Bruno Siciliano • Lorenzo Sciavicco
Luigi Villani • Giuseppe Oriolo

Robotics

Modelling, Planning and Control

Springer

Contents

Appendices

1

Introduction

Robotics is concerned with the study of those machines that can replace human beings in the execution of a task, as regards both physical activity and decision making. The goal of the introductory chapter is to point out the problems related to the use of *robots* in *industrial* applications, as well as the perspectives offered by *advanced robotics*. A classification of the most common mechanical structures of *robot manipulators* and *mobile robots* is presented. Topics of *modelling*, *planning* and *control* are introduced which will be examined in the following chapters. The chapter ends with a list of references dealing with subjects both of specific interest and of related interest to those covered by this textbook.

1.1 Robotics

Robotics has profound cultural roots. Over the course of centuries, human beings have constantly attempted to seek substitutes that would be able to mimic their behaviour in the various instances of interaction with the surrounding environment. Several motivations have inspired this continuous search referring to philosophical, economic, social and scientific principles.

One of human beings' greatest ambitions has been to give life to their artifacts. The legend of the Titan Prometheus, who molded humankind from clay, as well as that of the giant Talus, the bronze slave forged by Hephaestus, testify how Greek mythology was influenced by that ambition, which has been revisited in the tale of Frankenstein in modern times.

Just as the giant Talus was entrusted with the task of protecting the island of Crete from invaders, in the Industrial Age a mechanical creature (*automaton*) has been entrusted with the task of substituting a human being in subordinate labor duties. This concept was introduced by the Czech playwright Karel Čapek who wrote the play *Rossum's Universal Robots (R.U.R.)* in 1920. On that occasion he coined the term *robot* — derived from the term

robota that means executive labour in Slav languages — to denote the automaton built by Rossum who ends up by rising up against humankind in the science fiction tale.

In the subsequent years, in view of the development of science fiction, the behaviour conceived for the robot has often been conditioned by feelings. This has contributed to rendering the robot more and more similar to its creator.

It is worth noticing how Rossum's robots were represented as creatures made with organic material. The image of the robot as a mechanical artifact starts in the 1940s when the Russian Isaac Asimov, the well-known science fiction writer, conceived the robot as an automaton of human appearance but devoid of feelings. Its behaviour was dictated by a "positronic" brain programmed by a human being in such a way as to satisfy certain rules of ethical conduct. The term *robotics* was then introduced by Asimov as the science devoted to the study of robots which was based on the *three fundamental laws*:

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey the orders given by human beings, except when such orders would conflict with the first law.
3. A robot must protect its own existence, as long as such protection does not conflict with the first or second law.

These laws established rules of behaviour to consider as specifications for the design of a robot, which since then has attained the connotation of an industrial product designed by engineers or specialized technicians.

Science fiction has influenced the man and the woman in the street that continue to imagine the robot as a humanoid who can speak, walk, see, and hear, with an appearance very much like that presented by the robots of the movie *Metropolis*, a precursor of modern cinematography on robots, with *Star Wars* and more recently with *I, Robot* inspired by Asimov's novels.

According to a scientific interpretation of the science-fiction scenario, the robot is seen as a machine that, independently of its exterior, is able to modify the environment in which it operates. This is accomplished by carrying out actions that are conditioned by certain rules of behaviour intrinsic in the machine as well as by some data the robot acquires on its status and on the environment. In fact, *robotics* is commonly defined as the science studying the *intelligent connection between perception and action*.

With reference to this definition, a *robotic system* is in reality a complex system, functionally represented by multiple subsystems (Fig. 1.1).

The essential component of a robot is the *mechanical system* endowed, in general, with a locomotion apparatus (wheels, crawlers, mechanical legs) and a manipulation apparatus (mechanical arms, end-effectors, artificial hands). As an example, the mechanical system in Fig. 1.1 consists of two mechanical arms (manipulation apparatus), each of which is carried by a mobile vehicle

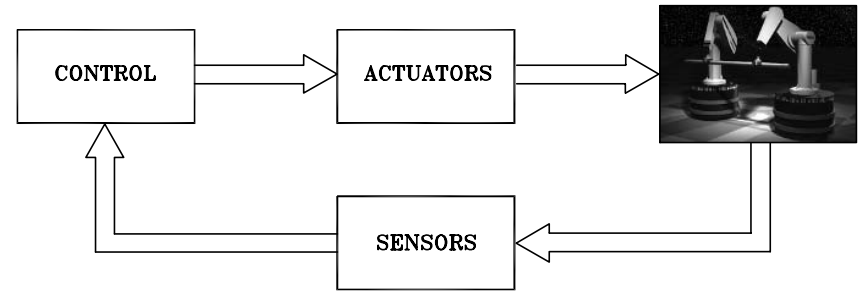

Fig. 1.1. Components of a robotic system

(locomotion apparatus). The realization of such a system refers to the context of design of articulated mechanical systems and choice of materials.

The capability to exert an action, both locomotion and manipulation, is provided by an *actuation system* which animates the mechanical components of the robot. The concept of such a system refers to the context of *motion control*, dealing with *servomotors*, *drives* and *transmissions*.

The capability for perception is entrusted to a *sensory system* which can acquire data on the internal status of the mechanical system (*proprioceptive sensors*, such as position transducers) as well as on the external status of the environment (*exteroceptive sensors*, such as force sensors and cameras). The realization of such a system refers to the context of materials properties, signal conditioning, data processing, and information retrieval.

The capability for connecting action to perception in an intelligent fashion is provided by a *control system* which can command the execution of the action in respect to the goals set by a task *planning* technique, as well as of the constraints imposed by the robot and the environment. The realization of such a system follows the same feedback principle devoted to *control* of human body functions, possibly exploiting the description of the robotic system's components (*modelling*). The context is that of cybernetics, dealing with control and supervision of robot motions, artificial intelligence and expert systems, the computational architecture and programming environment.

Therefore, it can be recognized that robotics is an interdisciplinary subject concerning the cultural areas of *mechanics*, *control*, *computers*, and *electronics*.

1.2 Robot Mechanical Structure

The key feature of a robot is its mechanical structure. Robots can be classified as those with a fixed base, *robot manipulators*, and those with a mobile base,

mobile robots. In the following, the geometrical features of the two classes are presented.

1.2.1 Robot Manipulators

The mechanical structure of a *robot manipulator* consists of a sequence of rigid bodies (*links*) interconnected by means of articulations (*joints*); a manipulator is characterized by an *arm* that ensures mobility, a *wrist* that confers dexterity, and an *end-effector* that performs the task required of the robot.

The fundamental structure of a manipulator is the serial or *open kinematic chain*. From a topological viewpoint, a kinematic chain is termed open when there is only one sequence of links connecting the two ends of the chain. Alternatively, a manipulator contains a *closed kinematic chain* when a sequence of links forms a loop.

A manipulator's mobility is ensured by the presence of joints. The articulation between two consecutive links can be realized by means of either a *prismatic* or a *revolute* joint. In an open kinematic chain, each prismatic or revolute joint provides the structure with a single degree of freedom (DOF). A prismatic joint creates a relative translational motion between the two links, whereas a revolute joint creates a relative rotational motion between the two links. Revolute joints are usually preferred to prismatic joints in view of their compactness and reliability. On the other hand, in a closed kinematic chain, the number of DOFs is less than the number of joints in view of the constraints imposed by the loop.

The *degrees of freedom* should be properly distributed along the mechanical structure in order to have a sufficient number to execute a given task. In the most general case of a task consisting of arbitrarily positioning and orienting an object in three-dimensional (3D) space, *six* DOFs are required, three for positioning a point on the object and three for orienting the object with respect to a reference coordinate frame. If more DOFs than task variables are available, the manipulator is said to be *redundant* from a kinematic viewpoint.

The *workspace* represents that portion of the environment the manipulator's end-effector can access. Its shape and volume depend on the manipulator structure as well as on the presence of mechanical joint limits.

The task required of the arm is to position the wrist which then is required to orient the end-effector. The type and sequence of the arm's DOFs, starting from the base joint, allows a classification of manipulators as *Cartesian*, *cylindrical*, *spherical*, *SCARA*, and *anthropomorphic*.

Cartesian geometry is realized by three prismatic joints whose axes typically are mutually orthogonal (Fig. 1.2). In view of the simple geometry, each DOF corresponds to a Cartesian space variable and thus it is natural to perform straight motions in space. The Cartesian structure offers very good mechanical stiffness. Wrist positioning accuracy is constant everywhere in the workspace. This is the volume enclosed by a rectangular parallel-piped

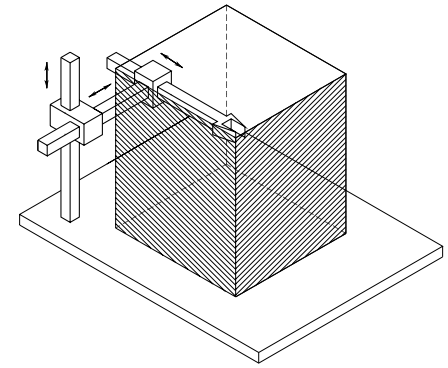

Fig. 1.2. Cartesian manipulator and its workspace

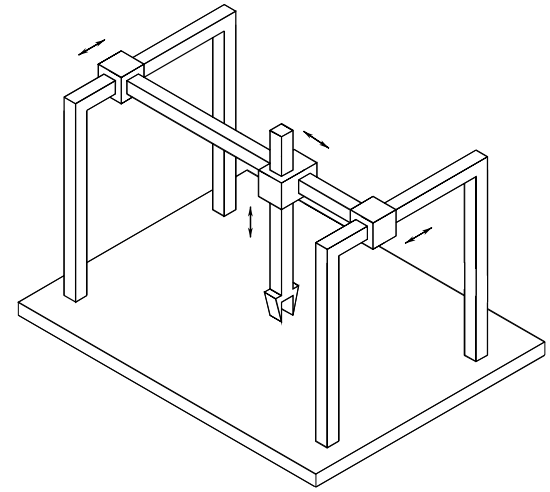

Fig. 1.3. Gantry manipulator

(Fig. 1.2). As opposed to high accuracy, the structure has low dexterity since all the joints are prismatic. The direction of approach in order to manipulate an object is from the side. On the other hand, if it is desired to approach an object from the top, the Cartesian manipulator can be realized by a *gantry* structure as illustrated in Fig. 1.3. Such a structure makes available a workspace with a large volume and enables the manipulation of objects of large dimensions and heavy weight. Cartesian manipulators are employed for material handling and assembly. The motors actuating the joints of a Cartesian manipulator are typically electric and occasionally pneumatic.

Cylindrical geometry differs from Cartesian in that the first prismatic joint is replaced with a revolute joint (Fig. 1.4). If the task is described in cylindri-

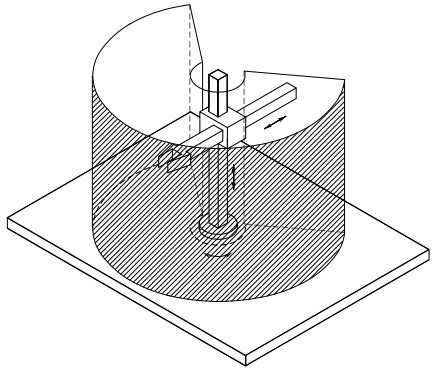

Fig. 1.4. Cylindrical manipulator and its workspace

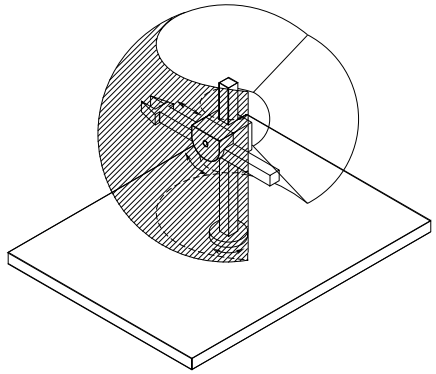

Fig. 1.5. Spherical manipulator and its workspace

cal coordinates, in this case each DOF also corresponds to a Cartesian space variable. The cylindrical structure offers good mechanical stiffness. Wrist positioning accuracy decreases as the horizontal stroke increases. The workspace is a portion of a hollow cylinder (Fig. 1.4). The horizontal prismatic joint makes the wrist of a cylindrical manipulator suitable to access horizontal cavities. Cylindrical manipulators are mainly employed for carrying objects even of large dimensions; in such a case the use of hydraulic motors is to be preferred to that of electric motors.

