientation. Such a description gives rise to the line parametric equation

$$\boldsymbol{r} = \boldsymbol{r}_0 + t\boldsymbol{v}, \; t \in \mathbb{R}$$

where $\boldsymbol{r}_0 \in \mathbb{R}^3$ is any point that belongs to the line, $\boldsymbol{v} \in \mathbb{R}^3$ is any vector parallel to the line, and $t \in \mathbb{R}$ is a free scalar parameter: only one free parameter is required because the line is an affine set of dimension 1. The vector $\boldsymbol{v}$ specifies only the spatial direction of the line, therefore its magnitude is actually not relevant: for this reason, without loss of generality, a unit vector is usually chosen.

Relationship between Lines Representations

Different representations of the same line have to be equivalent, of course.

The conversion from a cartesian representation to a parametric representation may be carried out by observing that the line has to lie on both planes thus it has to be orthogonal to both their normal vectors. A unit vector $\boldsymbol{v}$ parallel to the line might be given by

$$\boldsymbol{v} = \pm \frac{\boldsymbol{n}_1 \times \boldsymbol{n}_2}{\|\boldsymbol{n}_1 \times \boldsymbol{n}_2\|}$$

where the double sign indicates that both choices are equally valid. The solution of the system of equations yields infinite solutions depending on a certain free variable, thus a point $\boldsymbol{r}_0$ might be found singling out a specific value for this variable.

The conversion from a parametric representation to a cartesian representation may be carried out by observing that any two distinct planes fashioning the system of equations might be selected within the sheaf of planes through the line.

Appendix B

Rigid Body Kinematics

B.1 Rigid Transformations

B.1.1 Proper Rigid Transformations

A transformation of 3D space is called **rigid transformation** if it does not change the relative distance between any points subjected to it; for this reason, it is also called isometry, from the ancient greek words ἴσος (equal) and μέτρον (measure). Rigid transformations are strictly related to the displacement of rigid bodies, i.e. bodies that move without modifying their shape and size.

Formally, **reflections** are rigid transformations but do not represent feasible rigid body displacements, hence the above notion should be slightly amended to reject them. A transformation of 3D space is called **proper rigid transformation** if it does not change either the relative distance or the relative orientation between any points subjected to it; from a mathematical point of view, a (proper) rigid transformation is a vector field $\boldsymbol{f}\colon \mathbb{R}^3 \to \mathbb{R}^3$ such that the properties

$$\|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})\| = \|\boldsymbol{x} - \boldsymbol{y}\| \tag{B.1}$$

$$(\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{0})) \times (\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0})) = \boldsymbol{f}(\boldsymbol{x} \times \boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0}) \tag{B.2}$$

hold $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3$; they account respectively for relative distance and relative orientation invariance. Moreover, applying $\|\boldsymbol{u} - \boldsymbol{v}\|^2 = \|\boldsymbol{u}\|^2 + \|\boldsymbol{v}\|^2 - 2\boldsymbol{u} \cdot \boldsymbol{v}$, $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^3$ gives

$$\begin{gathered}(\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{0})) \cdot (\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0})) = \\ \frac{1}{2}\left(\|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{0})\|^2 + \|\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0})\|^2 - \|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})\|^2\right) = \\ = \frac{1}{2}\left(\|\boldsymbol{x} - \boldsymbol{0}\|^2 + \|\boldsymbol{y} - \boldsymbol{0}\|^2 - \|\boldsymbol{x} - \boldsymbol{y}\|^2\right) = \frac{1}{2}\left(\|\boldsymbol{x}\|^2 + \|\boldsymbol{y}\|^2 - \|\boldsymbol{x} - \boldsymbol{y}\|^2\right) = \boldsymbol{x} \cdot \boldsymbol{y}\end{gathered}$$

therefore the additional property

$$(\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{0})) \cdot (\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0})) = \boldsymbol{x} \cdot \boldsymbol{y} \tag{B.3}$$

is inferred from the one in eq. (B.1).

Considering the auxiliary function $\boldsymbol{g}\colon \mathbb{R}^3 \to \mathbb{R}^3$, defined as $\boldsymbol{g}(\boldsymbol{x}) \coloneqq \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{0})$, which has the origin as fixed point, i.e. $\boldsymbol{g}(\boldsymbol{0}) = \boldsymbol{0}$, eqs. (B.1), (B.3) and (B.2) can be homogeneously rewritten respectively as

$$\|\boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{g}(\boldsymbol{y})\| = \|\boldsymbol{x} - \boldsymbol{y}\|\,,\ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3 \tag{B.4a}$$

$$\boldsymbol{g}(\boldsymbol{x}) \cdot \boldsymbol{g}(\boldsymbol{y}) = \boldsymbol{x} \cdot \boldsymbol{y},\ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3 \tag{B.4b}$$

$$\boldsymbol{g}(\boldsymbol{x}) \times \boldsymbol{g}(\boldsymbol{y}) = \boldsymbol{g}(\boldsymbol{x} \times \boldsymbol{y})\,,\ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3 \tag{B.4c}$$

because $\boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{g}(\boldsymbol{y}) = (\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{0})) - (\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0})) = \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})$.

B.1.2 Affine Map

Applying eq. (B.3) to the convex combination of two points $\boldsymbol{x}$ and $\boldsymbol{y}$, i.e. $\lambda\boldsymbol{x} + (1-\lambda)\,\boldsymbol{y}$ where $\lambda \in [0, 1]$, and to another point $\boldsymbol{z}$, gives the condition

$$\begin{gathered}
(\boldsymbol{f}(\lambda\boldsymbol{x} + (1-\lambda)\,\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0})) \cdot (\boldsymbol{f}(\boldsymbol{z}) - \boldsymbol{f}(\boldsymbol{0})) = \\
= (\lambda\boldsymbol{x} + (1-\lambda)\,\boldsymbol{y}) \cdot \boldsymbol{z} = \lambda\boldsymbol{x} \cdot \boldsymbol{z} + (1-\lambda)\,\boldsymbol{y} \cdot \boldsymbol{z} = \\
= \lambda(\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{0})) \cdot (\boldsymbol{f}(\boldsymbol{z}) - \boldsymbol{f}(\boldsymbol{0})) + (1-\lambda)\,(\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0})) \cdot (\boldsymbol{f}(\boldsymbol{z}) - \boldsymbol{f}(\boldsymbol{0})) = \\
= (\lambda\boldsymbol{f}(\boldsymbol{x}) - \lambda\boldsymbol{f}(\boldsymbol{0}) + (1-\lambda)\,\boldsymbol{f}(\boldsymbol{y}) - (1-\lambda)\,\boldsymbol{f}(\boldsymbol{0})) \cdot (\boldsymbol{f}(\boldsymbol{z}) - \boldsymbol{f}(\boldsymbol{0})) = \\
= (\lambda\boldsymbol{f}(\boldsymbol{x}) + (1-\lambda)\,\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{f}(\boldsymbol{0})) \cdot (\boldsymbol{f}(\boldsymbol{z}) - \boldsymbol{f}(\boldsymbol{0}))
\end{gathered}$$

which must hold $\forall \boldsymbol{z} \in \mathbb{R}^3$, hence necessarily

$$\boldsymbol{f}(\lambda\boldsymbol{x} + (1-\lambda)\,\boldsymbol{y}) = \lambda\boldsymbol{f}(\boldsymbol{x}) + (1-\lambda)\,\boldsymbol{f}(\boldsymbol{y})\,,\ \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3,\ \forall \lambda \in [0, 1] \tag{B.5}$$

that is all isometries are affine maps. The converse implication is not true in general: not all affine maps may represent isometries, just those satisfying distance conservation.

Any affine map $\boldsymbol{f}\colon \mathbb{R}^3 \to \mathbb{R}^3$ can be written as $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$, where $\boldsymbol{A} \in \mathbb{R}^{3\times3}$ and $\boldsymbol{b} \in \mathbb{R}^{3\times1}$. Applying eq. (B.1) to $\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$ gives the constraint

$$\begin{aligned}
(\boldsymbol{x} - \boldsymbol{y})^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2 = \|(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}) - (\boldsymbol{A}\boldsymbol{y} + \boldsymbol{b})\|^2 = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}\boldsymbol{y}\|^2 = \\
= \|\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{y})\|^2 = (\boldsymbol{x} - \boldsymbol{y})^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{y})\,, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3
\end{aligned}$$

on $\boldsymbol{A}$ only. The matrix $\boldsymbol{M} \coloneqq \boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}$ of the quadratic form is symmetric positive semi-definite thus it is similar to a diagonal matrix $\boldsymbol{\Lambda} = \mathbf{diag}(\lambda_1, \lambda_2, \lambda_3)$ with non-negative entries, via an orthogonal matrix $\boldsymbol{U}$, i.e. $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathsf{T}}$. Let $\boldsymbol{p} \in \mathbb{R}^3$ be a generic vector and $\boldsymbol{q} \coloneqq \boldsymbol{U}^{\mathsf{T}}\boldsymbol{p}$: the above condition can be expanded as

$$\begin{aligned}
q_1^2 + q_2^2 + q_3^2 = \boldsymbol{q}^{\mathsf{T}}\boldsymbol{q} = \left(\boldsymbol{U}^{\mathsf{T}}\boldsymbol{p}\right)^{\mathsf{T}}\left(\boldsymbol{U}^{\mathsf{T}}\boldsymbol{p}\right) = \boldsymbol{p}^{\mathsf{T}}\boldsymbol{U}\boldsymbol{U}^{\mathsf{T}}\boldsymbol{p} = \boldsymbol{p}^{\mathsf{T}}\boldsymbol{p} = \boldsymbol{p}^{\mathsf{T}}\boldsymbol{M}\boldsymbol{p} = \boldsymbol{p}^{\mathsf{T}}\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathsf{T}}\boldsymbol{p} = \\
= \boldsymbol{q}^{\mathsf{T}}\boldsymbol{\Lambda}\boldsymbol{q} = \lambda_1 q_1^2 + \lambda_2 q_2^2 + \lambda_3 q_3^2, \quad \forall \boldsymbol{p} \in \mathbb{R}^3 \Leftrightarrow \forall \boldsymbol{q} \in \mathbb{R}^3
\end{aligned}$$

which, because $\lambda_1, \lambda_2, \lambda_3 \geq 0$, necessarily implies $\lambda_1 = \lambda_2 = \lambda_3 = 1$, i.e. $\boldsymbol{\Lambda} = \boldsymbol{I}_3$, thus $\boldsymbol{M} = \boldsymbol{A}^{\mathsf{T}}\boldsymbol{A} = \boldsymbol{U}\boldsymbol{I}_3\boldsymbol{U}^{\mathsf{T}} = \boldsymbol{U}\boldsymbol{U}^{\mathsf{T}} = \boldsymbol{I}_3$ and $\boldsymbol{A}$ is orthogonal: for this reason

$$\det \boldsymbol{A} = \pm 1 \tag{B.6}$$

represents a compact condition to characterise isometries.

On the other hand, among affine maps, just those satisfying the orientation conservation may represent proper isometries. Applying eq. (B.2) to $\boldsymbol{A}\boldsymbol{x}+\boldsymbol{b}$ gives the constraint

$$\begin{aligned}\boldsymbol{A}(\boldsymbol{x}\times\boldsymbol{y}) &= (\boldsymbol{A}(\boldsymbol{x}\times\boldsymbol{y})+\boldsymbol{b})-(\boldsymbol{A}\boldsymbol{0}+\boldsymbol{b}) = \\ &= ((\boldsymbol{A}\boldsymbol{x}+\boldsymbol{b})-(\boldsymbol{A}\boldsymbol{0}+\boldsymbol{b}))\times((\boldsymbol{A}\boldsymbol{y}+\boldsymbol{b})-(\boldsymbol{A}\boldsymbol{0}+\boldsymbol{b})) = \boldsymbol{A}\boldsymbol{x}\times\boldsymbol{A}\boldsymbol{y},\ \forall\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^3\end{aligned}$$

on $\boldsymbol{A}$ only. Writing $\boldsymbol{A}\coloneqq\begin{pmatrix}\boldsymbol{a}_1 & \boldsymbol{a}_2 & \boldsymbol{a}_3\end{pmatrix}$, the right hand side can be expanded as

$$\begin{gathered}\boldsymbol{A}\boldsymbol{x}\times\boldsymbol{A}\boldsymbol{y} = (x_1\boldsymbol{a}_1+x_2\boldsymbol{a}_2+x_3\boldsymbol{a}_3)\times(y_1\boldsymbol{a}_1+y_2\boldsymbol{a}_2+y_3\boldsymbol{a}_3) = \\ = (x_1y_2-x_2y_1)(\boldsymbol{a}_1\times\boldsymbol{a}_2)+(x_2y_3-x_3y_2)(\boldsymbol{a}_2\times\boldsymbol{a}_3)+(x_3y_1-x_1y_3)(\boldsymbol{a}_3\times\boldsymbol{a}_1) = \\ = \mathbf{adj}\,\boldsymbol{A}^{\mathsf{T}}(\boldsymbol{x}\times\boldsymbol{y}),\ \forall\boldsymbol{x},\boldsymbol{y}\in\mathbb{R}^3\end{gathered}$$

thanks to eq. (A.38), which implies $\boldsymbol{A} = \mathbf{adj}\,\boldsymbol{A}^{\mathsf{T}}$: applying eqs. (A.25) and (A.24) to $\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A} = \boldsymbol{A}^{\mathsf{T}}\,\mathbf{adj}\,\boldsymbol{A}^{\mathsf{T}}$ gives $(\det(\boldsymbol{A}))^2 = (\det(\boldsymbol{A}))^3 \Leftrightarrow \det(\boldsymbol{A}) = 0\vee 1$ that combined with eq. (B.6) finally yields the condition $\det(\boldsymbol{A}) = 1$ for proper isometries. Conversely, the specular property $\boldsymbol{A}\boldsymbol{x}\times\boldsymbol{A}\boldsymbol{y} = -\boldsymbol{A}(\boldsymbol{x}\times\boldsymbol{y})$ leads to the opposite condition $\det(\boldsymbol{A}) = -1$ for improper isometries.

If $\boldsymbol{A}$ is orthogonal, the affine map is a rigid transformation: this matrix is denoted by $\boldsymbol{R}$, for it represents a **rotation**, when $\det\boldsymbol{R} = +1$, or a **reflection**, when $\det\boldsymbol{R} = -1$. In constrast, no constraint is applied to $\boldsymbol{b}$: this vector is denoted by $\boldsymbol{t}$, for it represents a **translation**. Proper rigid transformations are the only kind of motion rigid bodies may undergo, hence, in the scope of kinematics, they are sometimes called **rigid motions**.

B.1.3 Rigid Displacement

Let $\boldsymbol{R}\in\mathbb{R}^{3\times3}$ and $\boldsymbol{t}\in\mathbb{R}^3$; if

$$\det\boldsymbol{R} = 1 \tag{B.7a}$$

$$\boldsymbol{R}\boldsymbol{R}^{\mathsf{T}} = \boldsymbol{R}^{\mathsf{T}}\boldsymbol{R} = \boldsymbol{I} \tag{B.7b}$$

then the map $\boldsymbol{f}\colon\mathbb{R}^3\to\mathbb{R}^3$ defined as $\boldsymbol{f}(\boldsymbol{r})\coloneqq\boldsymbol{R}\boldsymbol{r}+\boldsymbol{t}$ is a proper rigid transformation. The overall action of the rigid displacement can be split in two step: $\boldsymbol{r}$ is transformed first into $\boldsymbol{r}'\coloneqq\boldsymbol{R}\boldsymbol{r}$ by the rotation and then into $\boldsymbol{r}''\coloneqq\boldsymbol{r}'+\boldsymbol{t} = \boldsymbol{R}\boldsymbol{r}+\boldsymbol{t}$ by the translation; these vectors are made up of the coordinates

$$\boldsymbol{r}\coloneqq\begin{pmatrix}x\\y\\z\end{pmatrix},\ \boldsymbol{r}'\coloneqq\begin{pmatrix}x'\\y'\\z'\end{pmatrix}\ \boldsymbol{r}''\coloneqq\begin{pmatrix}x''\\y''\\z''\end{pmatrix}$$

of three corresponding points with respect to a given coordinate frame $\mathcal{R}$ with origin $\boldsymbol{o}$ and axes $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$. Let $\mathcal{R}'$ be another coordinate frame with origin $\boldsymbol{o}$ and axes $\boldsymbol{i}'$, $\boldsymbol{j}'$, $\boldsymbol{k}'$, such that the coordinates of the rotated point in $\mathcal{R}'$ are the same as the coordinates of

the original point in $\mathcal{R}$, i.e. $x'\boldsymbol{i} + y'\boldsymbol{j} + z'\boldsymbol{k} = x\boldsymbol{i}' + y\boldsymbol{j}' + z\boldsymbol{k}'$. This equation might be equivalently rewritten in matrix form as

$$\begin{pmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\end{pmatrix}\begin{pmatrix}x'\\ y'\\ z'\end{pmatrix} = \begin{pmatrix}\boldsymbol{i}' & \boldsymbol{j}' & \boldsymbol{k}'\end{pmatrix}\begin{pmatrix}x\\ y\\ z\end{pmatrix}$$

or, considering that matrices with orthonormal columns are orthogonal, also as

$$\begin{pmatrix}x'\\ y'\\ z'\end{pmatrix} = \begin{pmatrix}\boldsymbol{i}^{\mathsf{T}}\\ \boldsymbol{j}^{\mathsf{T}}\\ \boldsymbol{k}^{\mathsf{T}}\end{pmatrix}\begin{pmatrix}\boldsymbol{i}' & \boldsymbol{j}' & \boldsymbol{k}'\end{pmatrix}\begin{pmatrix}x\\ y\\ z\end{pmatrix} = \begin{pmatrix}\boldsymbol{i}^{\mathsf{T}}\boldsymbol{i}' & \boldsymbol{i}^{\mathsf{T}}\boldsymbol{j}' & \boldsymbol{i}^{\mathsf{T}}\boldsymbol{k}'\\ \boldsymbol{j}^{\mathsf{T}}\boldsymbol{i}' & \boldsymbol{j}^{\mathsf{T}}\boldsymbol{j}' & \boldsymbol{j}^{\mathsf{T}}\boldsymbol{k}'\\ \boldsymbol{k}^{\mathsf{T}}\boldsymbol{i}' & \boldsymbol{k}^{\mathsf{T}}\boldsymbol{j}' & \boldsymbol{k}^{\mathsf{T}}\boldsymbol{k}'\end{pmatrix}\begin{pmatrix}x\\ y\\ z\end{pmatrix}$$

giving thus the following expression of the rotation matrix

$$\boldsymbol{R} = \begin{pmatrix}\boldsymbol{i}^{\mathsf{T}}\\ \boldsymbol{j}^{\mathsf{T}}\\ \boldsymbol{k}^{\mathsf{T}}\end{pmatrix}\begin{pmatrix}\boldsymbol{i}' & \boldsymbol{j}' & \boldsymbol{k}'\end{pmatrix} = \begin{pmatrix}\boldsymbol{i}^{\mathsf{T}}\boldsymbol{i}' & \boldsymbol{i}^{\mathsf{T}}\boldsymbol{j}' & \boldsymbol{i}^{\mathsf{T}}\boldsymbol{k}'\\ \boldsymbol{j}^{\mathsf{T}}\boldsymbol{i}' & \boldsymbol{j}^{\mathsf{T}}\boldsymbol{j}' & \boldsymbol{j}^{\mathsf{T}}\boldsymbol{k}'\\ \boldsymbol{k}^{\mathsf{T}}\boldsymbol{i}' & \boldsymbol{k}^{\mathsf{T}}\boldsymbol{j}' & \boldsymbol{k}^{\mathsf{T}}\boldsymbol{k}'\end{pmatrix} \tag{B.8}$$

called **direction cosine matrix**, as its entries are the cosines of the angles between frame $\mathcal{R}'$ axes and frame $\mathcal{R}$ axes. Remembering eq. (A.48), the direction cosine matrix is nothing but a particular case of transition matrix between orthonormal bases.