Spherical geometry differs from cylindrical in that the second prismatic joint is replaced with a revolute joint (Fig. 1.5). Each DOF corresponds to a Cartesian space variable provided that the task is described in spherical coordinates. Mechanical stiffness is lower than the above two geometries and mechanical construction is more complex. Wrist positioning accuracy decreases as the radial stroke increases. The workspace is a portion of a hollow sphere (Fig. 1.5); it can also include the supporting base of the manipulator and thus

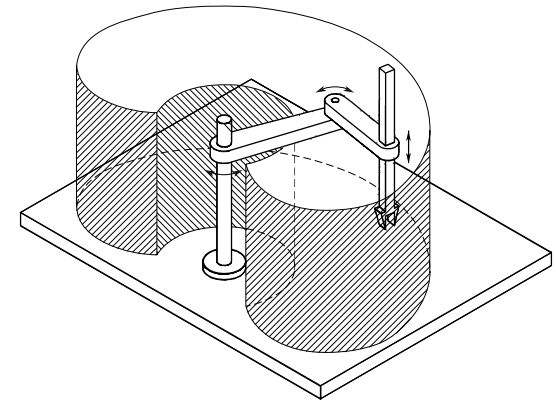

Fig. 1.6. SCARA manipulator and its workspace

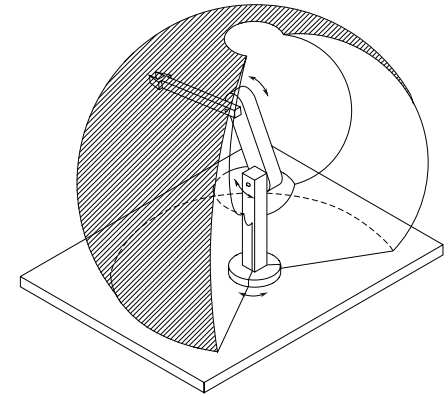

Fig. 1.7. Anthropomorphic manipulator and its workspace

it can allow manipulation of objects on the floor. Spherical manipulators are mainly employed for machining. Electric motors are typically used to actuate the joints.

A special geometry is *SCARA* geometry that can be realized by disposing two revolute joints and one prismatic joint in such a way that all the axes of motion are parallel (Fig. 1.6). The acronym SCARA stands for *Selective Compliance Assembly Robot Arm* and characterizes the mechanical features of a structure offering high stiffness to vertical loads and compliance to horizontal loads. As such, the SCARA structure is well-suited to vertical assembly tasks. The correspondence between the DOFs and Cartesian space variables is maintained only for the vertical component of a task described in Cartesian coordinates. Wrist positioning accuracy decreases as the distance of the wrist from the first joint axis increases. The typical workspace is illustrated

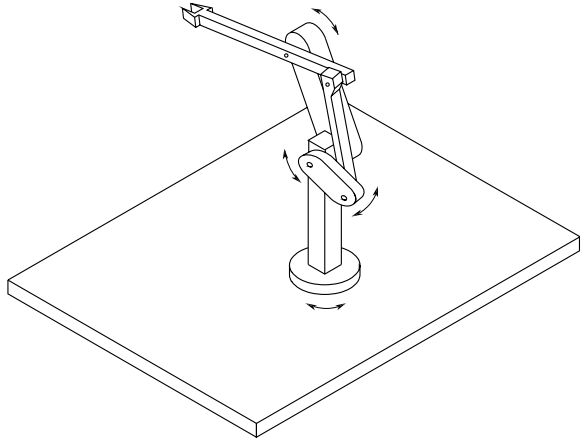

Fig. 1.8. Manipulator with parallelogram

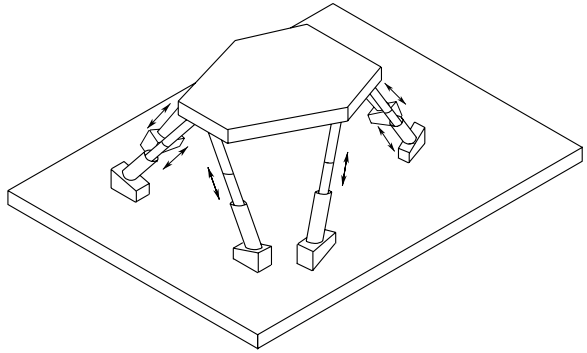

Fig. 1.9. Parallel manipulator

in Fig. 1.6. The SCARA manipulator is suitable for manipulation of small objects; joints are actuated by electric motors.

Anthropomorphic geometry is realized by three revolute joints; the revolute axis of the first joint is orthogonal to the axes of the other two which are parallel (Fig. 1.7). By virtue of its similarity with the human arm, the second joint is called the shoulder joint and the third joint the elbow joint since it connects the "arm" with the "forearm." The anthropomorphic structure is the most dexterous one, since all the joints are revolute. On the other hand, the correspondence between the DOFs and the Cartesian space variables is lost, and wrist positioning accuracy varies inside the workspace. This is approximately a portion of a sphere (Fig. 1.7) and its volume is large compared to manipulator encumbrance. Joints are typically actuated by electric motors. The range of industrial applications of anthropomorphic manipulators is wide.

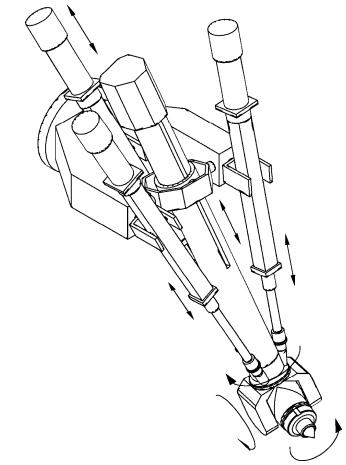

Fig. 1.10. Hybrid parallel-serial manipulator

According to the latest report by the *International Federation of Robotics* (IFR), up to 2005, 59% of installed robot manipulators worldwide has anthropomorphic geometry, 20% has Cartesian geometry, 12% has cylindrical geometry, and 8% has SCARA geometry.

All the previous manipulators have an open kinematic chain. Whenever larger payloads are required, the mechanical structure will have higher stiffness to guarantee comparable positioning accuracy. In such a case, resorting to a closed kinematic chain is advised. For instance, for an anthropomorphic structure, parallelogram geometry between the shoulder and elbow joints can be adopted, so as to create a closed kinematic chain (Fig. 1.8).

An interesting closed-chain geometry is *parallel* geometry (Fig. 1.9) which has multiple kinematic chains connecting the base to the end-effector. The fundamental advantage is seen in the high structural stiffness, with respect to open-chain manipulators, and thus the possibility to achieve high operational speeds; the drawback is that of having a reduced workspace.

The geometry illustrated in Fig. 1.10 is of hybrid type, since it consists of a parallel arm and a serial kinematic chain. This structure is suitable for the execution of manipulation tasks requiring large values of force along the vertical direction.

The manipulator structures presented above are required to position the wrist which is then required to orient the manipulator's end-effector. If arbitrary orientation in 3D space is desired, the wrist must possess at least three DOFs provided by revolute joints. Since the wrist constitutes the terminal part of the manipulator, it has to be compact; this often complicates its mechanical design. Without entering into construction details, the realization endowing the wrist with the highest dexterity is one where the three revolute

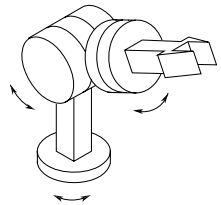

Fig. 1.11. Spherical wrist

axes intersect at a single point. In such a case, the wrist is called a *spherical wrist*, as represented in Fig. 1.11. The key feature of a spherical wrist is the decoupling between position and orientation of the end-effector; the arm is entrusted with the task of positioning the above point of intersection, whereas the wrist determines the end-effector orientation. Those realizations where the wrist is not spherical are simpler from a mechanical viewpoint, but position and orientation are coupled, and this complicates the coordination between the motion of the arm and that of the wrist to perform a given task.

The *end-effector* is specified according to the task the robot should execute. For material handling tasks, the end-effector consists of a gripper of proper shape and dimensions determined by the object to be grasped (Fig. 1.11). For machining and assembly tasks, the end-effector is a tool or a specialized device, e.g., a welding torch, a spray gun, a mill, a drill, or a screwdriver.

The versatility and flexibility of a robot manipulator should not induce the conviction that all mechanical structures are equivalent for the execution of a given task. The choice of a robot is indeed conditioned by the application which sets constraints on the workspace dimensions and shape, the maximum payload, positioning accuracy, and dynamic performance of the manipulator.

1.2.2 Mobile Robots

The main feature of *mobile robots* is the presence of a mobile base which allows the robot to move freely in the environment. Unlike manipulators, such robots are mostly used in service applications, where extensive, autonomous motion capabilities are required. From a mechanical viewpoint, a mobile robot consists of one or more rigid bodies equipped with a *locomotion* system. This description includes the following two main classes of mobile robots:[1]

- *Wheeled* mobile robots typically consist of a rigid body (*base* or *chassis*) and a system of wheels which provide motion with respect to the ground.

[1] Other types of mechanical locomotion systems are not considered here. Among these, it is worth mentioning *tracked locomotion*, very effective on uneven terrain, and *undulatory locomotion*, inspired by snake gaits, which can be achieved without specific devices. There also exist types of locomotion that are not constrained to the ground, such as flying and navigation.

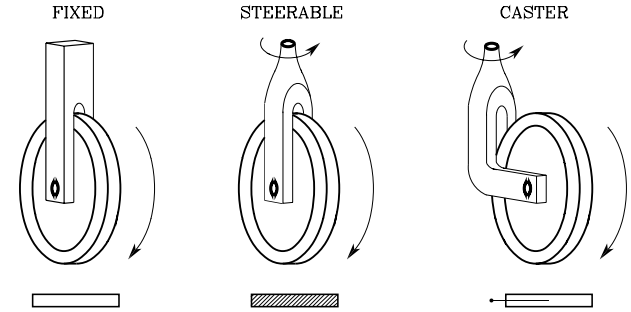

Fig. 1.12. The three types of conventional wheels with their respective icons

Other rigid bodies (*trailers*), also equipped with wheels, may be connected to the base by means of revolute joints.

- *Legged* mobile robots are made of multiple rigid bodies, interconnected by prismatic joints or, more often, by revolute joints. Some of these bodies form lower limbs, whose extremities (*feet*) periodically come in contact with the ground to realize locomotion. There is a large variety of mechanical structures in this class, whose design is often inspired by the study of living organisms (*biomimetic robotics*): they range from biped humanoids to hexapod robots aimed at replicating the biomechanical efficiency of insects.

Only wheeled vehicles are considered in the following, as they represent the vast majority of mobile robots actually used in applications. The basic mechanical element of such robots is indeed the wheel. Three types of conventional wheels exist, which are shown in Fig. 1.12 together with the icons that will be used to represent them:

- The *fixed wheel* can rotate about an axis that goes through the center of the wheel and is orthogonal to the wheel plane. The wheel is rigidly attached to the chassis, whose orientation with respect to the wheel is therefore constant.
- The *steerable wheel* has two axes of rotation. The first is the same as a fixed wheel, while the second is vertical and goes through the center of the wheel. This allows the wheel to change its orientation with respect to the chassis.
- The *caster wheel* has two axes of rotation, but the vertical axis does not pass through the center of the wheel, from which it is displaced by a constant *offset*. Such an arrangement causes the wheel to swivel automatically, rapidly aligning with the direction of motion of the chassis. This type of wheel is therefore introduced to provide a supporting point for static balance without affecting the mobility of the base; for instance, caster wheels are commonly used in shopping carts as well as in chairs with wheels.

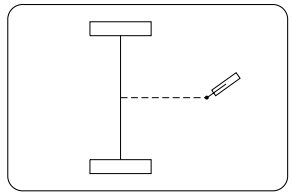

Fig. 1.13. A differential-drive mobile robot

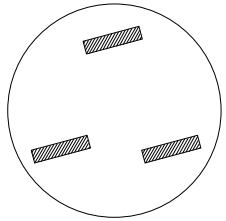

Fig. 1.14. A synchro-drive mobile robot

The variety of kinematic structures that can be obtained by combining the three conventional wheels is wide. In the following, the most relevant arrangements are briefly examined.

In a *differential-drive* vehicle there are two fixed wheels with a common axis of rotation, and one or more caster wheels, typically smaller, whose function is to keep the robot statically balanced (Fig. 1.13). The two fixed wheels are separately controlled, in that different values of angular velocity may be arbitrarily imposed, while the caster wheel is passive. Such a robot can rotate on the spot (i.e., without moving the midpoint between the wheels), provided that the angular velocities of the two wheels are equal and opposite.

A vehicle with similar mobility is obtained using a *synchro-drive* kinematic arrangement (Fig. 1.14). This robot has three aligned steerable wheels which are synchronously driven by only two motors through a mechanical coupling, e.g., a chain or a transmission belt. The first motor controls the rotation of the wheels around the horizontal axis, thus providing the driving force (traction) to the vehicle. The second motor controls the rotation of the wheels around the vertical axis, hence affecting their orientation. Note that the heading of the chassis does not change during the motion. Often, a third motor is used in this type of robot to rotate independently the upper part of the chassis (a turret) with respect to the lower part. This may be useful to orient arbitrarily a directional sensor (e.g., a camera) or in any case to recover an orientation error.

In a *tricycle* vehicle (Fig. 1.15) there are two fixed wheels mounted on a rear axle and a steerable wheel in front. The fixed wheels are driven by a single

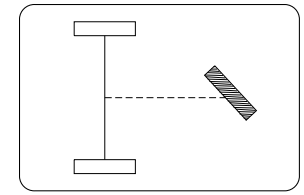

Fig. 1.15. A tricycle mobile robot

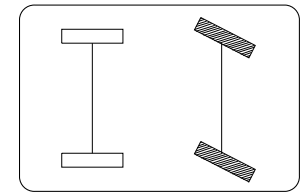

Fig. 1.16. A car-like mobile robot

motor which controls their traction,[2] while the steerable wheel is driven by another motor which changes its orientation, acting then as a steering device. Alternatively, the two rear wheels may be passive and the front wheel may provide traction as well as steering.