The axes of $\mathcal{R}'$ can be computed from the axes of $\mathcal{R}$, by rearranging eq. (B.8), from $\begin{pmatrix}\boldsymbol{i}' & \boldsymbol{j}' & \boldsymbol{k}'\end{pmatrix} = \begin{pmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\end{pmatrix}\boldsymbol{R}$, or, alternatively from $\begin{pmatrix}\boldsymbol{i}' & \boldsymbol{j}' & \boldsymbol{k}'\end{pmatrix} = \bar{\boldsymbol{R}}\begin{pmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\end{pmatrix}$ where

$$\bar{\boldsymbol{R}} \coloneqq \begin{pmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\end{pmatrix}\boldsymbol{R}\begin{pmatrix}\boldsymbol{i}^{\mathsf{T}}\\ \boldsymbol{j}^{\mathsf{T}}\\ \boldsymbol{k}^{\mathsf{T}}\end{pmatrix}$$

is the matrix representing the same rotation for vectors expressed in the coordinate frame having the standard basis vectors as axes. In the end, the direction cosine matrix is thus subject to a two fold interpretation: as rotation matrix, it takes the coordinates of a given point with respect to a certain frame and transforms them into the coordinates of the rotated point with respect to the *same* frame; as transition matrix, it takes the coordinates of a given point with respect to the rotated frame and transforms them into the coordinates of the *same* point with respect to the original frame.

B.2 Attitude Representation

The Swiss mathematician Leonhard Euler deeply studied the motion of rigid bodies in three dimensional space and laid the foundations of **attitude kinematics**, the branch of mechanics dealing with angular motions: he formulated the fundamental theorem for **rotations**, which states that, if at least a point of a rigid body remains still, the displacement of that body is a **rotation** about a certain axis that fixed point belongs to and there exists a unique spatial direction which is not altered by the displacement.

Such a claim is perfectly consistent with the characteristics just derived for **rotation matrices**: let $\boldsymbol{R} \in \mathrm{SO}(3)$ describe a rotation, then

$$\boldsymbol{R}\boldsymbol{v} = \boldsymbol{v} \tag{B.9}$$

for all vectors $\boldsymbol{v} \in \mathbb{R}^3$ parallel to the eigenvector associated to the eigenvalue 1.

B.2.1 Axis-Angle Representation

The matrix representing a rotation can be defined by considering the axis the rotation occur about, termed **axis of rotation** or **rotation axis**, and the angle the rotation occur by, termed **angle of rotation** or **rotation angle**. The rotation matrix is accordingly denoted as $\boldsymbol{R}(\boldsymbol{u}, \theta)$, where the unit vector $\boldsymbol{u} \in \mathbb{R}^3$ represents the rotation axis, while the scalar $\theta \in \mathbb{R}$ represents the rotation angle. In such a case, eq. (B.9) becomes

$$\boldsymbol{R}(\boldsymbol{u}, \theta)\, \boldsymbol{u} = \boldsymbol{u} \tag{B.10}$$

because the rotation axis is not modified by the rotation.

When the rotation axis coincides with one of the unit vectors $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$, the rotation matrices are denoted as

$$\boldsymbol{R}_x(\theta) \coloneqq \boldsymbol{R}(\boldsymbol{i}, \theta) \tag{B.11a}$$
$$\boldsymbol{R}_y(\theta) \coloneqq \boldsymbol{R}(\boldsymbol{j}, \theta) \tag{B.11b}$$
$$\boldsymbol{R}_z(\theta) \coloneqq \boldsymbol{R}(\boldsymbol{k}, \theta) \tag{B.11c}$$

using the labels x,y,z of the coordinate axes for the sake of conciseness.

Matrix Parametrisation

A vector $\boldsymbol{v} \in \mathbb{R}^3$ subject to the rotation is decomposed, through eqs. (A.1) and (A.2), as

$$\boldsymbol{v} = \boldsymbol{v}_{\parallel} + \boldsymbol{v}_{\perp}$$

where

$$\boldsymbol{v}_{\parallel} = \big(\boldsymbol{v}^{\mathsf{T}}\boldsymbol{u}\big)\boldsymbol{u} = \boldsymbol{u}\big(\boldsymbol{u}^{\mathsf{T}}\boldsymbol{v}\big) = \boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\boldsymbol{v}$$

is the axial component of $\boldsymbol{v}$ with respect to $\boldsymbol{u}$, and

$$\boldsymbol{v}_{\perp} = \boldsymbol{v} - \big(\boldsymbol{v}^{\mathsf{T}}\boldsymbol{u}\big)\boldsymbol{u} = \boldsymbol{v} - \boldsymbol{u}\big(\boldsymbol{u}^{\mathsf{T}}\boldsymbol{v}\big) = \big(\boldsymbol{I} - \boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}\big)\boldsymbol{v}$$

is the normal component of $\boldsymbol{v}$ with respect to $\boldsymbol{u}$: the axial component is left as it is, whereas the normal component is rotated of an angle θ away from the plane identified by the axis $\boldsymbol{u}$ and the vector $\boldsymbol{v}$ itself.

The vectors $\boldsymbol{v}_{\parallel}$, $\boldsymbol{v}_{\perp}$, $\boldsymbol{u} \times \boldsymbol{v}$ clearly make up a right-handed triple that serves as basis of $\mathbb{R}^3$: the original vector $\boldsymbol{v}$ has only two orthogonal components $\boldsymbol{v}_{\parallel}$ and $\boldsymbol{v}_{\perp}$, while the rotated vector $\boldsymbol{R}(\boldsymbol{u}, \theta)\, \boldsymbol{v}$ has also a component along $\boldsymbol{u} \times \boldsymbol{v}$. The vectors $\boldsymbol{v}_{\perp}$ and $\boldsymbol{u} \times \boldsymbol{v}$ define a plane orthogonal to the rotation axis and have the same magnitude, because

$\|\boldsymbol{v}_\perp\|^2 = (\boldsymbol{v} - (\boldsymbol{v}\cdot\boldsymbol{u})\,\boldsymbol{u})\cdot(\boldsymbol{v} - (\boldsymbol{v}\cdot\boldsymbol{u})\,\boldsymbol{u}) = \|\boldsymbol{v}\|^2 - (\boldsymbol{u}\cdot\boldsymbol{v})^2 = \|\boldsymbol{u}\times\boldsymbol{v}\|^2$, thus the vector rotated by θ has components $\cos\theta\,\boldsymbol{v}_\perp$ and $\sin\theta\,\boldsymbol{u}\times\boldsymbol{v}$ along $\boldsymbol{v}_\perp$ and $\boldsymbol{u}\times\boldsymbol{v}$ respectively, on the plane perpendicular to the rotation axis, in compliance with the sense of rotation. The transformation due to the rotation is thus given by

$$\boldsymbol{R}(\boldsymbol{u},\theta)\,\boldsymbol{v} = \boldsymbol{v}_\parallel + \cos\theta\,\boldsymbol{v}_\perp + \sin\theta\,\boldsymbol{u}\times\boldsymbol{v}$$

or equivalently, making $\boldsymbol{v}$ explicit, by

$$\boldsymbol{R}(\boldsymbol{u},\theta)\,\boldsymbol{v} = \boldsymbol{u}\boldsymbol{u}^\mathsf{T}\boldsymbol{v} + \cos\theta\big(\boldsymbol{I} - \boldsymbol{u}\boldsymbol{u}^\mathsf{T}\big)\boldsymbol{v} + \sin\theta\,\boldsymbol{S}(\boldsymbol{u})\,\boldsymbol{v}$$

which holds $\forall\boldsymbol{v}\in\mathbb{R}^3$ hence

$$\boldsymbol{R}(\boldsymbol{u},\theta) = (\cos\theta)\,\boldsymbol{I} + (\sin\theta)\,\boldsymbol{S}(\boldsymbol{u}) + (1-\cos\theta)\,\boldsymbol{u}\boldsymbol{u}^\mathsf{T} \tag{B.12}$$

is the parametrisation of the rotation matrix based on $\boldsymbol{u}$ and θ. Applying eq. (A.37) to $\boldsymbol{u}$ yields $\boldsymbol{S}^2(\boldsymbol{u}) = \boldsymbol{u}\boldsymbol{u}^\mathsf{T} - \boldsymbol{I}$, thus the rotation matrix may be compactly expressed as

$$\boldsymbol{R}(\boldsymbol{u},\theta) = \boldsymbol{I} + (\sin\theta)\,\boldsymbol{S}(\boldsymbol{u}) + (1-\cos\theta)\,\boldsymbol{S}^2(\boldsymbol{u}) \tag{B.13}$$

using only trigonometric functions of the rotation angle and the skew symmetric matrix associated to the rotation axis. Its entries are given by

$$\boldsymbol{R}(\boldsymbol{u},\theta) = \begin{pmatrix} \mathrm{c}_\theta + (1-\mathrm{c}_\theta)\,u_1^2 & (1-\mathrm{c}_\theta)\,u_1u_2 - \mathrm{s}_\theta u_3 & (1-\mathrm{c}_\theta)\,u_1u_3 + \mathrm{s}_\theta u_2 \\ (1-\mathrm{c}_\theta)\,u_2u_1 + \mathrm{s}_\theta u_3 & \mathrm{c}_\theta + (1-\mathrm{c}_\theta)\,u_2^2 & (1-\mathrm{c}_\theta)\,u_2u_3 - \mathrm{s}_\theta u_1 \\ (1-\mathrm{c}_\theta)\,u_3u_1 - \mathrm{s}_\theta u_2 & (1-\mathrm{c}_\theta)\,u_3u_2 + \mathrm{s}_\theta u_1 & \mathrm{c}_\theta + (1-\mathrm{c}_\theta)\,u_3^2 \end{pmatrix} \tag{B.14}$$

where the shrunk notation $\mathrm{c}_\theta \coloneqq \cos\theta$ and $\mathrm{s}_\theta \coloneqq \sin\theta$ is used for the sake of conciseness. Only three out of the four parameters u_1, u_2, u_3 and θ are actually independent because $u_1^2 + u_2^2 + u_3^2 = 1$ must hold; in compliance, six constraints are implicitly set by the orthogonality condition $\boldsymbol{R}(\boldsymbol{u},\theta)^\mathsf{T}\,\boldsymbol{R}(\boldsymbol{u},\theta) = \boldsymbol{I}$.