A *car-like* vehicle has two fixed wheels mounted on a rear axle and two steerable wheels mounted on a front axle, as shown in Fig. 1.16. As in the previous case, one motor provides (front or rear) traction while the other changes the orientation of the front wheels with respect to the vehicle. It is worth pointing out that, to avoid slippage, the two front wheels must have a different orientation when the vehicle moves along a curve; in particular, the internal wheel is slightly more steered with respect to the external one. This is guaranteed by the use of a specific device called *Ackermann steering*.

Finally, consider the robot in Fig. 1.17, which has three caster wheels usually arranged in a symmetric pattern. The traction velocities of the three wheels are independently driven. Unlike the previous cases, this vehicle is *omnidirectional*: in fact, it can move instantaneously in any Cartesian direction, as well as re-orient itself on the spot.

In addition to the above conventional wheels, there exist other special types of wheels, among which is notably the *Mecanum* (or *Swedish*) *wheel*, shown in Fig. 1.18. This is a fixed wheel with passive rollers placed along the external rim; the axis of rotation of each roller is typically inclined by 45° with respect to the plane of the wheel. A vehicle equipped with four such wheels mounted in pairs on two parallel axles is also omnidirectional.

[2] The distribution of the traction torque on the two wheels must take into account the fact that in general they move with different speeds. The mechanism which equally distributes traction is the *differential*.

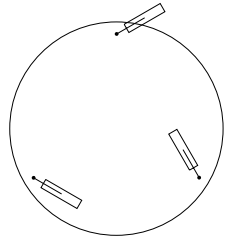

Fig. 1.17. An omnidirectional mobile robot with three independently driven caster wheels

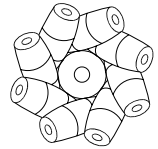

Fig. 1.18. A Mecanum (or Swedish) wheel

In the design of a wheeled robot, the mechanical balance of the structure does not represent a problem in general. In particular, a three-wheel robot is statically balanced as long as its center of mass falls inside the *support triangle*, which is defined by the contact points between the wheels and ground. Robots with more than three wheels have a support *polygon*, and thus it is typically easier to guarantee the above balance condition. It should be noted, however, that when the robot moves on uneven terrain a suspension system is needed to maintain the contact between each wheel and the ground.

Unlike the case of manipulators, the *workspace* of a mobile robot (defined as the portion of the surrounding environment that the robot can access) is potentially unlimited. Nevertheless, the local mobility of a non-omnidirectional mobile robot is always reduced; for instance, the tricycle robot in Fig. 1.15 cannot move instantaneously in a direction parallel to the rear wheel axle. Despite this fact, the tricycle can be manoeuvered so as to obtain, at the end of the motion, a net displacement in that direction. In other words, many mobile robots are subject to constraints on the admissible instantaneous motions, without actually preventing the possibility of attaining any position and orientation in the workspace. This also implies that the number of DOFs of the robot (meant as the number of admissible instantaneous motions) is lower than the number of its configuration variables.

It is obviously possible to merge the mechanical structure of a manipulator with that of a mobile vehicle by mounting the former on the latter. Such a robot is called a *mobile manipulator* and combines the dexterity of the articulated arm with the unlimited mobility of the base. An example of such a mechanical structure is shown in Fig. 1.19. However, the design of a mobile manipulator involves additional difficulties related, for instance, to the static

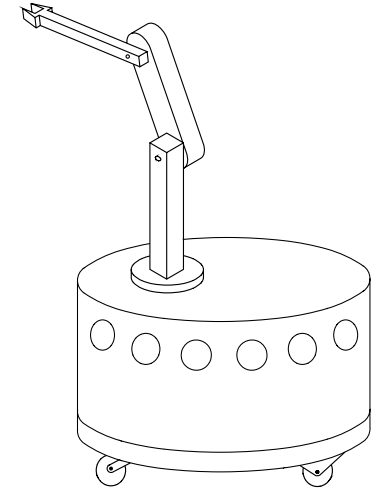

Fig. 1.19. A mobile manipulator obtained by mounting an anthropomorphic arm on a differential-drive vehicle

and dynamic mechanical balance of the robot, as well as to the actuation of the two systems.

1.3 Industrial Robotics

Industrial robotics is the discipline concerning robot design, control and applications in industry, and its products have by now reached the level of a mature technology. The connotation of a robot for industrial applications is that of operating in a *structured environment* whose geometrical or physical characteristics are mostly known a priori. Hence, limited autonomy is required.

The early industrial robots were developed in the 1960s, at the confluence of two technologies: numerical control machines for precise manufacturing, and teleoperators for remote radioactive material handling. Compared to its precursors, the first robot manipulators were characterized by:

- versatility, in view of the employment of different end-effectors at the tip of the manipulator,
- adaptability to a priori unknown situations, in view of the use of sensors,
- positioning accuracy, in view of the adoption of feedback control techniques,
- execution repeatability, in view of the programmability of various operations.

During the subsequent decades, industrial robots have gained a wide popularity as essential components for the realization of automated manufacturing

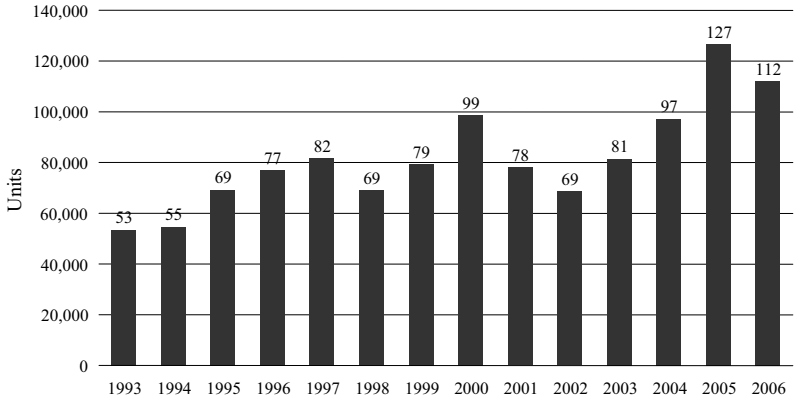

Fig. 1.20. Yearly installations of industrial robots worldwide

systems. The main factors having determined the spread of robotics technology in an increasingly wider range of applications in the manufacturing industry are reduction of manufacturing costs, increase of productivity, improvement of product quality standards and, last but not least, the possibility of eliminating harmful or off-putting tasks for the human operator in a manufacturing system.

By its usual meaning, the term *automation* denotes a technology aimed at replacing human beings with machines in a manufacturing process, as regards not only the execution of physical operations but also the intelligent processing of information on the status of the process. Automation is then the synthesis of industrial technologies typical of the manufacturing process and computer technology allowing information management. The three levels of automation one may refer to are rigid automation, programmable automation, and flexible automation.

Rigid automation deals with a factory context oriented to the mass manufacture of products of the same type. The need to manufacture large numbers of parts with high productivity and quality standards demands the use of fixed operational sequences to be executed on the workpiece by special purpose machines.

Programmable automation deals with a factory context oriented to the manufacture of low-to-medium batches of products of different types. A programmable automated system permits changing easy the sequence of operations to be executed on the workpieces in order to vary the range of products. The machines employed are more versatile and are capable of manufacturing different objects belonging to the same group technology. The majority of the products available on the market today are manufactured by programmable automated systems.

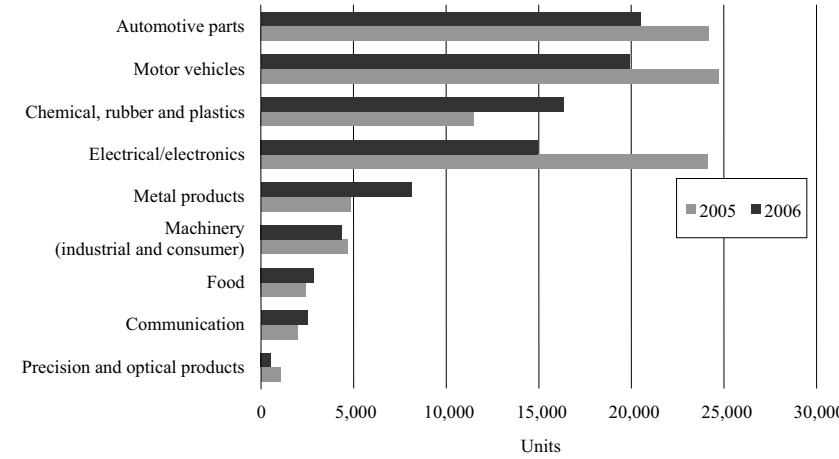

Fig. 1.21. Yearly supply of industrial robots by main industries

Flexible automation represents the evolution of programmable automation. Its goal is to allow manufacturing of variable batches of different products by minimizing the time lost for reprogramming the sequence of operations and the machines employed to pass from one batch to the next. The realization of a flexible manufacturing system (FMS) demands strong integration of computer technology with industrial technology.

The *industrial robot* is a machine with significant characteristics of versatility and flexibility. According to the widely accepted definition of the Robot Institute of America, *a robot is a reprogrammable multifunctional manipulator designed to move materials, parts, tools or specialized devices through variable programmed motions for the performance of a variety of tasks*. Such a definition, dating back to 1980, reflects the current status of robotics technology.

By virtue of its programmability, the industrial robot is a typical component of programmable automated systems. Nonetheless, robots can be entrusted with tasks in both rigid and flexible automated systems.

According to the above-mentioned IFR report, up to 2006 nearly one million industrial robots are in use worldwide, half of which are in Asia, one third in Europe, and 16% in North America. The four countries with the largest number of robots are Japan, Germany, United States and Italy. The figures for robot installations in the last 15 years are summarized in the graph in Fig. 1.20; by the end of 2007, an increase of 10% in sales with respect to the previous year is foreseen, with milder increase rates in the following years, reaching a worldwide figure of 1,200,000 units at work by the end of 2010.

In the same report it is shown how the average service life of an industrial robot is about 12 years, which may increase to 15 in a few years from now. An interesting statistic is robot density based on the total number of persons employed: this ranges from 349 robots in operation per 10,000 workers to

Fig. 1.22. Examples of AGVs for material handling (courtesy of E&K Automation GmbH)

187 in Korea, 186 in Germany, and 13 in Italy. The United States has just 99 robots per 10,000 workers. The average cost of a 6-axis industrial robot, including the control unit and development software, ranges from 20,000 to 60,000 euros, depending on the size and applications.

The automotive industry is still the predominant user of industrial robots. The graph in Fig. 1.21 referring to 2005 and 2006, however, reveals how both the chemical industry and the electrical/electronics industry are gaining in importance, and new industrial applications, such as metal products, constitute an area with a high potential investment.

Industrial robots present three fundamental capacities that make them useful for a manufacturing process: *material handling*, *manipulation*, and *measurement*.

In a manufacturing process, each object has to be transferred from one location in the factory to another in order to be stored, manufactured, assembled, and packed. During transfer, the physical characteristics of the object do not undergo any alteration. The robot's capability to pick up an object, move it in space on predefined paths and release it makes the robot itself an ideal candidate for material handling operations. Typical applications include:

- palletizing (placing objects on a pallet in an ordered way),
- warehouse loading and unloading,
- mill and machine tool tending,
- part sorting,
- packaging.

In these applications, besides robots, *Automated Guided Vehicles* (AGV) are utilized which ensure handling of parts and tools around the shop floor

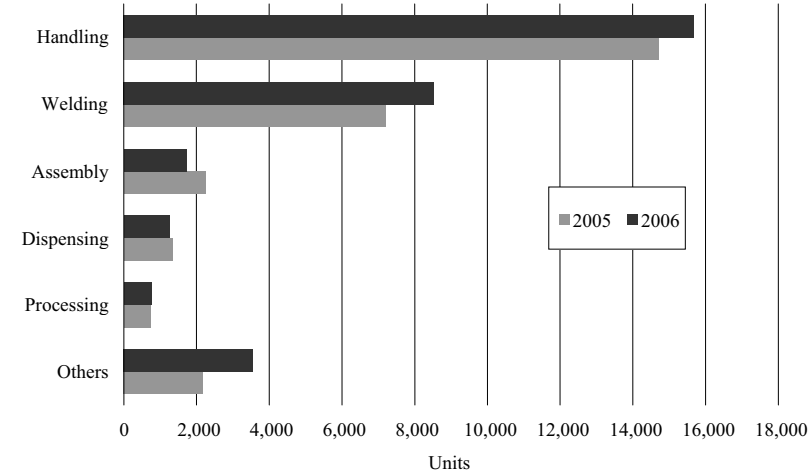

Fig. 1.23. Yearly supply of industrial robots in Europe for manufacturing operations

from one manufacturing cell to the next (Fig. 1.22). As compared to the traditional fixed guide paths for vehicles (inductive guide wire, magnetic tape, or optical visible line), modern AGVs utilize high-tech systems with onboard microprocessors and sensors (laser, odometry, GPS) which allow their localization within the plant layout, and manage their work flow and functions, allowing their complete integration in the FMS. The mobile robots employed in advanced applications can be considered as the natural evolution of the AGV systems, as far as enhanced autonomy is concerned.

Manufacturing consists of transforming objects from raw material into finished products; during this process, the part either changes its own physical characteristics as a result of machining, or loses its identity as a result of an assembly of more parts. The robot's capability to manipulate both objects and tools make it suitable to be employed in manufacturing. Typical applications include:

- arc and spot welding,
- painting and coating,
- gluing and sealing,
- laser and water jet cutting,
- milling and drilling,
- casting and die spraying,
- deburring and grinding,
- screwing, wiring and fastening,
- assembly of mechanical and electrical groups,
- assembly of electronic boards.

Fig. 1.24. The AdeptOne XL robot (courtesy of Adept Technology Inc)

Besides material handling and manipulation, in a manufacturing process it is necessary to perform measurements to test product quality. The robot's capability to explore 3D space together with the availability of measurements on the manipulator's status allow a robot to be used as a measuring device. Typical applications include:

- object inspection,
- contour finding,
- detection of manufacturing imperfections.

The graph in Fig. 1.23 reports the number of robots employed in Europe in 2005 and 2006 for various operations, which reveals how material handling requires twice as many robots employed for welding, whereas a limited number of robots is still employed for assembly.

In the following some industrial robots are illustrated in terms of their features and application fields.

The AdeptOne XL robot in Fig. 1.24 has a four-joint SCARA structure. Direct drive motors are employed. The maximum reach is 800 mm, with a repeatability of 0.025 mm horizontally and 0.038 mm vertically. Maximum speeds are 1200 mm/s for the prismatic joint, while they range from to 650 to 3300 deg/s for the three revolute joints. The maximum payload[3] is 12 kg. Typical industrial applications include small-parts material handling, assembly and packaging.

[3] Repeatability and payload are classical parameters found in industrial robot data sheets. The former gives a measure of the manipulator's ability to return to a previously reached position, while the latter indicates the average load to be carried at the robot's end-effector.

Fig. 1.25. The COMAU Smart NS robot (courtesy of COMAU SpA Robotica)

Fig. 1.26. The ABB IRB 4400 robot (courtesy of ABB Robotics)

The Comau SMART NS robot in Fig. 1.25 has a six-joint anthropomorphic structure with spherical wrist. In its four versions, the outreach ranges from 1650 and 1850 mm horizontally, with a repeatability of 0.05 mm. Maximum speeds range from 155 to 170 deg/s for the inner three joints, and from 350 to 550 deg/s for the outer three joints. The maximum payload is 16 kg. Both floor and ceiling mounting positions are allowed. Typical industrial applications include arc welding, light handling, assembly and technological processes.

The ABB IRB 4400 robot in Fig. 1.26 also has a six-joint anthropomorphic structure, but unlike the previous open-chain structure, it possesses a closed chain of parallelogram type between the shoulder and elbow joints. The outreach ranges from 1960 to 2550 mm for the various versions, with a

Fig. 1.27. The KUKA KR 60 Jet robot (courtesy of KUKA Roboter GmbH)

repeatability from 0.07 to 0.1 mm. The maximum speed at the end-effector is 2200 mm/s. The maximum payload is 60 kg. Floor or shelf-mounting is available. Typical industrial applications include material handling, machine tending, grinding, gluing, casting, die spraying and assembly.

The KUKA KR 60 Jet robot in Fig. 1.27 is composed of a five-axis structure, mounted on a sliding track with a gantry-type installation; the upright installation is also available. The linear unit has a stroke from a minimum of 400 mm to a maximum of 20 m (depending on customer's request), and a maximum speed of 3200 mm/s. On the other hand, the robot has a payload of 60 kg, an outreach of 820 mm and a repeatability of 0.15 mm. Maximum speeds are 120 deg/s and 166 deg/s for the first two joints, while they range from 260 to 322 deg/s for the outer three joints. Typical industrial applications include machine tending, arc welding, deburring, coating, sealing, plasma and waterjet cutting.

The ABB IRB340 FlexPicker robot in Fig. 1.28 adopts a parallel geometry with four axes; in view of its reduced weight and floor mounting, the robot can transport 150 objects a minute (cycle time of just 0.4 s), reaching record speeds of 10 m/s and accelerations of 100 m/s^2, for a payload of 1 kg, with a repeatability of 0.1 mm. In its 'clean' aluminum version, it is particularly suitable for packaging in the food and pharmaceutical industries.

The Fanuc M-16iB robot in Fig. 1.29 has a six-joint anthropomorphic structure with a spherical wrist. In its two versions, the outreach varies from 1667 to 1885 mm horizontally, with a repeatability of 0.1 mm. Maximum speeds range from 165 to 175 deg/s for the inner three joints, and from 340 to 520 deg/s for the outer three joints. Payload varies from 10 to 20 kg. The peculiarity of this robot consists of the integrated sensors in the control unit, including a servoing system based on 3D vision and a six-axis force sensor.

Fig. 1.28. The ABB IRB 340 FlexPicker robot (courtesy of ABB Robotics)

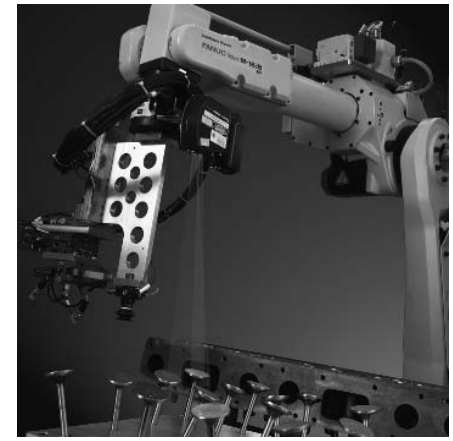

Fig. 1.29. The Fanuc M-16iB robot (courtesy of Fanuc Ltd)

The robot is utilized for handling arbitrarily located objects, deburring, sealing and waterjet cutting.

The Light Weight Robot (LWR) in Fig. 1.30 with a seven-axis structure was introduced in 2006 as the outcome of technology transfer from DLR (the German Aerospace Agency) to KUKA. In view of the adoption of lightweight materials, as well as the adoption of torque sensors at the joints, the robot can manipulate a payload of 7 to 14 kg, in the face of a weight of the structure of just 15 kg. The horizontal outreach is 868 mm, with joint speeds ranging from 110 to 210 deg/s. On the other hand, the presence of the seventh axis of motion confers kinematic redundancy to the robot, which can then be reconfigured into more dexterous postures for the execution of given tasks. Such

Fig. 1.30. The KUKA LWR robot (courtesy of KUKA Roboter GmbH)

a manipulator represents one of the most advanced industrial products and, in view of its lightweight feature, it offers interesting performance for interaction with the environment, ensuring an inherent safety in case of contact with human beings.

In most industrial applications requiring object manipulation, typical grippers are utilized as end-effectors. Nevertheless, whenever enhanced manipulability and dexterity is desired, multifingered robot hands are available.

The BarrettHand (Fig. 1.31), endowed with a fixed finger and two mobile fingers around the base of the palm, allows the manipulation of objects of different dimension, shape and orientation.

The SCHUNK Antropomorphic Hand (SAH) in Fig. 1.32 is the outcome of technology transfer from DLR and Harbin Institute of Technology (China) to SCHUNK. Characterized by three independent aligned fingers and an opposing finger which is analogous to the human thumb. The finger joints are endowed with magnetic angular sensors and torque sensors. This hand offers good dexterity and approaches the characteristics of the human hand.

LWR technology has been employed for the realization of the two arms of Justin, a humanoid manipulator made by DLR, composed of a three-joint torso with an anthropomorphic structure, two seven-axis arms and a sensorized head. The robot is illustrated in Fig. 1.33 in the execution of a bimanual manipulation task; the hands employed are previous versions of the SAH anthropomorphic hand.

The applications listed describe the current employment of robots as components of industrial automation systems. They all refer to strongly structured working environments and thus do not exhaust all the possible utilizations of robots for industrial applications. Whenever it is desired to tackle problems requiring the adaptation of the robot to a changeable working environment, the fall-out of advanced robotics products are of concern. In this regard, the

Fig. 1.31. The BarrettHand (courtesy of Barrett Technology Inc)

Fig. 1.32. The SCHUNK Anthropomorphic Hand (courtesy of SCHUNK Intec Ltd)

lightweight robot, the hands and the humanoid manipulator presented above are to be considered at the transition from traditional industrial robotics systems toward those innovative systems of advanced robotics.

1.4 Advanced Robotics

The expression *advanced robotics* usually refers to the science studying robots with marked characteristics of *autonomy*, operating in scarcely structured or *unstructured environments*, whose geometrical or physical characteristics would not be known a priori.

Nowadays, advanced robotics is still in its youth. It has indeed featured the realization of prototypes only, because the associated technology is not yet mature. There are many motivations which strongly encourage advances in knowledge within this field. They range from the need for automata whenever human operators are not available or are not safe (*field robots*), to the opportunity of developing products for potentially wide markets which are aimed at improving quality of life (*service robots*).

The graph in Fig. 1.34 reports the number of robots in stock for non-industrial applications at the end of 2006 and the forecast to 2010. Such applications are characterized by the complexity level, the uncertainty and variability of the environment with which the robot interacts, as shown in the following examples.

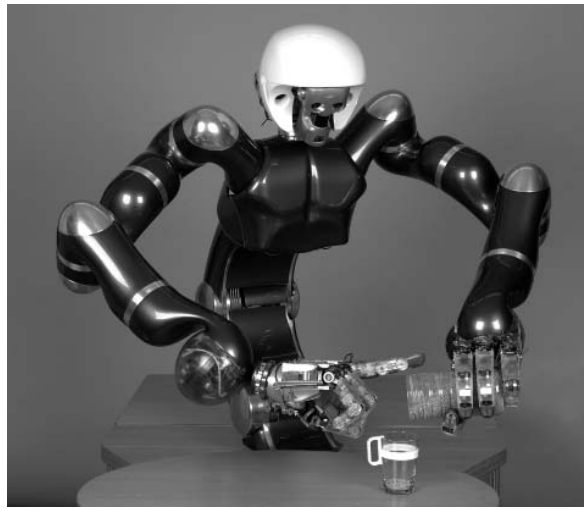

Fig. 1.33. The Justin humanoid robot manipulator (courtesy of DLR)

1.4.1 Field Robots

The context is that of deploying robots in areas where human beings could not survive or be exposed to unsustainable risks. Such robots should carry out exploration tasks and report useful data on the environment to a remote operator, using suitable onboard sensors. Typical scenarios are the exploration of a volcano, the intervention in areas contaminated by poisonous gas or radiation, or the exploration of the deep ocean or space. As is well known, NASA succeeded in delivering some mobile robots (rovers) to Mars (Fig. 1.35) which navigated on the Martian soil, across rocks, hills and crevasses. Such rovers were partially teleoperated from earth and have successfully explored the environment with sufficient autonomy. Some mini-robots were deployed on September 11, 2001 at Ground Zero after the collapse of the Twin Towers in New York, to penetrate the debris in the search for survivors.

A similar scenario is that of disasters caused by fires in tunnels or earthquakes; in such occurrences, there is a danger of further explosions, escape of harmful gases or collapse, and thus human rescue teams may cooperate with robot rescue teams. Also in the military field, unmanned autonomous aircrafts and missiles are utilized, as well as teleoperated robots with onboard cameras to explore buildings. The 'Grand Challenge' of October 2005 (Fig. 1.36) was financially supported by the US Department of Defense (DARPA) with the goal of developing autonomous vehicles to carry weapons and sensors, thus reducing soldier employment.

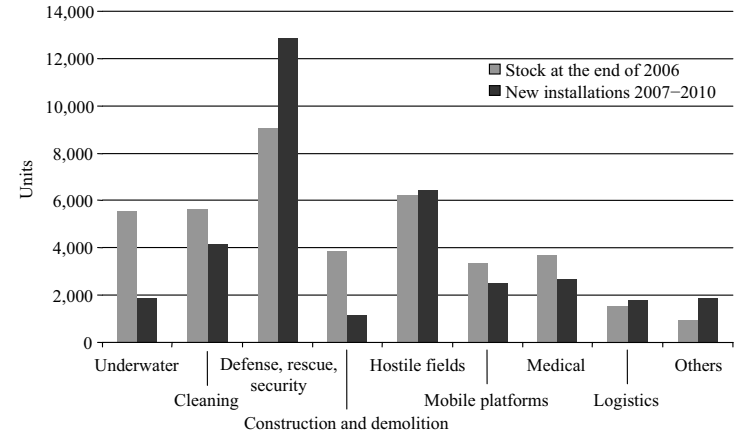

Fig. 1.34. Robots on stock for non-industrial applications

Fig. 1.35. The Sojourner rover was deployed by the Pathfinder lander and explored 250 m^2 of Martian soil in 1997 (courtesy of NASA)

1.4.2 Service Robots

Autonomous vehicles are also employed for civil applications, i.e., for mass transit systems (Fig. 1.37), thus contributing to the reduction of pollution levels. Such vehicles are part of the so-called Intelligent Transportation Systems (ITS) devoted to traffic management in urban areas. Another feasible application where the adoption of mobile robots offers potential advantages is museum guided tours (Fig. 1.38).

Many countries are investing in establishing the new market of service robots which will co-habitat with human beings in everyday life. According to the above-mentioned IFR report, up to 2005 1.9 million service robots for domestic applications (Fig. 1.39) and 1 million toy robots have been sold.