Power Series

Skew symmetric matrices associated to unit vectors feature an interesting property: since

$$\begin{aligned}\boldsymbol{S}^3(\boldsymbol{u}) &= \boldsymbol{S}(\boldsymbol{u})\,\boldsymbol{S}^2(\boldsymbol{u}) = \boldsymbol{S}(\boldsymbol{u})\left(\boldsymbol{u}\boldsymbol{u}^\mathsf{T} - \boldsymbol{I}\right) = -\boldsymbol{S}(\boldsymbol{u}) \\ \boldsymbol{S}^4(\boldsymbol{u}) &= \boldsymbol{S}(\boldsymbol{u})\,\boldsymbol{S}^3(\boldsymbol{u}) = -\boldsymbol{S}(\boldsymbol{u})\,\boldsymbol{S}(\boldsymbol{u}) = -\boldsymbol{S}^2(\boldsymbol{u})\end{aligned}$$

then, in general, all successive powers of $\boldsymbol{S}(\boldsymbol{u})$ are given by

$$\boldsymbol{S}^{2k+1}(\boldsymbol{u}) = (-1)^k\,\boldsymbol{S}(\boldsymbol{u})\,,\ \boldsymbol{S}^{2k}(\boldsymbol{u}) = (-1)^k\,\boldsymbol{S}^2(\boldsymbol{u}) \tag{B.15}$$

so $\boldsymbol{S}(\boldsymbol{u})$ generates a cyclic group. Therefore, remembering the Taylor series expansions

$$\sin\theta = \sum_{k=0}^{\infty}\frac{\theta^{2k+1}}{(2k+1)!},\ \cos\theta = \sum_{k=0}^{\infty}\frac{\theta^{2k}}{(2k)!}$$

the rotation matrix can be further rewritten as

$$\boldsymbol{R}(\boldsymbol{u},\theta) = \boldsymbol{I} + \boldsymbol{S}(\boldsymbol{u}) \sum_{k=0}^{\infty} \frac{(-1)^k \, (\theta)^{2k+1}}{(2k+1)!} + \boldsymbol{S}^2(\boldsymbol{u}) \sum_{k=1}^{\infty} \frac{(-1)^k \, (\theta)^{2k}}{(2k)!} =$$
$$= \boldsymbol{I} + \sum_{k=0}^{\infty} \frac{(\boldsymbol{S}(\boldsymbol{u})\,\theta)^{2k+1}}{(2k+1)!} + \sum_{k=1}^{\infty} \frac{(\boldsymbol{S}(\boldsymbol{u})\,\theta)^{2k}}{(2k)!} = \sum_{k=0}^{\infty} \frac{(\boldsymbol{S}(\boldsymbol{u})\,\theta)^{k}}{k!}$$

or, more concisely as

$$\boldsymbol{R}(\boldsymbol{u},\theta) = \sum_{k=0}^{\infty} \frac{(\boldsymbol{S}(\boldsymbol{u})\,\theta)^{k}}{k!} = e^{\boldsymbol{S}(\boldsymbol{u})\theta} \tag{B.16}$$

using the matrix exponential notation.[1] Sometimes it gets alternatively denoted as

$$\boldsymbol{R}\,(\boldsymbol{\theta}) = \sum_{k=0}^{\infty} \frac{\boldsymbol{S}^k(\boldsymbol{\theta})}{k!} = e^{\boldsymbol{S}(\boldsymbol{\theta})} \tag{B.17}$$

where $\boldsymbol{\theta} \coloneqq \theta\boldsymbol{u}$ is a vector, termed **rotation vector**, having the rotation axis as its direction and the rotation angle as its magnitude.

Properties

The rotation matrix is manifestly a periodic function of θ with period 2π, for it appears only within trigonometric functions. All the entries are even functions of θ, u_1, u_2, u_3, for all **odd** functions of the parameters appear only inside products or squares, therefore

$$\boldsymbol{R}(-\boldsymbol{u},-\theta) = \boldsymbol{R}(\boldsymbol{u},\theta)$$
$$\boldsymbol{R}(\boldsymbol{u},-\theta) = \boldsymbol{R}(-\boldsymbol{u},\theta)$$

that is rotations are invariant under reversion of *both* sense and axis.

Thanks to eq. (B.15), the product of $\boldsymbol{R}(\boldsymbol{u},\theta_1)$ and $\boldsymbol{R}(\boldsymbol{u},\theta_2)$ is given by

$$\big(\boldsymbol{I} + \sin\theta_1 \boldsymbol{S}(\boldsymbol{u}) + (1-\cos\theta_1)\boldsymbol{S}^2(\boldsymbol{u})\big)\big(\boldsymbol{I} + \sin\theta_2 \boldsymbol{S}(\boldsymbol{u}) + (1-\cos\theta_2)\boldsymbol{S}^2(\boldsymbol{u})\big) =$$
$$= \boldsymbol{I} + (\sin\theta_1 + \sin\theta_2)\boldsymbol{S}(\boldsymbol{u}) + (1 - \cos\theta_1 + \sin\theta_1\sin\theta_2 + 1 - \cos\theta_2)\boldsymbol{S}^2(\boldsymbol{u}) +$$
$$+ ((1-\cos\theta_1)\sin\theta_2 + \sin\theta_1(1-\cos\theta_2))\boldsymbol{S}^3(\boldsymbol{u}) + (1-\cos\theta_1)(1-\cos\theta_2)\boldsymbol{S}^4(\boldsymbol{u}) =$$
$$= \boldsymbol{I} + (\sin\theta_1 + \sin\theta_2 - \sin\theta_2 + \cos\theta_1\sin\theta_2 - \sin\theta_1 + \sin\theta_1\cos\theta_2)\boldsymbol{S}(\boldsymbol{u}) +$$
$$+ (1 - \cos\theta_1 + \sin\theta_1\sin\theta_2 + 1 - \cos\theta_2 + 1 + \cos\theta_1 + \cos\theta_2 - \cos\theta_1\cos\theta_2)\boldsymbol{S}^2(\boldsymbol{u}) =$$
$$= \boldsymbol{I} + (\cos\theta_1\sin\theta_2 + \sin\theta_1\cos\theta_2)\boldsymbol{S}(\boldsymbol{u}) + (1 + \sin\theta_1\sin\theta_2 - \cos\theta_1\cos\theta_2)\boldsymbol{S}^2(\boldsymbol{u}) =$$
$$= \boldsymbol{I} + \sin(\theta_1+\theta_2)\boldsymbol{S}(\boldsymbol{u}) + (1 - \cos(\theta_1+\theta_2))\boldsymbol{S}^2(\boldsymbol{u})$$

hence it yields

$$\boldsymbol{R}(\boldsymbol{u},\theta_1)\,\boldsymbol{R}(\boldsymbol{u},\theta_2) = \boldsymbol{R}(\boldsymbol{u},\theta_1+\theta_2)$$

[1] The **matrix exponential** is defined as $e^{\boldsymbol{A}} \coloneqq \sum_{k=0}^{\infty} \frac{\boldsymbol{A}^k}{k!}$.

that is a sequence of rotations about the *same* axis is equivalent to a single rotation.

The symmetries of the rotation matrix imply that

$$\boldsymbol{R}^{\mathsf{T}}(\boldsymbol{u},\theta) = \boldsymbol{R}(\boldsymbol{u},-\theta) = \boldsymbol{R}(-\boldsymbol{u},\theta)$$

hence reversing either the rotation axis or the rotation sense gives the inverse rotation. Substituting $\theta = \pm\pi$ into eq. (B.12) gives

$$\boldsymbol{R}(\boldsymbol{u},\pm\pi) = \boldsymbol{R}(-\boldsymbol{u},\pm\pi) = -\boldsymbol{I} + 2\boldsymbol{u}\boldsymbol{u}^{\mathsf{T}}$$

thus rotations by a straight angle are invariant under axis reversion.

Let $\bar{\boldsymbol{R}}$ be another rotation: performing it and its inverse respectively after and before the rotation $\boldsymbol{R}(\boldsymbol{u},\theta)$ gives

$$\begin{aligned}\bar{\boldsymbol{R}}\boldsymbol{I}\bar{\boldsymbol{R}}^{\mathsf{T}} + (\sin\theta)\,\bar{\boldsymbol{R}}\boldsymbol{S}(\boldsymbol{u})\,\bar{\boldsymbol{R}}^{\mathsf{T}} + (1-\cos\theta)\bar{\boldsymbol{R}}\boldsymbol{S}(\boldsymbol{u})\,\bar{\boldsymbol{R}}^{\mathsf{T}}\bar{\boldsymbol{R}}\boldsymbol{S}(\boldsymbol{u})\,\bar{\boldsymbol{R}}^{\mathsf{T}} = \\ = \boldsymbol{I} + (\sin\theta)\boldsymbol{S}\big(\bar{\boldsymbol{R}}\boldsymbol{u}\big) + (1-\cos\theta)\boldsymbol{S}^2\big(\bar{\boldsymbol{R}}\boldsymbol{u}\big)\end{aligned}$$

thanks to eq. (A.40), implying that

$$\bar{\boldsymbol{R}}\boldsymbol{R}(\boldsymbol{u},\theta)\,\bar{\boldsymbol{R}}^{\mathsf{T}} = \boldsymbol{R}\big(\bar{\boldsymbol{R}}\boldsymbol{u},\theta\big) \tag{B.18}$$

which is a rotation by the *same* angle θ about the *rotated* axis $\bar{\boldsymbol{R}}\boldsymbol{u}$.

Parameters Extraction

The rotation matrix is here denoted by

$$\boldsymbol{R} = \big(r_{ij}\big)_{\substack{1\le i\le 3\\ 1\le j\le 3}}$$

for the sake of simplicity. Observing eq. (B.14) easily gives the trace

$$\operatorname{tr}\boldsymbol{R} = 3\cos\theta + (1-\cos\theta)\,\|\boldsymbol{u}\|^2 = 1 + 2\cos\theta \tag{B.19}$$

which in turn gives the rotation angle

$$\theta = \arccos\left(\frac{\operatorname{tr}\boldsymbol{R} - 1}{2}\right)$$

inverting the cosine. If $\sin\theta \neq 0$, remembering eq. (B.13) suggests that

$$\boldsymbol{S}(\boldsymbol{u}) = \frac{1}{2\sin\theta}\left(\boldsymbol{R} - \boldsymbol{R}^{\mathsf{T}}\right)$$

therefore the rotation axis can be computed as

$$\boldsymbol{u} = \frac{1}{2\sin\theta}\begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix}$$

once the rotation angle has been found. If $\cos\theta = 1$, $\boldsymbol{R}$ degenerates into $\boldsymbol{I}$ hence $\boldsymbol{u}$ is undefined, whereas if $\cos\theta = -1$, $\boldsymbol{R}$ degenerates into $\boldsymbol{I} + 2\boldsymbol{S}^2(\boldsymbol{u}) = 2\boldsymbol{u}\boldsymbol{u}^\mathsf{T} - \boldsymbol{I}$, hence $\boldsymbol{u}$ can be computed normalising any non zero column of the dyad $\boldsymbol{R} + \boldsymbol{I} = 2\boldsymbol{u}\boldsymbol{u}^\mathsf{T}$. Using eq. (B.19) into eq. (A.43) gives the characteristic polynomial

$$\begin{aligned} p(\boldsymbol{R}) &= (\lambda - 1)\left(\lambda^2 + (1 - \operatorname{tr}\boldsymbol{R})\,\lambda + 1\right) = (\lambda - 1)\left(\lambda^2 - 2\cos\theta\lambda + 1\right) = \\ &= (\lambda - 1)\left(\lambda^2 - e^{+i\theta}\lambda - e^{-i\theta}\lambda + 1\right) = (\lambda - 1)\left(\lambda - e^{+i\theta}\right)\left(\lambda - e^{-i\theta}\right) \end{aligned}$$

which highlights the eigenvalues 1, $e^{+i\theta}$ and $e^{+i\theta}$.