Technology is ready to transform into commercial products the prototypes of robotic aids to enhance elderly and impaired people's autonomy in everyday life; autonomous wheelchairs, mobility aid lifters, feeding aids and rehabilitation robots allowing tetraplegics to perform manual labor tasks are examples of such service devices. In perspective, other than an all-purpose robot waiter,

Fig. 1.36. The unmanned car Stanley autonomously completed a path of 132 miles in the record time of 6 h and 53 min (courtesy of DARPA)

Fig. 1.37. The Cycab is an electrically-driven vehicle for autonomous transportation in urban environments (courtesy of INRIA)

assistance, and healthcare systems integrating robotic and telematic modules will be developed for home service management (domotics).

Several robotic systems are employed for medical applications. Surgery assistance systems exploit a robot's high accuracy to position a tool, i.e., for hip prosthesis implant. Yet, in minimally-invasive surgery, i.e., cardiac surgery, the surgeon operates while seated comfortably at a console viewing a 3D image of the surgical field, and operating the surgical instruments remotely by means of a haptic interface (Fig. 1.40).

Further, in diagnostic and endoscopic surgery systems, small teleoperated robots travels through the cavities of human body, i.e., in the gastrointestinal system, bringing live images or intervening in situ for biopsy, dispensing drugs or removing neoplasms.

Fig. 1.38. Rhino, employing the synchro-drive mobile base B21 by Real World Interface, was one of the first robots for museum guided tours (courtesy of Deutsches Museum Bonn)

Fig. 1.39. The vacuum robot Roomba, employing a differential-drive kinematics, autonomously sweeps and cleans floors (courtesy of I-Robot Corp)

Finally, in motor rehabilitation systems, a hemiplegic patient wears an exoskeleton, which actively interacts, sustains and corrects the movements according to the physiotherapist's programmed plan.

Another wide market segment comes from entertainment, where robots are used as toy companions for children, and life companions for the elderly, such as humanoid robots (Fig. 1.41) and the pet robots (Fig. 1.42) being developed in Japan. It is reasonable to predict that service robots will be naturally integrated into our society. Tomorrow, robots will be as pervasive and personal as today's personal computers, or just as TV sets in the homes of 20 years ago. Robotics will then become ubiquitous, a challenge under discussion within the scientific community.

1.5 Robot Modelling, Planning and Control

In all robot applications, completion of a generic task requires the execution of a specific motion prescribed to the robot. The correct execution of such

Fig. 1.40. The da Vinci robotic system for laparoscopic surgery (courtesy of Intuitive Surgical Inc)

motion is entrusted to the control system which should provide the robot's actuators with the commands consistent with the desired motion. Motion control demands an accurate analysis of the characteristics of the mechanical structure, actuators, and sensors. The goal of such analysis is the derivation of the mathematical models describing the input/output relationship characterizing the robot components. Modelling a robot manipulator is therefore a necessary premise to finding motion control strategies.

Significant topics in the study of modelling, planning and control of robots which constitute the subject of subsequent chapters are illustrated below.

1.5.1 Modelling

Kinematic analysis of the mechanical structure of a robot concerns the description of the motion with respect to a fixed reference Cartesian frame by ignoring the forces and moments that cause motion of the structure. It is meaningful to distinguish between kinematics and differential kinematics. With reference to a robot manipulator, *kinematics* describes the analytical relationship between the joint positions and the end-effector position and orientation. *Differential kinematics* describes the analytical relationship between the joint motion and the end-effector motion in terms of velocities, through the manipulator Jacobiann.

The formulation of the kinematics relationship allows the study of two key problems of robotics, namely, the direct kinematics problem and the inverse kinematics problem. The former concerns the determination of a systematic, general method to describe the end-effector motion as a function of the joint motion by means of linear algebra tools. The latter concerns the

Fig. 1.41. The Asimo humanoid robot, launched in 1996, has been endowed with even more natural locomotion and human-robot interaction skills (courtesy of Honda Motor Company Ltd)

Fig. 1.42. The AIBO dog had been the most widely diffused entertainment robot in the recent years (courtesy of Sony Corp)

inverse problem; its solution is of fundamental importance to transform the desired motion, naturally prescribed to the end-effector in the workspace, into the corresponding joint motion.

The availability of a manipulator's kinematic model is also useful to determine the relationship between the forces and torques applied to the joints and the forces and moments applied to the end-effector in *static* equilibrium configurations.

Chapter 2 is dedicated to the study of kinematics. Chapter 3 is dedicated to the study of differential kinematics and statics, whereas Appendix A provides a useful brush-up on *linear algebra*.

Kinematics of a manipulator represents the basis of a systematic, general derivation of its *dynamics*, i.e., the equations of motion of the manipulator as a function of the forces and moments acting on it. The availability of the dynamic model is very useful for mechanical design of the structure, choice of actuators, determination of control strategies, and computer simulation of

manipulator motion. Chapter 7 is dedicated to the study of dynamics, whereas Appendix B recalls some fundamentals on *rigid body mechanics*.

Modelling of *mobile robots* requires a preliminary analysis of the kinematic constraints imposed by the presence of wheels. Depending on the mechanical structure, such constraints can be integrable or not; this has direct consequence on a robot's mobility. The *kinematic model* of a mobile robot is essentially the description of the admissible instantaneous motions in respect of the constraints. On the other hand, the *dynamic model* accounts for the reaction forces and describes the relationship between the above motions and the generalized forces acting on the robot. These models can be expressed in a canonical form which is convenient for design of planning and control techniques. Kinematic and dynamic analysis of mobile robots is developed in Chap. 11, while Appendix D contains some useful concepts of *differential geometry*.

1.5.2 Planning

With reference to the tasks assigned to a manipulator, the issue is whether to specify the motion at the joints or directly at the end-effector. In material handling tasks, it is sufficient to assign only the pick-up and release locations of an object (point-to-point motion), whereas, in machining tasks, the end-effector has to follow a desired trajectory (path motion). The goal of *trajectory planning* is to generate the timing laws for the relevant variables (joint or end-effector) starting from a concise description of the desired motion. Chapter 4 is dedicated to trajectory planning for robot manipulators.

The motion planning problem for a mobile robot concerns the generation of trajectories to take the vehicle from a given initial configuration to a desired final configuration. Such a problem is more complex than that of robot manipulators, since trajectories have to be generated in respect of the kinematic constraints imposed by the wheels. Some solution techniques are presented in Chap. 11, which exploit the specific differential structure of the mobile robots' kinematic models.

Whenever obstacles are present in a mobile robot's workspace, the planned motions must be safe, so as to avoid collisions. Such a problem, known as *motion planning*, can be formulated in an effective fashion for both robot manipulators and mobile robots utilizing the configuration space concept. The solution techniques are essentially of algorithmic nature and include exact, probabilistic and heuristic methods. Chapter 12 is dedicated to motion planning problem, while Appendix E provides some basic concepts on *graph search algorithms*.

1.5.3 Control

Realization of the motion specified by the control law requires the employment of *actuators* and *sensors*. The functional characteristics of the most commonly used actuators and sensors for robots are described in Chap. 5.

Chapter 6 is concerned with the hardware/software *architecture* of a robot's *control system* which is in charge of implementation of control laws as well as of interface with the operator.

The trajectories generated constitute the reference inputs to the *motion control* system of the mechanical structure. The problem of *robot manipulator* control is to find the time behaviour of the forces and torques to be delivered by the joint actuators so as to ensure the execution of the reference trajectories. This problem is quite complex, since a manipulator is an articulated system and, as such, the motion of one link influences the motion of the others. Manipulator equations of motion indeed reveal the presence of coupling dynamic effects among the joints, except in the case of a Cartesian structure with mutually orthogonal axes. The synthesis of the joint forces and torques cannot be made on the basis of the sole knowledge of the dynamic model, since this does not completely describe the real structure. Therefore, manipulator control is entrusted to the closure of feedback loops; by computing the deviation between the reference inputs and the data provided by the proprioceptive sensors, a feedback control system is capable of satisfying accuracy requirements on the execution of the prescribed trajectories.

Chapter 8 is dedicated to the presentation of motion control techniques, whereas Appendix C illustrates the basic principles of *feedback control*.

Control of a *mobile robot* substantially differs from the analogous problem for robot manipulators. This is due, in turn, to the availability of fewer control inputs than the robot has configuration variables. An important consequence is that the structure of a controller allowing a robot to follow a trajectory (tracking problem) is unavoidably different from that of a controller aimed at taking the robot to a given configuration (regulation problem). Further, since a mobile robot's proprioceptive sensors do not yield any data on the vehicle's configuration, it is necessary to develop localization methods for the robot in the environment. The control design problem for wheeled mobile robots is treated in Chap. 11.

If a manipulation task requires interaction between the robot and the environment, the control problem should account for the data provided by the exteroceptive sensors; the forces exchanged at the contact with the environment, and the objects' position as detected by suitable cameras. Chapter 9 is dedicated to *force control* techniques for robot manipulators, while Chap. 10 presents *visual control* techniques.

Bibliography

In the last 30 years, the robotics field has stimulated the interest of an increasing number of scholars. A truly respectable international research community has been established. Literature production has been conspicuous, both in terms of textbooks and scientific monographs and in terms of journals dedicated to robotics. Therefore, it seems appropriate to close this introduction

by offering a selection of *bibliographical reference sources* to those readers who wish to make a thorough study of robotics.

Besides indicating those basic textbooks sharing an affinity of contents with this one, the following lists include specialized books on related subjects, collections of contributions on the state of the art of research, scientific journals, and series of international conferences.

Basic textbooks

- J. Angeles, *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*, Springer-Verlag, New York, 1997.
- H. Asada, J.-J.E. Slotine, *Robot Analysis and Control*, Wiley, New York, 1986.
- G.A. Bekey, *Autonomous Robots*, MIT Press, Cambridge, MA, 2005.
- C. Canudas de Wit, B. Siciliano, G. Bastin, (Eds.), *Theory of Robot Control*, Springer-Verlag, London, 1996.
- J.J. Craig, *Introduction to Robotics: Mechanics and Control*, 3rd ed., Pearson Prentice Hall, Upper Saddle River, NJ, 2004.
- A.J. Critchlow, *Introduction to Robotics*, Macmillan, New York, 1985.
- J.F. Engelberger, *Robotics in Practice*, Amacom, New York, 1980.
- J.F. Engelberger, *Robotics in Service*, MIT Press, Cambridge, MA, 1989.
- K.S. Fu, R.C. Gonzalez, C.S.G. Lee, *Robotics: Control, Sensing, Vision, and Intelligence*, McGraw-Hill, New York, 1987.
- W. Khalil, E. Dombre, *Modeling, Identification and Control of Robots*, Hermes Penton Ltd, London, 2002.
- A.J. Koivo, *Fundamentals for Control of Robotic Manipulators*, Wiley, New York, 1989.
- Y. Koren, *Robotics for Engineers*, McGraw-Hill, New York, 1985.
- F.L. Lewis, C.T. Abdallah, D.M. Dawson, *Control of Robot Manipulators*, Macmillan, New York, 1993.
- P.J. McKerrow, *Introduction to Robotics*, Addison-Wesley, Sydney, Australia, 1991.
- R.M. Murray, Z. Li, S.S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, Boca Raton, FL, 1994.
- S.B. Niku, *Introduction to Robotics: Analysis, Systems, Applications*, Prentice-Hall, Upper Saddle River, NJ, 2001.
- R.P. Paul, *Robot Manipulators: Mathematics, Programming, and Control* MIT Press, Cambridge, MA, 1981.
- R.J. Schilling, *Fundamentals of Robotics: Analysis and Control*, Prentice-Hall, Englewood Cliffs, NJ, 1990.
- L. Sciavicco, B. Siciliano, *Modelling and Control of Robot Manipulators*, 2nd ed., Springer, London, UK, 2000.
- W.E. Snyder, *Industrial Robots: Computer Interfacing and Control*, Prentice-Hall, Englewood Cliffs, NJ, 1985.

- M.W. Spong, S. Hutchinson, M. Vidyasagar, *Robot Modeling and Control*, Wiley, New York, 2006.
- M. Vukobratović, *Introduction to Robotics*, Springer-Verlag, Berlin, Germany, 1989.
- T. Yoshikawa, *Foundations of Robotics*, MIT Press, Boston, MA, 1990.

Specialized books

Topics of related interest to robot modelling, planning and control are:

- manipulator mechanical design,
- manipulation tools,
- manipulators with elastic members,
- parallel robots,
- locomotion apparatus,
- mobile robots,
- underwater and space robots,
- control architectures
- motion and force control,
- robot vision,
- multisensory data fusion,
- telerobotics,
- human-robot interaction.

The following books are dedicated to these topics:

- G. Antonelli, *Underwater Robots: Motion and Force Control of Vehicle-Manipulator Systems*, 2nd ed., Springer, Heidelberg, Germany, 2006.
- R.C. Arkin, *Behavior-Based Robotics*, MIT Press, Cambridge, MA, 1998.
- J. Baeten, J. De Schutter, *Integrated Visual Servoing and Force Control: The Task Frame Approach*, Springer, Heidelberg, Germany, 2003.
- M. Buehler, K. Iagnemma, S. Singh, (Eds.), *The 2005 DARPA Grand Challenge: The Great Robot Race*, Springer, Heidelberg, Germany, 2007.
- J.F. Canny, *The Complexity of Robot Motion Planning*, MIT Press, Cambridge, MA, 1988.
- H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*, MIT Press, Cambridge, MA, 2005.
- P.I. Corke, *Visual Control of Robots: High-Performance Visual Servoing*, Research Studies Press, Taunton, UK, 1996.
- M.R. Cutkosky, *Robotic Grasping and Fine Manipulation*, Kluwer, Boston, MA, 1985.
- H.F. Durrant-Whyte, *Integration, Coordination and Control of Multi-Sensor Robot Systems*, Kluwer, Boston, MA, 1988.
- A. Ellery, *An Introduction to Space Robotics*, Springer-Verlag, London, UK, 2000.