B.2.2 Euler Angles

Furthermore, Euler proved that a rotation may be described also by means of a sequence of three rotations, where the axes are vector constants while the angles are scalar variable; specifically, the overall rotation matrix is given by the product

$$\boldsymbol{R}\big(\boldsymbol{u}_\phi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \theta)\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big) \tag{B.20}$$

where ϕ, θ, ψ are known as **Euler angles**.

Necessity and Sufficiency

The Euler angles formulation can be used in place of the axis-angle representation, provided that it is capable of representing *whichever* rotation simply by tuning the values of the angles. In particular, there must exist two combinations of angles ϕ, θ, ψ such that the resulting rotation transform $\boldsymbol{u}_\psi$ into $\boldsymbol{u}_\phi$ and $-\boldsymbol{u}_\phi$: the condition

$$\begin{gathered} \pm\boldsymbol{u}_\phi = \boldsymbol{R}\big(\boldsymbol{u}_\phi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \theta)\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big)\, \boldsymbol{u}_\psi = \boldsymbol{R}\big(\boldsymbol{u}_\phi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \theta)\, \boldsymbol{u}_\psi \iff \\ \boldsymbol{R}(\boldsymbol{u}_\theta, \theta)\, \boldsymbol{u}_\psi = \pm\boldsymbol{R}\big(\boldsymbol{u}_\phi, -\phi\big)\, \boldsymbol{u}_\phi = \pm\boldsymbol{u}_\phi \end{gathered}$$

implies the two fold constraint $\pm\boldsymbol{u}_\phi \cdot \boldsymbol{u}_\theta = \boldsymbol{u}_\psi \cdot \boldsymbol{u}_\theta$, satisfied only if

$$\boldsymbol{u}_\phi \cdot \boldsymbol{u}_\theta = \boldsymbol{u}_\psi \cdot \boldsymbol{u}_\theta = 0 \tag{B.21}$$

which gives a necessary condition for the matrix in eq. (B.20) to represent *any* rotation. In order to prove that this condition is also sufficient, it must be shown that the matrix in eq. (B.20) is capable of transforming a given orthonormal basis into any other orthonormal basis. Let $\overline{\theta}$ be the angle such that $\boldsymbol{u}_\phi = \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_\psi$; since

$$\boldsymbol{R}\big(\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_\psi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \theta)\, \boldsymbol{R}(\boldsymbol{u}_\psi, \psi) = \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\, \boldsymbol{R}(\boldsymbol{u}_\theta, -\overline{\theta})\, \boldsymbol{R}(\boldsymbol{u}_\theta, \theta)\, \boldsymbol{R}(\boldsymbol{u}_\psi, \psi)$$

owing to eq. (B.18), then eq. (B.20) may be rewritten as

$$\boldsymbol{R}\big(\boldsymbol{u}_\phi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \theta)\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big) = \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big) \tag{B.22}$$

where $\tilde{\theta} \coloneqq \theta - \overline{\theta}$.

The three unit vectors $\boldsymbol{u}_\psi$, $\boldsymbol{u}_\theta$, $\boldsymbol{u}_\psi \times \boldsymbol{u}_\theta$ clearly make up an orthonormal basis of $\mathbb{R}^3$. Let $\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_1 = \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big)\, \boldsymbol{u}_\psi$: since

$$\boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}(\boldsymbol{u}_\psi, \psi)\, \boldsymbol{u}_\psi = \boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{u}_\psi =$$
$$= \boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\, \big(\cos\tilde{\theta}\, \boldsymbol{u}_\psi + \sin\tilde{\theta}\, \boldsymbol{S}(\boldsymbol{u}_\theta)\, \boldsymbol{u}_\psi\big) = \cos\tilde{\theta}\, \boldsymbol{u}_\psi + \sin\phi \sin\tilde{\theta}\, \boldsymbol{u}_\theta - \cos\phi \sin\tilde{\theta}\, \boldsymbol{S}(\boldsymbol{u}_\theta)\, \boldsymbol{u}_\psi$$

then $\boldsymbol{u}_1$ is a linear combination of $\boldsymbol{u}_\psi$, $\boldsymbol{u}_\theta$, $\boldsymbol{u}_\psi \times \boldsymbol{u}_\theta$. The angles $\tilde{\theta}$ and ψ may be properly tuned to transform $\boldsymbol{u}_\psi$ into *any* desired unit vector $\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_1$, while θ is automatically given by $\theta = \tilde{\theta} + \overline{\theta}$. Let $\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_2 = \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big)\, \boldsymbol{u}_\psi$: since

$$\boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}(\boldsymbol{u}_\psi, \psi)\, \boldsymbol{u}_\theta = \boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, (\cos\psi\, \boldsymbol{u}_\theta + \sin\psi\, \boldsymbol{S}(\boldsymbol{u}_\psi)\, \boldsymbol{u}_\theta) =$$
$$= \boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\, \big(\sin\tilde{\theta} \sin\psi\, \boldsymbol{u}_\psi + \cos\tilde{\theta} \sin\psi\, \boldsymbol{S}(\boldsymbol{u}_\psi)\, \boldsymbol{u}_\theta + \cos\psi\, \boldsymbol{u}_\theta\big) = \sin\tilde{\theta} \sin\psi\, \boldsymbol{u}_\psi +$$
$$+ (\cos\phi\cos\psi - \sin\phi\cos\tilde{\theta}\sin\psi)\boldsymbol{u}_\theta + (\cos\phi\cos\tilde{\theta}\sin\psi + \sin\phi\cos\psi)\boldsymbol{S}(\boldsymbol{u}_\psi)\, \boldsymbol{u}_\theta$$

then also $\boldsymbol{u}_2$ is a linear combination of $\boldsymbol{u}_\psi$, $\boldsymbol{u}_\theta$, $\boldsymbol{u}_\psi \times \boldsymbol{u}_\theta$. Moreover

$$\boldsymbol{u}_1 \cdot \boldsymbol{u}_2 = \cos\tilde{\theta} \sin\tilde{\theta} \sin\psi + \sin\phi \sin\tilde{\theta} \cos\phi \cos\psi - \sin\phi^2 \sin\tilde{\theta} \cos\tilde{\theta} \sin\psi +$$
$$- \cos\phi^2 \sin\tilde{\theta} \cos\tilde{\theta} \sin\psi - \cos\phi \sin\tilde{\theta} \sin\phi \cos\psi = 0$$

shows that $\boldsymbol{u}_2$ and $\boldsymbol{u}_1$ are orthogonal as well, in compliance with eq. (A.34) applied to eq. (B.21). The other angle ϕ may be properly tuned to transform $\boldsymbol{u}_\theta$ into *any* desired unit vector $\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_2$ on the plane orthogonal to the unit vector $\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_1$. Let $\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_3 = \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}(\boldsymbol{u}_\psi, \psi)\, (\boldsymbol{u}_\psi \times \boldsymbol{u}_\theta)$: since, from eq. (A.42),

$$\boldsymbol{R}\big(\boldsymbol{u}_\psi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big)\, (\boldsymbol{u}_\psi \times \boldsymbol{u}_\theta) =$$
$$= \boldsymbol{R}\big(\boldsymbol{u}_\psi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big)\, \boldsymbol{u}_\psi \times \boldsymbol{R}\big(\boldsymbol{u}_\psi, \phi\big)\, \boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\, \boldsymbol{R}\big(\boldsymbol{u}_\psi, \psi\big)\, \boldsymbol{u}_\theta = \boldsymbol{u}_1 \times \boldsymbol{u}_2$$

then $\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_3 = \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, (\boldsymbol{u}_1 \times \boldsymbol{u}_2) = \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_1 \times \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\, \boldsymbol{u}_2$ completes the basis.

Axes Sequence

When the unit vectors $\boldsymbol{u}_\phi$, $\boldsymbol{u}_\theta$, $\boldsymbol{u}_\psi$ representing the axes of the Euler rotations are taken among the standard basis vectors $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, $\boldsymbol{e}_3$, the rotation matrix is denoted as

$$\boldsymbol{R}_{ijk}\, (\psi, \theta, \phi) = \boldsymbol{R}(\boldsymbol{e}_k, \phi)\, \boldsymbol{R}\big(\boldsymbol{e}_j, \theta\big)\, \boldsymbol{R}(\boldsymbol{e}_i, \psi)$$

with $i, j, k \in \{1, 2, 3\}$ and $j \neq i$, $j \neq k$, in compliance with eq. (B.21).

According to the axes sequence, two families of Euler angles can be identified: when $i = k$, the angles are said **proper** or **symmetric**, whereas, when $i \neq k$ the angles are said **improper** or **asymmetric**; angles of symmetric sets are also termed simply **Euler angles**, while angles of asymmetric sets are also termed **Tait-Bryan angles**, **Cardan angles**, or **Roll-Pitch-Yaw**, terms borrowed from nautical and aeronautical jargon.

If the rotation axes are picked among the coordinate axes $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ of a given reference frame, the individual rotation matrices are traditionally denoted as in eq. (B.11). The most common symmetric Euler rotations are the zxz sequence $\boldsymbol{R}_z(\phi)\, \boldsymbol{R}_x(\theta)\, \boldsymbol{R}_z(\psi)$ and the zyz sequence $\boldsymbol{R}_z(\phi)\, \boldsymbol{R}_y(\theta)\, \boldsymbol{R}_z(\psi)$, while the most common asymmetric Euler rotations are the zyx sequence $\boldsymbol{R}_x(\phi)\, \boldsymbol{R}_y(\theta)\, \boldsymbol{R}_z(\psi)$ and the xyz sequence $\boldsymbol{R}_z(\phi)\, \boldsymbol{R}_y(\theta)\, \boldsymbol{R}_x(\psi)$.