- A.R. Fraser, R.W. Daniel, *Perturbation Techniques for Flexible Manipulators*, Kluwer, Boston, MA, 1991.
- B.K. Ghosh, N. Xi, T.-J. Tarn, (Eds.), *Control in Robotics and Automation: Sensor-Based Integration*, Academic Press, San Diego, CA, 1999.
- K. Goldberg, (Ed.), *The Robot in the Garden: Telerobotics and Telepistemology in the Age of the Internet*, MIT Press, Cambridge, MA, 2000.
- S. Hirose, *Biologically Inspired Robots*, Oxford University Press, Oxford, UK, 1993.
- B.K.P. Horn, *Robot Vision*, McGraw-Hill, New York, 1986.
- K. Iagnemma, S. Dubowsky, *Mobile Robots in Rough Terrain Estimation: Motion Planning, and Control with Application to Planetary Rovers Series*, Springer, Heidelberg, Germany, 2004.
- R. Kelly, V. Santibañez, A. Loría, *Control of Robot Manipulators in Joint Space*, Springer-Verlag, London, UK, 2005.
- J.-C. Latombe, *Robot Motion Planning*, Kluwer, Boston, MA, 1991.
- M.T. Mason, *Mechanics of Robotic Manipulation*, MIT Press, Cambridge, MA, 2001.
- M.T. Mason, J.K. Salisbury, *Robot Hands and the Mechanics of Manipulation*, MIT Press, Cambridge, MA, 1985.
- J.-P. Merlet, *Parallel Robots*, 2nd ed., Springer, Dordrecht, The Netherlands, 2006.
- R.R. Murphy, *Introduction to AI Robotics*, MIT Press, Cambridge, MA, 2000.
- C. Natale, *Interaction Control of Robot Manipulators: Six-degrees-of-freedom Tasks*, Springer, Heidelberg, Germany, 2003.
- M. Raibert, *Legged Robots that Balance*, MIT Press, Cambridge, MA, 1985.
- E.I. Rivin, *Mechanical Design of Robots*, McGraw-Hill, New York, 1987.
- B. Siciliano, L. Villani, *Robot Force Control*, Kluwer, Boston, MA, 2000.
- R. Siegwart, *Introduction to Autonomous Mobile Robots*, MIT Press, Cambridge, MA, 2004.
- S. Thrun, W. Burgard, D. Fox, *Probabilistic Robotics*, MIT Press, Cambridge, MA, 2005.
- D.J. Todd, *Walking Machines, an Introduction to Legged Robots*, Chapman Hall, London, UK, 1985.
- L.-W. Tsai, *Robot Analysis: The Mechanics of Serial and Parallel Manipulators*, Wiley, New York, 1999.

Edited collections on the state of the art of research

- M. Brady, (Ed.), *Robotics Science*, MIT Press, Cambridge, MA, 1989.
- M. Brady, J.M. Hollerbach, T.L. Johnson, T. Lozano-Pérez, M.T. Mason, (Eds.), *Robot Motion: Planning and Control*, MIT Press, Cambridge, MA, 1982.
- R.C. Dorf, *International Encyclopedia of Robotics*, Wiley, New York, 1988.

- V.D. Hunt, *Industrial Robotics Handbook*, Industrial Press, New York, 1983.
- O. Khatib, J.J. Craig, T. Lozano-Pérez, (Eds.), *The Robotics Review 1*, MIT Press, Cambridge, MA, 1989.
- O. Khatib, J.J. Craig, T. Lozano-Pérez, (Eds.), *The Robotics Review 2*, MIT Press, Cambridge, MA., 1992.
- T.R. Kurfess, (Ed.), *Robotics and Automation Handbook*, CRC Press, Boca Raton, FL, 2005.
- B. Siciliano, O. Khatib, (Eds.), *Springer Handbook of Robotics*, Springer, Heidelberg, Germany, 2008.
- C.S.G. Lee, R.C. Gonzalez, K.S. Fu, (Eds.), *Tutorial on Robotics*, 2nd ed., IEEE Computer Society Press, Silver Spring, MD, 1986.
- M.W. Spong, F.L. Lewis, C.T. Abdallah, (Eds.), *Robot Control: Dynamics, Motion Planning, and Analysis*, IEEE Press, New York, 1993.

Scientific journals

- Advanced Robotics
- Autonomous Robots
- IEEE Robotics and Automation Magazine
- IEEE Transactions on Robotics
- International Journal of Robotics Research
- Journal of Field Robotics
- Journal of Intelligent and Robotic Systems
- Robotica
- Robotics and Autonomous Systems

Series of international scientific conferences

- IEEE International Conference on Robotics and Automation
- IEEE/RSJ International Conference on Intelligent Robots and Systems
- International Conference on Advanced Robotics
- International Symposium of Robotics Research
- International Symposium on Experimental Robotics
- Robotics: Science and Systems

The above journals and conferences represent the reference sources for the international scientific community. Many other robotics journals and conferences exist which are devoted to specific topics, such as kinematics, control, vision, algorithms, haptics, industrial applications, space and underwater exploration, humanoid robotics, and human-robot interaction. On the other hand, several journals and prestigious conferences in other fields, such as mechanics, control, sensors, and artificial intelligence, offer generous space to robotics topics.

2

Kinematics

A *manipulator* can be schematically represented from a mechanical viewpoint as a kinematic chain of rigid bodies (*links*) connected by means of revolute or prismatic *joints*. One end of the chain is constrained to a base, while an *end-effector* is mounted to the other end. The resulting motion of the structure is obtained by composition of the elementary motions of each link with respect to the previous one. Therefore, in order to manipulate an object in space, it is necessary to describe the end-effector position and orientation. This chapter is dedicated to the derivation of the *direct kinematics equation* through a systematic, general approach based on linear algebra. This allows the end-effector position and orientation (*pose*) to be expressed as a function of the joint variables of the mechanical structure with respect to a reference frame. Both open-chain and closed-chain kinematic structures are considered. With reference to a *minimal representation of orientation*, the concept of *operational space* is introduced and its relationship with the *joint space* is established. Furthermore, a *calibration* technique of the manipulator kinematic parameters is presented. The chapter ends with the derivation of solutions to the *inverse kinematics problem*, which consists of the determination of the joint variables corresponding to a given end-effector pose.

2.1 Pose of a Rigid Body

A *rigid body* is completely described in space by its *position* and *orientation* (in brief *pose*) with respect to a reference frame. As shown in Fig. 2.1, let O–xyz be the orthonormal reference frame and $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ be the unit vectors of the frame axes.

The position of a point O' on the rigid body with respect to the coordinate frame O–xyz is expressed by the relation

$$\boldsymbol{o}' = o'_x\boldsymbol{x} + o'_y\boldsymbol{y} + o'_z\boldsymbol{z},$$

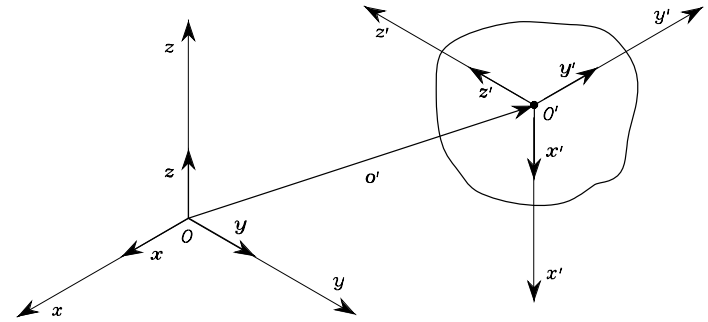

Fig. 2.1. Position and orientation of a rigid body

where o'_x, o'_y, o'_z denote the components of the vector $\boldsymbol{o}' \in \mathbb{R}^3$ along the frame axes; the position of O' can be compactly written as the (3×1) vector

$$\boldsymbol{o}' = \begin{bmatrix} o'_x \\ o'_y \\ o'_z \end{bmatrix}. \tag{2.1}$$

Vector $\boldsymbol{o}'$ is a bound vector since its line of application and point of application are both prescribed, in addition to its direction and norm.

In order to describe the rigid body orientation, it is convenient to consider an orthonormal frame attached to the body and express its unit vectors with respect to the reference frame. Let then O'–$x'y'z'$ be such a frame with origin in O' and $\boldsymbol{x}'$, $\boldsymbol{y}'$, $\boldsymbol{z}'$ be the unit vectors of the frame axes. These vectors are expressed with respect to the reference frame O–xyz by the equations:

$$\begin{aligned} \boldsymbol{x}' &= x'_x\boldsymbol{x} + x'_y\boldsymbol{y} + x'_z\boldsymbol{z} \\ \boldsymbol{y}' &= y'_x\boldsymbol{x} + y'_y\boldsymbol{y} + y'_z\boldsymbol{z} \\ \boldsymbol{z}' &= z'_x\boldsymbol{x} + z'_y\boldsymbol{y} + z'_z\boldsymbol{z}. \end{aligned} \tag{2.2}$$

The components of each unit vector are the direction cosines of the axes of frame O'–$x'y'z'$ with respect to the reference frame O–xyz.

2.2 Rotation Matrix

By adopting a compact notation, the three unit vectors in (2.2) describing the body orientation with respect to the reference frame can be combined in the (3×3) matrix

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{x}' & \boldsymbol{y}' & \boldsymbol{z}' \end{bmatrix} = \begin{bmatrix} x'_x & y'_x & z'_x \\ x'_y & y'_y & z'_y \\ x'_z & y'_z & z'_z \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}'^T\boldsymbol{x} & \boldsymbol{y}'^T\boldsymbol{x} & \boldsymbol{z}'^T\boldsymbol{x} \\ \boldsymbol{x}'^T\boldsymbol{y} & \boldsymbol{y}'^T\boldsymbol{y} & \boldsymbol{z}'^T\boldsymbol{y} \\ \boldsymbol{x}'^T\boldsymbol{z} & \boldsymbol{y}'^T\boldsymbol{z} & \boldsymbol{z}'^T\boldsymbol{z} \end{bmatrix}, \tag{2.3}$$

which is termed *rotation matrix*.

It is worth noting that the column vectors of matrix $\boldsymbol{R}$ are mutually orthogonal since they represent the unit vectors of an orthonormal frame, i.e.,

$$\boldsymbol{x}'^T\boldsymbol{y}' = 0 \qquad \boldsymbol{y}'^T\boldsymbol{z}' = 0 \qquad \boldsymbol{z}'^T\boldsymbol{x}' = 0.$$

Also, they have unit norm

$$\boldsymbol{x}'^T\boldsymbol{x}' = 1 \qquad \boldsymbol{y}'^T\boldsymbol{y}' = 1 \qquad \boldsymbol{z}'^T\boldsymbol{z}' = 1.$$

As a consequence, $\boldsymbol{R}$ is an *orthogonal* matrix meaning that

$$\boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}_3 \tag{2.4}$$

where $\boldsymbol{I}_3$ denotes the (3×3) identity matrix.

If both sides of (2.4) are postmultiplied by the inverse matrix $\boldsymbol{R}^{-1}$, the useful result is obtained:

$$\boldsymbol{R}^T = \boldsymbol{R}^{-1}, \tag{2.5}$$

that is, the transpose of the rotation matrix is equal to its inverse. Further, observe that $\det(\boldsymbol{R}) = 1$ if the frame is right-handed, while $\det(\boldsymbol{R}) = -1$ if the frame is left-handed.

The above-defined rotation matrix belongs to the *special orthonormal group* $SO(m)$ of the real $(m \times m)$ matrices with othonormal columns and determinant equal to 1; in the case of spatial rotations it is $m = 3$, whereas in the case of planar rotations it is $m = 2$.

2.2.1 Elementary Rotations

Consider the frames that can be obtained via *elementary rotations* of the reference frame about one of the coordinate axes. These rotations are positive if they are made counter-clockwise about the relative axis.

Suppose that the reference frame O–xyz is rotated by an angle α about axis z (Fig. 2.2), and let O–$x'y'z'$ be the rotated frame. The unit vectors of the new frame can be described in terms of their components with respect to the reference frame. Consider the frames that can be obtained via *elementary rotations* of the reference frame about one of the coordinate axes. These rotations are positive if they are made counter-clockwise about the relative axis.