Ambiguity

Putting $\boldsymbol{R}(\boldsymbol{u}_\psi, -\pi)\,\boldsymbol{R}(\boldsymbol{u}_\psi, \pi) = \boldsymbol{I}$ before and after $\boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})$ into eq. (B.22) yields

$$\begin{aligned}
&\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\,\boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \psi) = \\
&= \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \phi)\,\boldsymbol{R}(\boldsymbol{u}_\psi, -\pi)\,\boldsymbol{R}(\boldsymbol{u}_\psi, \pi)\,\boldsymbol{R}(\boldsymbol{u}_\theta, \tilde{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, -\pi)\,\boldsymbol{R}(\boldsymbol{u}_\psi, \pi)\,\boldsymbol{R}(\boldsymbol{u}_\psi, \psi) = \\
&= \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \phi - \pi)\,\boldsymbol{R}\big(\boldsymbol{R}(\boldsymbol{u}_\psi, \pi)\,\boldsymbol{u}_\theta, \tilde{\theta}\big)\,\boldsymbol{R}(\boldsymbol{u}_\psi, \psi + \pi) = \\
&= \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \phi - \pi)\,\boldsymbol{R}(-\boldsymbol{u}_\theta, \tilde{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \psi + \pi) = \\
&= \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \phi - \pi)\,\boldsymbol{R}(\boldsymbol{u}_\theta, -\tilde{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \psi + \pi) = \\
&= \boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \phi - \pi)\,\boldsymbol{R}(\boldsymbol{u}_\theta, -\overline{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\theta, \overline{\theta} - \tilde{\theta})\,\boldsymbol{R}(\boldsymbol{u}_\psi, \psi + \pi)
\end{aligned}$$

because of eq. (B.18), giving rise to the equality

$$\boldsymbol{R}(\boldsymbol{u}_\phi, \phi)\,\boldsymbol{R}(\boldsymbol{u}_\theta, \theta)\,\boldsymbol{R}(\boldsymbol{u}_\psi, \psi) = \boldsymbol{R}(\boldsymbol{u}_\phi, \phi - \pi)\,\boldsymbol{R}(\boldsymbol{u}_\theta, 2\overline{\theta} - \theta)\,\boldsymbol{R}(\boldsymbol{u}_\psi, \psi + \pi) \tag{B.23}$$

which puts in evidence that this attitude representation is intrinsically subject to an ambiguity. Euler rotations feature $\overline{\theta} = 0$, because the first and the third rotation axes are coincident, whereas Roll-Pitch-Yaw rotations feature $\overline{\theta} = \pm\frac{\pi}{2}$, because the three rotation axes are all orthogonal to each other.

B.3 Homogeneous Representation

B.3.1 Homogeneous Coordinates

The **homogeneous coordinate** or **projective coordinate** representation of a point in a n-dimensional space is a $(n+1)$-tuple[2] that returns the original point when subject to a certain **perspective projection** transformation; the original point is the perspective projection of the homogeneous point over a certain hyperplane in the $(n+1)$-dimensional space, termed projection hyperplane or viewplane, with a certain centre of projection. The perspective projection of the homogeneous coordinates $\boldsymbol{h}$ into the point $\boldsymbol{x}$ they represent can be rendered through the matrix product

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \frac{k}{h_{n+1}} \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_n \\ h_{n+1} \end{pmatrix}$$

succinctly written as

$$\boldsymbol{x} = \frac{k}{h_{n+1}} \begin{pmatrix} \boldsymbol{I}_n & \boldsymbol{0} \end{pmatrix} \boldsymbol{h}.$$

where $k \neq 0$ is a scalar constant.

[2] In mathematics, a tuple is a *finite* sequence of elements; specifically, an n-tuple, where n is a non negative integer, is a sequence of n elements.

Perspective Camera Example

A **perspective camera** is considered for the sake of illustration: the aperture is located at the origin of the 3D space and the viewing direction is oriented with the z axis; the image plane is orthogonal to the viewing direction and is located behind the aperture, at a distance equal to the focal length, so it has equation $z = -f$. A 3D point with coordinates (x, y, z) is located in front of the camera, i.e. $z \geq 0$: a light ray emitted by the point goes through the aperture and hits the image plane. The coordinates (u, v) of the point mapped on the image plane depend not only on the (x, y) coordinates, but also on the z coordinate, of the original point and may be computed, provided that the value of the focal length is known, as

$$u = -\frac{f}{z}x, \quad v = -\frac{f}{z}y$$

or, in compact form, as

$$\begin{pmatrix} u \\ v \end{pmatrix} = -\frac{f}{z}\begin{pmatrix} x \\ y \end{pmatrix}$$

therefore the map on the image plane is obtained scaling and reflecting the point.

The 3D point (x, y, z) is just one homogeneous coordinate representation of the 2D point (u, v) among all possible ones; for instance, another point (x', y', z') with the same abscissa-applicate and ordinate-applicate aspect ratios

$$\frac{x'}{z'} = \frac{x}{z}, \quad \frac{y'}{z'} = \frac{y}{z}$$

yields the same perspective projection on the image plane and gives thus a homogeneous coordinate representation of (u, v) too. More in general, any 3D point $\left(-\frac{w}{f}u, -\frac{w}{f}v, w\right)$ with $w \neq 0$, is a valid homogeneous coordinate representation of the 2D point (u, v).

Three dimensional space

The same concept may be generalised to points in spaces with any dimension: a homogeneous coordinate representation of a certain point can be given if there exists a perspective projection capable of recreating that point from a point in a space with augmented dimension. A point with coordinates

$$\boldsymbol{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

may be represented by means of the homogeneous coordinates

$$\boldsymbol{h} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

provided that the condition

$$x = \frac{a}{d}, \quad y = \frac{b}{d}, \quad z = \frac{c}{d}, \quad d \neq 0$$

is satisfied. Different values of the scaling factor d give rise to as many valid homogeneous coordinate representations of the same point, hence multiplying homogeneous coordinates by a non zero constant does not modify the point they represent. As particular case, if the scaling factor is chosen to be unitary, the homogeneous coordinates can be immediately derived as

$$\boldsymbol{h} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{r} \\ 1 \end{pmatrix} \tag{B.24}$$

by simply adding 1, to the original 3D coordinates, as fourth coordinate. Homogeneous coordinates with zero scaling factor are linked to points at infinity and represent thus *pure* directions by convention.

B.3.2 Homogeneous Transformation

Any transformation a point in the three dimensional space could be subject to, e.g., rotation, dilation, translation, reflection, etc., might be elegantly and compactly rendered, making use of homogeneous coordinates representation, through a simple linear map, even when the transformation is not linear *per se*. The 4×4 matrices that pre-multiply arrays of homogeneous coordinates in order to apply some transformation to the 3D points they represent, are called **homogeneous transformation matrices**.

In order to reflect the structure of homogeneous coordinates, a homogeneous transformation matrix $\boldsymbol{T}$ is partitioned as

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{A} & \boldsymbol{b} \\ \boldsymbol{c}^{\mathsf{T}} & d \end{pmatrix} \tag{B.25}$$

where $\boldsymbol{A} \in \mathbb{R}^{3\times3}$ is a matrix that accounts for linear transformation, e.g., rotation, reflection, selective scaling, shear, $\boldsymbol{b} \in \mathbb{R}^{3\times1}$ is a vector that accounts for translation, $\boldsymbol{c} \in \mathbb{R}^{3\times1}$ is a vector that accounts for perspective transformation and $d \in \mathbb{R}$ is a scalar that accounts for overall scaling. Exactly like homogeneous coordinates, multiplying homogeneous transformation matrices by a non zero constant does not modify the transformation they represent.[3]

[3]For the sake of simplicity, everything that follows is derived using homogeneous coordinates and matrices with unitary scaling factor, but this is not mandatory whatsoever, as the results are valid regardless.

Translation

A point $\boldsymbol{r}$ subject to a translation defined by $\boldsymbol{t}$ becomes $\boldsymbol{r}+\boldsymbol{t}$, hence, if it is represented with homogeneous coordinates, then

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}$$

is the homogeneous transformation matrix corresponding to the translation. When the translation is defined parametrically, this matrix is also denoted as

$$\mathbf{Trans}\,(\boldsymbol{v}, d) = \begin{pmatrix} \boldsymbol{I} & d\boldsymbol{v} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}$$

where d is translation distance and $\boldsymbol{v}$ is the translation direction.

Rotation

A point $\boldsymbol{r}$ subject to a rotation defined by $\boldsymbol{R}$ becomes $\boldsymbol{R}\boldsymbol{r}$, hence, if it is represented with homogeneous coordinates, then

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}$$

is the homogeneous transformation matrix corresponding to the rotation. When the rotation is defined parametrically to a certain axis, this matrix is also denoted as

$$\mathbf{Rot}\,(\boldsymbol{u}, \theta) = \begin{pmatrix} \boldsymbol{R}(\boldsymbol{u}, \theta) & \boldsymbol{0} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}$$

where θ is the rotation angle and $\boldsymbol{u}$ is the rotation axis.

Reflection

A point $\boldsymbol{r}$ subject to a reflection with respect to the plane defined by $\boldsymbol{n}^\mathsf{T}\boldsymbol{r} + d = 0$ becomes

$$\boldsymbol{r} - 2\frac{\boldsymbol{n}\cdot\boldsymbol{r}+d}{\boldsymbol{n}\cdot\boldsymbol{n}}\boldsymbol{n} = \left(\boldsymbol{I} - 2\frac{\boldsymbol{n}\boldsymbol{n}^\mathsf{T}}{\boldsymbol{n}^\mathsf{T}\boldsymbol{n}}\right)\boldsymbol{r} - 2\frac{d\boldsymbol{n}}{\boldsymbol{n}^\mathsf{T}\boldsymbol{n}}$$

because of eq. (A.1), hence, if it is represented with homogeneous coordinates, then

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{I} - 2\frac{\boldsymbol{n}\boldsymbol{n}^\mathsf{T}}{\boldsymbol{n}^\mathsf{T}\boldsymbol{n}} & -2\frac{d\boldsymbol{n}}{\boldsymbol{n}^\mathsf{T}\boldsymbol{n}} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}$$

is the homogeneous transformation matrix corresponding to the reflection.

Full Transformation

Points in the three dimensional space may be modified by several subsequent transformations of diverse nature: using homogeneous coordinates, a sequence of transformations could be easily represented through consecutive multiplication by corresponding homogeneous transformation matrices. Only proper rigid transformations, that is to say rotations and translations, are hereinafter contemplated.

The combination of a rotation, described by the matrix $\boldsymbol{R}$, and a translation, described by the vector $\boldsymbol{t}$, is considered: the overall transformation depends on the order in which they are performed. In particular, the homogeneous transformation matrix

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} \tag{B.26}$$

represents the transformation $\boldsymbol{r} \to \boldsymbol{R}\boldsymbol{r} \to \boldsymbol{R}\boldsymbol{r} + \boldsymbol{t}$ that corresponds to performing first the rotation and then the translation, whereas the homogeneous transformation matrix

$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{I} & \boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{R}\boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} \tag{B.27}$$

represents the transformation $\boldsymbol{r} \to \boldsymbol{r} + \boldsymbol{t} \to \boldsymbol{R}(\boldsymbol{r} + \boldsymbol{t})$ that corresponds to performing first the translation and then the rotation.

Inverse Transformation

The homogeneous transformation matrix of the inverse transformation is, of course, the inverse of the homogeneous transformation matrix of the direct transformation. Inverting the matrices in eq. (B.26) gives the homogeneous transformation matrix

$$\boldsymbol{T}^{-1} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{I} & \boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix}^{-1} = \begin{pmatrix} \boldsymbol{R}^\mathsf{T} & \boldsymbol{0} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{I} & -\boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}^\mathsf{T} & -\boldsymbol{R}^\mathsf{T}\boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} \tag{B.28}$$

that represents the inverse transformation, because multiplying the matrix by the homogeneous coordinates of the transformed vector

$$\begin{pmatrix} \boldsymbol{R}^\mathsf{T} & -\boldsymbol{R}^\mathsf{T}\boldsymbol{t} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{R}\boldsymbol{r} + \boldsymbol{t} \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}^\mathsf{T}\boldsymbol{R}\boldsymbol{r} + \boldsymbol{R}^\mathsf{T}\boldsymbol{t} - \boldsymbol{R}^\mathsf{T}\boldsymbol{t} \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{r} \\ 1 \end{pmatrix}$$

gives back exactly the original point. Homogeneous transformation matrices are always invertible: applying eq. (A.22) to eq. (B.26) indeed yields $\det \boldsymbol{T} = \det \boldsymbol{R} \det 1 = 1$.

B.3.3 Reference Frames

A reference frame is completely specified through the position of its origin and the orientation of its three coordinate axes. Reference frames are of paramount importance

in the study of rigid body kinematics: as a matter of fact, attaching a reference frame to a given rigid body naturally gives a convenient tool for describing its kinematics, as the origin may be related to the body displacement while the orientation of the axes represent the body attitude. Let $\mathcal{R}_\mathrm{E}$ be a reference frame: the translation vector

$$\boldsymbol{t}_\mathrm{E} \coloneqq \boldsymbol{o}_\mathrm{E} = \begin{pmatrix} o_{\mathrm{E}x} \\ o_{\mathrm{E}y} \\ o_{\mathrm{E}z} \end{pmatrix}$$

describes the position of $\mathcal{R}_\mathrm{E}$, while the rotation matrix

$$\boldsymbol{R}_\mathrm{E} \coloneqq \begin{pmatrix} \boldsymbol{i}_\mathrm{E} & \boldsymbol{j}_\mathrm{E} & \boldsymbol{k}_\mathrm{E} \end{pmatrix} = \begin{pmatrix} i_{\mathrm{E}x} & j_{\mathrm{E}y} & k_{\mathrm{E}z} \\ i_{\mathrm{E}y} & j_{\mathrm{E}y} & k_{\mathrm{E}z} \\ i_{\mathrm{E}z} & j_{\mathrm{E}z} & k_{\mathrm{E}z} \end{pmatrix}$$

describes the orientation of $\mathcal{R}_\mathrm{E}$. For this reason, the homogeneous transformation matrix

$$\boldsymbol{T}_\mathrm{E} \coloneqq \begin{pmatrix} \boldsymbol{R}_\mathrm{E} & \boldsymbol{t}_\mathrm{E} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{i}_\mathrm{E} & \boldsymbol{j}_\mathrm{E} & \boldsymbol{k}_\mathrm{E} & \boldsymbol{o}_\mathrm{E} \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} i_{\mathrm{E}x} & j_{\mathrm{E}x} & k_{\mathrm{E}x} & o_{\mathrm{E}x} \\ i_{\mathrm{E}y} & j_{\mathrm{E}y} & k_{\mathrm{E}y} & o_{\mathrm{E}y} \\ i_{\mathrm{E}z} & j_{\mathrm{E}z} & k_{\mathrm{E}z} & o_{\mathrm{E}z} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

compactly gives a full representation of $\mathcal{R}_\mathrm{E}$. This matrix might be interpreted also as a collection of the homogeneous coordinates related to the vectors defining the reference frame: the origin is a point, so its homogeneous coordinates have scaling factor 1, whereas the axes are directions, so their homogeneous coordinates have scaling factor 0.

Relationship between Reference Frames

Homogeneous transformation matrices are nothing but a mathematical tool to represent tightly the combination of rotations and translations. As regard reference frames description, they may be employed to represent the rigid transformation relating reference frames to an *absolute* reference frame. Sometimes, it might be necessary to describe a reference frame with respect to another *generic* reference frame, which may or may not be a preferable choice. In such a case, the homogeneous transformation matrix gives just the representation of the relative relationship between two reference frames, that have different roles, e.g., world frame and body frame, fixed frame and moving frame, ideal frame and real frame, and so on.

The homogeneous transformation matrix describing the representation of a frame $\mathcal{R}_\mathrm{F}$ into another frame $\mathcal{R}_\mathrm{E}$ is denoted as $\boldsymbol{T}_\mathrm{F}^\mathrm{E}$, where the subscript designates the represented frame while the superscript designates the representing frame. It has the following straightforward block structure

$$\boldsymbol{T}_\mathrm{F}^\mathrm{E} \coloneqq \begin{pmatrix} \boldsymbol{R}_\mathrm{F}^\mathrm{E} & \boldsymbol{t}_\mathrm{F}^\mathrm{E} \\ \boldsymbol{0}^\mathsf{T} & 1 \end{pmatrix} \tag{B.29}$$

where $\boldsymbol{R}_\mathrm{F}^\mathrm{E}$ represents the orientation of the coordinate axes of $\mathcal{R}_\mathrm{F}$ in $\mathcal{R}_\mathrm{E}$, and $\boldsymbol{t}_\mathrm{F}^\mathrm{E}$ represents the position of the origin of $\mathcal{R}_\mathrm{F}$ in $\mathcal{R}_\mathrm{E}$.

Change of Reference Frame

Considering that a reference frame is essentially a tool necessary for unambiguously representing a physical object in the three dimensional space, it might be seen as a generalisation of the concept of basis of the euclidean space. In the light of this interpretation, the homogeneous transformation matrix describing the relationship between two reference frames takes the role of a change of basis matrix.

For better clarification, even though the position of a point P in the space is an obviously unequivocal concept, it might have two different representations $\boldsymbol{r}_{\mathrm{P}}^{\mathrm{E}}$ and $\boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}}$ in two different reference frames $\mathcal{R}_{\mathrm{E}}$ and $\mathcal{R}_{\mathrm{F}}$ respectively. If the relationship between these frames is known, the position in frame E can be computed as

$$\boldsymbol{r}_{\mathrm{P}}^{\mathrm{E}} = \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}} \boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}} + \boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}} \tag{B.30}$$

whenever the orientation and the position of $\mathcal{R}_{\mathrm{F}}$ in $\mathcal{R}_{\mathrm{E}}$ are known. Thanks to eq. (B.29), this relation may be rewritten succinctly as

$$\boldsymbol{h}_{\mathrm{P}}^{\mathrm{E}} = \boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{F}} \tag{B.31}$$

where $\boldsymbol{h}_{\mathrm{P}}^{\mathrm{E}}$ and $\boldsymbol{h}_{\mathrm{P}}^{\mathrm{F}}$ are respectively the homogeneous coordinates of P in $\mathcal{R}_{\mathrm{E}}$ and $\mathcal{R}_{\mathrm{F}}$.

Similarly, $\boldsymbol{T}_{\mathrm{E}}^{\mathrm{F}}$ describes the representation of $\mathcal{R}_{\mathrm{E}}$ into $\mathcal{R}_{\mathrm{F}}$ thus

$$\boldsymbol{h}_{\mathrm{P}}^{\mathrm{F}} = \boldsymbol{T}_{\mathrm{E}}^{\mathrm{F}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{E}} \tag{B.32}$$

is the inverse of eq. (B.31), and

$$\boldsymbol{T}_{\mathrm{E}}^{\mathrm{F}} = \boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}\,-1} \tag{B.33}$$

because, trivially, $\boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}\,-1} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{E}} = \boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}\,-1} \boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}} \boldsymbol{h}_{\mathrm{P}}^{\mathrm{F}} = \boldsymbol{h}_{\mathrm{P}}^{\mathrm{F}}$. Applying eq. (B.28) gives the relation

$$\boldsymbol{R}_{\mathrm{E}}^{\mathrm{F}} = \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}\,\mathsf{T}} \tag{B.34a}$$

$$\boldsymbol{t}_{\mathrm{E}}^{\mathrm{F}} = -\boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}\,\mathsf{T}} \boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}} \tag{B.34b}$$

between the rotation and translation blocks of eqs. (B.31) and (B.32). The same result could have been proved also by computing $\boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}}$ from eq. (B.30) as

$$\boldsymbol{r}_{\mathrm{P}}^{\mathrm{E}} = \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}} \boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}} + \boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}} \;\Leftrightarrow\; \boldsymbol{r}_{\mathrm{P}}^{\mathrm{E}} - \boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}} = \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}} \boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}} \;\Leftrightarrow\; \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}\,\mathsf{T}} \left(\boldsymbol{r}_{\mathrm{P}}^{\mathrm{E}} - \boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}} \right) = \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}\,\mathsf{T}} \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}} \boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}} \;\Leftrightarrow$$
$$\boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}\,\mathsf{T}} \left(\boldsymbol{r}_{\mathrm{P}}^{\mathrm{E}} - \boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}} \right) = \boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}} \;\Leftrightarrow\; \boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}} = \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}\,\mathsf{T}} \boldsymbol{r}_{\mathrm{P}}^{\mathrm{E}} - \boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}\,\mathsf{T}} \boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}}$$

and matching it with $\boldsymbol{r}_{\mathrm{P}}^{\mathrm{F}} = \boldsymbol{R}_{\mathrm{E}}^{\mathrm{F}} \boldsymbol{r}_{\mathrm{P}}^{\mathrm{E}} + \boldsymbol{t}_{\mathrm{E}}^{\mathrm{F}}$.