Suppose that the reference frame O–xyz is rotated by an angle α about axis z (Fig. 2.2), and let O–$x'y'z'$ be the rotated frame. The unit vectors of the new frame can be described in terms of their components with respect to the reference frame, i.e.,

$$\boldsymbol{x}' = \begin{bmatrix} \cos\alpha \\ \sin\alpha \\ 0 \end{bmatrix} \qquad \boldsymbol{y}' = \begin{bmatrix} -\sin\alpha \\ \cos\alpha \\ 0 \end{bmatrix} \qquad \boldsymbol{z}' = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

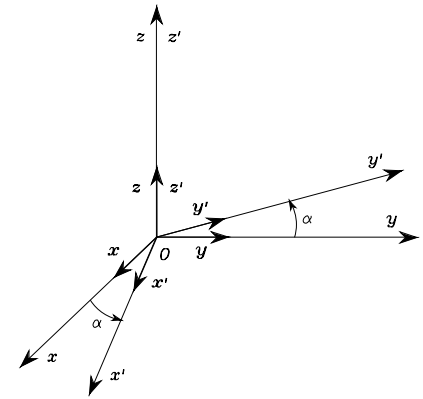

Fig. 2.2. Rotation of frame O–xyz by an angle α about axis z

Hence, the rotation matrix of frame O–$x'y'z'$ with respect to frame O–xyz is

$$\boldsymbol{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.6}$$

In a similar manner, it can be shown that the rotations by an angle β about axis y and by an angle γ about axis x are respectively given by

$$\boldsymbol{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \tag{2.7}$$

$$\boldsymbol{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}. \tag{2.8}$$

These matrices will be useful to describe rotations about an arbitrary axis in space.

It is easy to verify that for the elementary rotation matrices in (2.6)–(2.8) the following property holds:

$$\boldsymbol{R}_k(-\vartheta) = \boldsymbol{R}_k^T(\vartheta) \qquad\qquad k = x, y, z. \tag{2.9}$$

In view of (2.6)–(2.8), the rotation matrix can be attributed a geometrical meaning; namely, the matrix $\boldsymbol{R}$ describes the rotation about an axis in space needed to align the axes of the reference frame with the corresponding axes of the body frame.

2.2.2 Representation of a Vector

In order to understand a further geometrical meaning of a rotation matrix, consider the case when the origin of the body frame coincides with the origin

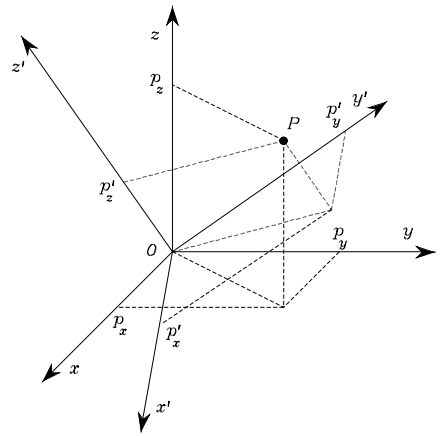

Fig. 2.3. Representation of a point P in two different coordinate frames

of the reference frame (Fig. 2.3); it follows that $\boldsymbol{o}' = \boldsymbol{0}$, where $\boldsymbol{0}$ denotes the (3×1) null vector. A point P in space can be represented either as

$$\boldsymbol{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

with respect to frame O–xyz, or as

$$\boldsymbol{p}' = \begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix}$$

with respect to frame O–$x'y'z'$.

Since $\boldsymbol{p}$ and $\boldsymbol{p}'$ are representations of the same point P, it is

$$\boldsymbol{p} = p'_x \boldsymbol{x}' + p'_y \boldsymbol{y}' + p'_z \boldsymbol{z}' = \begin{bmatrix} \boldsymbol{x}' & \boldsymbol{y}' & \boldsymbol{z}' \end{bmatrix} \boldsymbol{p}'$$

and, accounting for (2.3), it is

$$\boldsymbol{p} = \boldsymbol{R}\boldsymbol{p}'. \tag{2.10}$$

The rotation matrix $\boldsymbol{R}$ represents the *transformation matrix* of the vector coordinates in frame O–$x'y'z'$ into the coordinates of the same vector in frame O–xyz. In view of the orthogonality property (2.4), the inverse transformation is simply given by

$$\boldsymbol{p}' = \boldsymbol{R}^T \boldsymbol{p}. \tag{2.11}$$

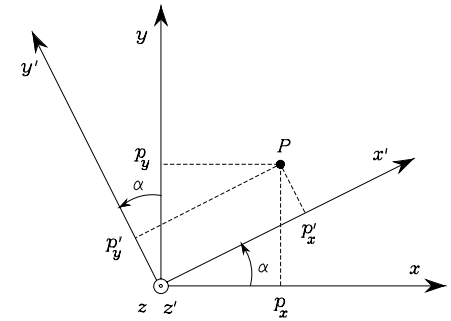

Fig. 2.4. Representation of a point P in rotated frames

Example 2.1

Consider two frames with common origin mutually rotated by an angle α about the axis z. Let $\boldsymbol{p}$ and $\boldsymbol{p}'$ be the vectors of the coordinates of a point P, expressed in the frames O–xyz and O–$x'y'z'$, respectively (Fig. 2.4). On the basis of simple geometry, the relationship between the coordinates of P in the two frames is

$$\begin{aligned} p_x &= p'_x \cos\alpha - p'_y \sin\alpha \\ p_y &= p'_x \sin\alpha + p'_y \cos\alpha \\ p_z &= p'_z. \end{aligned}$$

Therefore, the matrix (2.6) represents not only the orientation of a frame with respect to another frame, but it also describes the transformation of a vector from a frame to another frame with the same origin.

2.2.3 Rotation of a Vector

A rotation matrix can be also interpreted as the matrix operator allowing rotation of a vector by a given angle about an arbitrary axis in space. In fact, let $\boldsymbol{p}'$ be a vector in the reference frame O–xyz; in view of orthogonality of the matrix $\boldsymbol{R}$, the product $\boldsymbol{R}\boldsymbol{p}'$ yields a vector $\boldsymbol{p}$ with the same norm as that of $\boldsymbol{p}'$ but rotated with respect to $\boldsymbol{p}'$ according to the matrix $\boldsymbol{R}$. The norm equality can be proved by observing that $\boldsymbol{p}^T\boldsymbol{p} = \boldsymbol{p}'^T\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{p}'$ and applying (2.4). This interpretation of the rotation matrix will be revisited later.

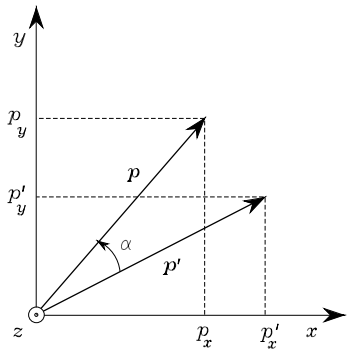

Fig. 2.5. Rotation of a vector

Example 2.2

Consider the vector $\boldsymbol{p}$ which is obtained by rotating a vector $\boldsymbol{p}'$ in the plane xy by an angle α about axis z of the reference frame (Fig. 2.5). Let (p'_x, p'_y, p'_z) be the coordinates of the vector $\boldsymbol{p}'$. The vector $\boldsymbol{p}$ has components

$$\begin{aligned} p_x &= p'_x \cos\alpha - p'_y \sin\alpha \\ p_y &= p'_x \sin\alpha + p'_y \cos\alpha \\ p_z &= p'_z. \end{aligned}$$

It is easy to recognize that $\boldsymbol{p}$ can be expressed as

$$\boldsymbol{p} = \boldsymbol{R}_z(\alpha)\boldsymbol{p}',$$

where $\boldsymbol{R}_z(\alpha)$ is the same rotation matrix as in (2.6).

In sum, a rotation matrix attains three *equivalent geometrical meanings*:

- It describes the mutual orientation between two coordinate frames; its column vectors are the direction cosines of the axes of the rotated frame with respect to the original frame.
- It represents the coordinate transformation between the coordinates of a point expressed in two different frames (with common origin).
- It is the operator that allows the rotation of a vector in the same coordinate frame.

2.3 Composition of Rotation Matrices

In order to derive composition rules of rotation matrices, it is useful to consider the expression of a vector in two different reference frames. Let then O–$x_0y_0z_0$,

O–$x_1y_1z_1$, O–$x_2y_2z_2$ be three frames with common origin O. The vector $\boldsymbol{p}$ describing the position of a generic point in space can be expressed in each of the above frames; let $\boldsymbol{p}^0$, $\boldsymbol{p}^1$, $\boldsymbol{p}^2$ denote the expressions of $\boldsymbol{p}$ in the three frames.[1]

At first, consider the relationship between the expression $\boldsymbol{p}^2$ of the vector $\boldsymbol{p}$ in Frame 2 and the expression $\boldsymbol{p}^1$ of the same vector in Frame 1. If $\boldsymbol{R}_i^j$ denotes the rotation matrix of Frame i with respect to Frame j, it is

$$\boldsymbol{p}^1 = \boldsymbol{R}_2^1 \boldsymbol{p}^2. \tag{2.12}$$

Similarly, it turns out that

$$\boldsymbol{p}^0 = \boldsymbol{R}_1^0 \boldsymbol{p}^1 \tag{2.13}$$

$$\boldsymbol{p}^0 = \boldsymbol{R}_2^0 \boldsymbol{p}^2. \tag{2.14}$$

On the other hand, substituting (2.12) in (2.13) and using (2.14) gives

$$\boldsymbol{R}_2^0 = \boldsymbol{R}_1^0 \boldsymbol{R}_2^1. \tag{2.15}$$

The relationship in (2.15) can be interpreted as the composition of successive rotations. Consider a frame initially aligned with the frame O–$x_0y_0z_0$. The rotation expressed by matrix $\boldsymbol{R}_2^0$ can be regarded as obtained in two steps:

- First rotate the given frame according to $\boldsymbol{R}_1^0$, so as to align it with frame O–$x_1y_1z_1$.
- Then rotate the frame, now aligned with frame O–$x_1y_1z_1$, according to $\boldsymbol{R}_2^1$, so as to align it with frame O–$x_2y_2z_2$.

Notice that the overall rotation can be expressed as a sequence of partial rotations; each rotation is defined with respect to the preceding one. The frame with respect to which the rotation occurs is termed *current frame*. Composition of successive rotations is then obtained by postmultiplication of the rotation matrices following the given order of rotations, as in (2.15). With the adopted notation, in view of (2.5), it is

$$\boldsymbol{R}_i^j = (\boldsymbol{R}_j^i)^{-1} = (\boldsymbol{R}_j^i)^T. \tag{2.16}$$

Successive rotations can be also specified by constantly referring them to the initial frame; in this case, the rotations are made with respect to a *fixed frame*. Let $\boldsymbol{R}_1^0$ be the rotation matrix of frame O–$x_1y_1z_1$ with respect to the fixed frame O–$x_0y_0z_0$. Let then $\bar{\boldsymbol{R}}_2^0$ denote the matrix characterizing frame O–$x_2y_2z_2$ with respect to Frame 0, which is obtained as a rotation of Frame 1 according to the matrix $\bar{\boldsymbol{R}}_2^1$. Since (2.15) gives a composition rule of successive rotations about the axes of the current frame, the overall rotation can be regarded as obtained in the following steps:

[1] Hereafter, the superscript of a vector or a matrix denotes the frame in which its components are expressed.

- First realign Frame 1 with Frame 0 by means of rotation $\boldsymbol{R}_0^1$.
- Then make the rotation expressed by $\bar{\boldsymbol{R}}_2^1$ with respect to the current frame.
- Finally compensate for the rotation made for the realignment by means of the inverse rotation $\boldsymbol{R}_1^0$.

Since the above rotations are described with respect to the current frame, the application of the composition rule (2.15) yields

$$\bar{\boldsymbol{R}}_2^0 = \boldsymbol{R}_1^0 \boldsymbol{R}_0^1 \bar{\boldsymbol{R}}_2^1 \boldsymbol{R}_1^0.$$

In view of (2.16), it is

$$\bar{\boldsymbol{R}}_2^0 = \bar{\boldsymbol{R}}_2^1 \boldsymbol{R}_1^0 \tag{2.17}$$

where the resulting $\bar{\boldsymbol{R}}_2^0$ is different from the matrix $\boldsymbol{R}_2^0$ in (2.15). Hence, it can be stated that composition of successive rotations with respect to a fixed frame is obtained by premultiplication of the single rotation matrices in the order of the given sequence of rotations.

By recalling the meaning of a rotation matrix in terms of the orientation of a current frame with respect to a fixed frame, it can be recognized that its columns are the direction cosines of the axes of the current frame with respect to the fixed frame, while its rows (columns of its transpose and inverse) are the direction cosines of the axes of the fixed frame with respect to the current frame.

An important issue of composition of rotations is that the matrix product is not commutative. In view of this, it can be concluded that two rotations in general do not commute and its composition depends on the order of the single rotations.

Example 2.3

Consider an object and a frame attached to it. Figure 2.6 shows the effects of two successive rotations of the object with respect to the current frame by changing the order of rotations. It is evident that the final object orientation is different in the two cases. Also in the case of rotations made with respect to the current frame, the final orientations differ (Fig. 2.7). It is interesting to note that the effects of the sequence of rotations with respect to the fixed frame are interchanged with the effects of the sequence of rotations with respect to the current frame. This can be explained by observing that the order of rotations in the fixed frame commutes with respect to the order of rotations in the current frame.