Interpretation

Transition matrices might be interpreted as either changes of coordinates or transformations of bases. It could be interesting to extend this binary perspective to homogeneous transformation matrices representing the relationship between reference frames: the homogeneous coordinates relation in eq. (B.31) can be seen indeed in two different ways.

According to the so called **alias** or **passive** interpretation, $\boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}}$ gives the description of $\mathcal{R}_{\mathrm{F}}$ in $\mathcal{R}_{\mathrm{E}}$, for $\boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}}$ describes the relative orientation of $\mathcal{R}_{\mathrm{F}}$ in $\mathcal{R}_{\mathrm{E}}$ and $\boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}}$ describes the relative position of $\mathcal{R}_{\mathrm{F}}$ in $\mathcal{R}_{\mathrm{E}}$. From this point of view, $\mathcal{R}_{\mathrm{F}}$ and $\mathcal{R}_{\mathrm{E}}$ are two distinct reference frames: the position of a point is represented in a different frame, though it does not actually change (*alias* means "otherwise").

According to the so called **alibi** or **active** interpretation, $\boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}}$ gives the transformation of $\mathcal{R}_{\mathrm{F}}$ in $\mathcal{R}_{\mathrm{E}}$, for the matrix $\boldsymbol{R}_{\mathrm{F}}^{\mathrm{E}}$ transforms the axes of $\mathcal{R}_{\mathrm{E}}$ into the axes of $\mathcal{R}_{\mathrm{F}}$ and $\boldsymbol{t}_{\mathrm{F}}^{\mathrm{E}}$ transforms the origin of $\mathcal{R}_{\mathrm{E}}$ into the origin of $\mathcal{R}_{\mathrm{F}}$. From this point of view, $\mathcal{R}_{\mathrm{E}}$ and $\mathcal{R}_{\mathrm{F}}$ are two versions of the same reference: the position of a point is represented in the same frame, though it does actually change (*alibi* means "elsewhere").

Reference Frame Matrix Transformation

Sometimes it might be useful to apply generic operators, not necessarily rigid transformations, to 3D points: some of them can be represented as linear maps when using homogeneous coordinates. It is better to specify the frame of representation of the homogeneous coordinates multiplied by the matrix, as left superscript: for instance, the action of a certain operator in $\mathcal{R}_{\mathrm{E}}$ to a point with representation in $\mathcal{R}_{\mathrm{F}}$, may be described by either transforming the homogeneous coordinates from $\mathcal{R}_{\mathrm{F}}$ to $\mathcal{R}_{\mathrm{E}}$ and then multiplying them by ${}^{\mathrm{E}}\boldsymbol{M}$, i.e. as ${}^{\mathrm{E}}\boldsymbol{M}\boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}}\boldsymbol{h}^{\mathrm{F}}$, or, multiplying by ${}^{\mathrm{F}}\boldsymbol{M}$ the homogeneous coordinates and then transforming them from $\mathcal{R}_{\mathrm{F}}$ to $\mathcal{R}_{\mathrm{E}}$, i.e. as $\boldsymbol{T}_{\mathrm{F}}^{\mathrm{EF}}\boldsymbol{M}\boldsymbol{h}^{\mathrm{F}}$.

The two descriptions must be completely equivalent, hence ${}^{\mathrm{E}}\boldsymbol{M}\boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}}\boldsymbol{h}^{\mathrm{F}} = \boldsymbol{T}_{\mathrm{F}}^{\mathrm{EF}}\boldsymbol{M}\boldsymbol{h}^{\mathrm{F}}$: since this constraint must be satisfied independently of the specific homogeneous coordinates, i.e. ${}^{\mathrm{E}}\boldsymbol{M}\boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}} = \boldsymbol{T}_{\mathrm{F}}^{\mathrm{EF}}\boldsymbol{M}$, then the two fold relation

$$
{}^{\mathrm{E}}\boldsymbol{M} = \boldsymbol{T}_{\mathrm{F}}^{\mathrm{EF}}\boldsymbol{M}\boldsymbol{T}_{\mathrm{E}}^{\mathrm{F}} \tag{B.35a}
$$

$$
{}^{\mathrm{F}}\boldsymbol{M} = \boldsymbol{T}_{\mathrm{E}}^{\mathrm{FE}}\boldsymbol{M}\boldsymbol{T}_{\mathrm{F}}^{\mathrm{E}} \tag{B.35b}
$$

gives the transformation of the matrix under change of reference frame.

Bibliography

[1] Richard P. Paul, *Robot Manipulators: Mathematics, Programming and Control*, The MIT Press, Cambridge, Massachusetts, United States, October 1981.

[2] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, 3rd edition, Pearson Prentice Hall, Upper Saddle River, New Jersey, United States, August 2004.

[3] R. M. Murray, Z. Li, S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, London, United Kingdom, July 2017.

[4] J. Angeles, A. Kecskemethy, *Kinematics and Dynamics of Multi-Body Systems*, Springer, Vienna, Austria, March 2004.

[5] J. Angeles, *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*, 4th Edition, Springer, Cham, Switzerland, January 2014.

[6] B. Siciliano, O. Khatib, *Handbook of Robotics*, 2nd edition, Springer, Cham, Switzerland, June 2016.

[7] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, *Robotics: Modelling, Planning and Control*, Springer, London, United Kingdom, October 2010.

[8] J. Denavit, R. S. Hartenberg, *Kinematic Synthesis of Linkages*, McGraw-Hill, New York, United States, December 1964.

[9] J. Denavit, R. S. Hartenberg, *A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices*, ASME Journal of Applied Mechanics, vol. 22, no. 2, pp. 215-221, June 1955.

[10] S. Hayati, M. Mirmirani, *Improving the Absolute Positioning Accuracy of Robot Manipulators*, Journal of Robotic Systems, vol. 2, no. 4, pp. 397-413, January 1985.

[11] W. K. Veitschegger, C. Wu, *Robot Accuracy Analysis Based on Kinematics*, IEEE Journal of Robotics and Automation, vol. 2, no. 3, pp. 171-179, September 1986.

[12] R. P. Paul, B. Shimano, G. E. Mayer, *Kinematic Control Equations for Simple Manipulators*, IEEE Transactions on Systems, Man, and Cybernetics, vol. 11, no. 6, pp. 449-455, June 1981.

[13] L. J. Everett, M. Driels, B. W. Mooring, *Kinematic Modelling for Robot Calibration*, Proceedings of 1987 IEEE International Conference on Robotics and Automation, pp. 183-189, April 1987.

[14] K. Schröer, S. L. Albright, M. Grethlein, *Complete, minimal and model-continuous kinematic models for robot calibration*, Robotics and Computer-Integrated Manufacturing, vol. 13, no. 1, pp. 73-85, March 1997.

[15] B. Mooring, M. Driels, Z. Roth, *Fundamentals of Manipulator Calibration*, John Wiley & Sons, New York, United States, April 1991.

[16] Z. S. Roth, B. W. Mooring, B. Ravani, *An Overview of Robot Calibration*, IEEE Journal of Robotics and Automation, vol. 3, no. 5, pp. 377-385, October 1987.

[17] B. W. Mooring, T. J. Pack, *Calibration Procedure For An Industrial Robot*, Proceedings of 1988 IEEE International Conference on Robotics and Automation, vol. 2, pp. 786-791, April 1988.

[18] B. W. Mooring, S. S. Padavala, *The Effect of Kinematic Model Complexity on Manipulator Accuracy*, Proceedings of 1989 International Conference on Robotics and Automation, vol. 1, pp. 593-598, May 1989.

[19] W. K. Veitschegger, C. Wu, *Robot Calibration and Compensation*, IEEE Journal of Robotics and Automation, vol. 4, no. 6, pp. 643-656, December 1988.

[20] D. E. Whitney, C. A. Lozinski, J. M. Rourke, *Industrial Robot Forward Calibration Method and Results*, ASME Journal of Dynamic Systems, Measurement, and Control, vol. 108, no. 1, pp. 1-8, March 1986.

[21] L. J. Everett, T. -W. Hsu, *The Theory of Kinematic Parameter Identification for Industrial Robots*, Journal of Dynamic Systems, Measurement, and Control, ASME, vol. 110, no. 3, pp. 96-100, March 1988.

[22] R. Li Y. Zhao, *Dynamic error compensation for industrial robot based on thermal effect model*, Measurement, vol. 88, pp. 113-120, June 2016.

[23] R. Waiboer, R. Aarts, B. Jonker, *Velocity Dependence of Joint Friction in Robotic Manipulators with Gear Transmissions*, International Journal of Human Resources Development and Management, pp. 1-19, June 2005.

[24] C. Canuto, A. Tabacco, *Mathematical Analysis II*, 2nd Edition, Springer, Cham, Switzerland, December 2014.

[25] L. E. Scales, *Introduction to Non-Linear Optimization*, Palgrave Macmillan, Basingstoke, United Kingdom, January 1985.

[26] Y. Nesterov, *Introductory Lectures on Convex Optimization, A Basic Course*, Springer Science & Business Media, New York, NY, United States, December 2003.

[27] P. Chen, *Hessian Matrix vs. Gauss–Newton Hessian Matrix*, SIAM Journal on Numerical Analysis, vol. 49 no. 4, pp. 1417–1435, July 2011.

[28] J. L. Crassidis, F. Landis Markley, *Fundamentals of Spacecraft Attitude Determination and Control*, Springer, New York, NY, United States, June 2014.

[29] International Bureau of Weights and Measures - Joint Committee for Guides in Metrology 200:2012, *International vocabulary of metrology – Basic and general concepts and associated terms (VIM*, 3rd edition, Version 2008 with minor corrections, JCGM 2012.

[30] International Organization for Standardization, ISO 9283, *Manipulating Industrial Robots - Performance Criteria and Related Test Methods*, August 2005.

[31] International Organization for Standardization, ISO 9946, *Manipulating Industrial Robots - Presentation of Characteristics*, March 2005.