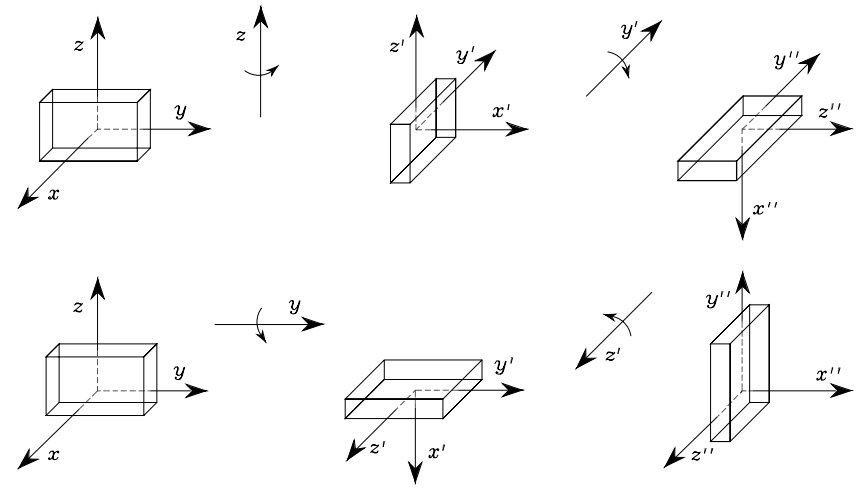

Fig. 2.6. Successive rotations of an object about axes of current frame

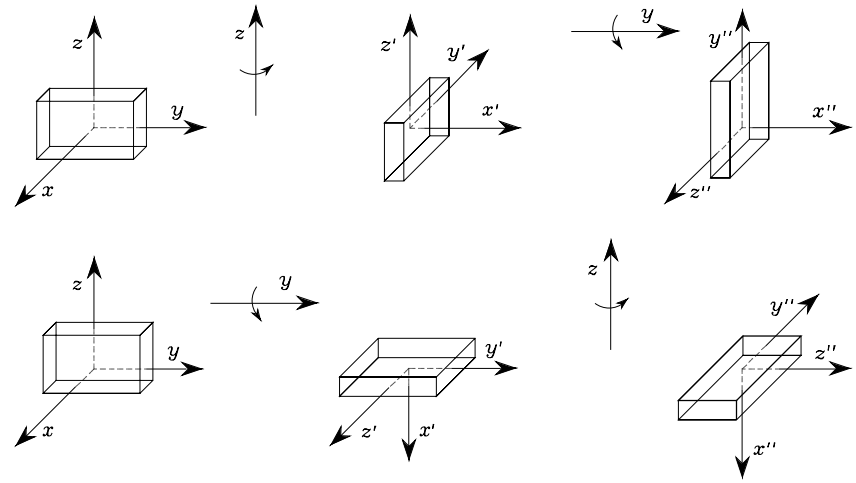

Fig. 2.7. Successive rotations of an object about axes of fixed frame

2.4 Euler Angles

Rotation matrices give a redundant description of frame orientation; in fact, they are characterized by nine elements which are not independent but related by six constraints due to the orthogonality conditions given in (2.4). This implies that *three parameters* are sufficient to describe orientation of a rigid body

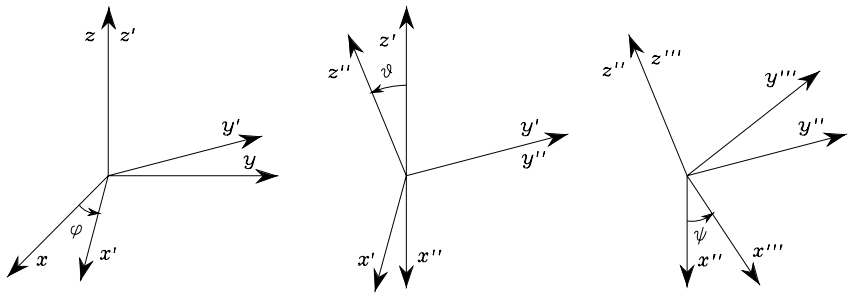

Fig. 2.8. Representation of Euler angles ZYZ

in space. A representation of orientation in terms of three independent parameters constitutes a *minimal representation*. In fact, a minimal representation of the special orthonormal group $SO(m)$ requires $m(m-1)/2$ parameters; thus, three parameters are needed to parameterize $SO(3)$, whereas only one parameter is needed for a planar rotation $SO(2)$.

A minimal representation of orientation can be obtained by using a set of three angles $\boldsymbol{\phi} = [\varphi \quad \vartheta \quad \psi]^T$. Consider the rotation matrix expressing the elementary rotation about one of the coordinate axes as a function of a single angle. Then, a generic rotation matrix can be obtained by composing a suitable sequence of three elementary rotations while guaranteeing that two successive rotations are not made about parallel axes. This implies that 12 distinct sets of angles are allowed out of all 27 possible combinations; each set represents a triplet of *Euler angles*. In the following, two sets of Euler angles are analyzed; namely, the ZYZ angles and the ZYX (or Roll–Pitch–Yaw) angles.

2.4.1 ZYZ Angles

The rotation described by *ZYZ angles* is obtained as composition of the following elementary rotations (Fig. 2.8):

- Rotate the reference frame by the angle φ about axis z; this rotation is described by the matrix $\boldsymbol{R}_z(\varphi)$ which is formally defined in (2.6).
- Rotate the current frame by the angle ϑ about axis y'; this rotation is described by the matrix $\boldsymbol{R}_{y'}(\vartheta)$ which is formally defined in (2.7).
- Rotate the current frame by the angle ψ about axis z''; this rotation is described by the matrix $\boldsymbol{R}_{z''}(\psi)$ which is again formally defined in (2.6).

The resulting frame orientation is obtained by composition of rotations with respect to *current frames*, and then it can be computed via postmultiplication of the matrices of elementary rotation, i.e.,[2]

$$\begin{aligned}\boldsymbol{R}(\boldsymbol{\phi}) &= \boldsymbol{R}_z(\varphi)\boldsymbol{R}_{y'}(\vartheta)\boldsymbol{R}_{z''}(\psi) \\ &= \begin{bmatrix} c_\varphi c_\vartheta c_\psi - s_\varphi s_\psi & -c_\varphi c_\vartheta s_\psi - s_\varphi c_\psi & c_\varphi s_\vartheta \\ s_\varphi c_\vartheta c_\psi + c_\varphi s_\psi & -s_\varphi c_\vartheta s_\psi + c_\varphi c_\psi & s_\varphi s_\vartheta \\ -s_\vartheta c_\psi & s_\vartheta s_\psi & c_\vartheta \end{bmatrix}.\end{aligned} \tag{2.18}$$

It is useful to solve the *inverse problem*, that is to determine the set of Euler angles corresponding to a given rotation matrix

$$\boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}.$$

Compare this expression with that of $\boldsymbol{R}(\boldsymbol{\phi})$ in (2.18). By considering the elements $[1,3]$ and $[2,3]$, under the assumption that $r_{13} \neq 0$ and $r_{23} \neq 0$, it follows that

$$\varphi = \text{Atan2}(r_{23}, r_{13})$$

where $\text{Atan2}(y,x)$ is the arctangent function of two arguments[3]. Then, squaring and summing the elements $[1,3]$ and $[2,3]$ and using the element $[3,3]$ yields

$$\vartheta = \text{Atan2}\left(\sqrt{r_{13}^2 + r_{23}^2}, r_{33}\right).$$

The choice of the positive sign for the term $\sqrt{r_{13}^2 + r_{23}^2}$ limits the range of feasible values of ϑ to $(0, \pi)$. On this assumption, considering the elements $[3,1]$ and $[3,2]$ gives

$$\psi = \text{Atan2}(r_{32}, -r_{31}).$$

In sum, the requested solution is

$$\begin{aligned}\varphi &= \text{Atan2}(r_{23}, r_{13}) \\ \vartheta &= \text{Atan2}\left(\sqrt{r_{13}^2 + r_{23}^2}, r_{33}\right) \\ \psi &= \text{Atan2}(r_{32}, -r_{31}).\end{aligned} \tag{2.19}$$

It is possible to derive another solution which produces the same effects as solution (2.19). Choosing ϑ in the range $(-\pi, 0)$ leads to

$$\varphi = \text{Atan2}(-r_{23}, -r_{13})$$

[2] The notations c_ϕ and s_ϕ are the abbreviations for $\cos\phi$ and $\sin\phi$, respectively; short-hand notations of this kind will be adopted often throughout the text.

[3] The function $\text{Atan2}(y,x)$ computes the arctangent of the ratio y/x but utilizes the sign of each argument to determine which quadrant the resulting angle belongs to; this allows the correct determination of an angle in a range of 2π.

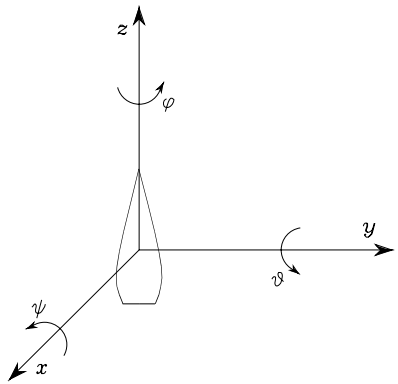

Fig. 2.9. Representation of Roll–Pitch–Yaw angles

$$\begin{aligned}\vartheta &= \mathrm{Atan2}\left(-\sqrt{r_{13}^2 + r_{23}^2}, r_{33}\right) \\ \psi &= \mathrm{Atan2}(-r_{32}, r_{31}).\end{aligned} \tag{2.20}$$

Solutions (2.19), (2.20) degenerate when $s_\vartheta = 0$; in this case, it is possible to determine only the sum or difference of φ and ψ. In fact, if $\vartheta = 0, \pi$, the successive rotations of φ and ψ are made about axes of current frames which are parallel, thus giving equivalent contributions to the rotation; see Problem 2.2.[4]

2.4.2 RPY Angles

Another set of Euler angles originates from a representation of orientation in the (aero)nautical field. These are the ZYX angles, also called *Roll–Pitch–Yaw angles*, to denote the typical changes of attitude of an (air)craft. In this case, the angles $\boldsymbol{\phi} = [\,\varphi \quad \vartheta \quad \psi\,]^T$ represent rotations defined with respect to a fixed frame attached to the centre of mass of the craft (Fig. 2.9).

The rotation resulting from Roll–Pitch–Yaw angles can be obtained as follows:

- Rotate the reference frame by the angle ψ about axis x (yaw); this rotation is described by the matrix $\boldsymbol{R}_x(\psi)$ which is formally defined in (2.8).
- Rotate the reference frame by the angle ϑ about axis y (pitch); this rotation is described by the matrix $\boldsymbol{R}_y(\vartheta)$ which is formally defined in (2.7).
- Rotate the reference frame by the angle φ about axis z (roll); this rotation is described by the matrix $\boldsymbol{R}_z(\varphi)$ which is formally defined in (2.6).

[4] In the following chapter, it will be seen that these configurations characterize the so-called representation *singularities* of the Euler angles.

The resulting frame orientation is obtained by composition of rotations with respect to the *fixed frame*, and then it can be computed via premultiplication of the matrices of elementary rotation, i.e.,[5]

$$\begin{aligned}\boldsymbol{R}(\boldsymbol{\phi}) &= \boldsymbol{R}_z(\varphi)\boldsymbol{R}_y(\vartheta)\boldsymbol{R}_x(\psi) \\ &= \begin{bmatrix} c_\varphi c_\vartheta & c_\varphi s_\vartheta s_\psi - s_\varphi c_\psi & c_\varphi s_\vartheta c_\psi + s_\varphi s_\psi \\ s_\varphi c_\vartheta & s_\varphi s_\vartheta s_\psi + c_\varphi c_\psi & s_\varphi s_\vartheta c_\psi - c_\varphi s_\psi \\ -s_\vartheta & c_\vartheta s_\psi & c_\vartheta c_\psi \end{bmatrix}.\end{aligned} \tag{2.21}$$

As for the Euler angles ZYZ, the *inverse solution* to a given rotation matrix

$$\boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix},$$

can be obtained by comparing it with the expression of $\boldsymbol{R}(\boldsymbol{\phi})$ in (2.21). The solution for ϑ in the range $(-\pi/2, \pi/2)$ is

$$\begin{aligned}\varphi &= \mathrm{Atan2}(r_{21}, r_{11}) \\ \vartheta &= \mathrm{Atan2}\left(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2}\right) \\ \psi &= \mathrm{Atan2}(r_{32}, r_{33}).\end{aligned} \tag{2.22}$$

The other equivalent solution for ϑ in the range $(\pi/2, 3\pi/2)$ is

$$\begin{aligned}\varphi &= \mathrm{Atan2}(-r_{21}, -r_{11}) \\ \vartheta &= \mathrm{Atan2}\left(-r_{31}, -\sqrt{r_{32}^2 + r_{33}^2}\right) \\ \psi &= \mathrm{Atan2}(-r_{32}, -r_{33}).\end{aligned} \tag{2.23}$$

Solutions (2.22), (2.23) degenerate when $c_\vartheta = 0$; in this case, it is possible to determine only the sum or difference of φ and ψ.

2.5 Angle and Axis

A nonminimal representation of orientation can be obtained by resorting to *four parameters* expressing a rotation of a given angle about an axis in space. This can be advantageous in the problem of trajectory planning for a manipulator's end-effector orientation.

Let $\boldsymbol{r} = [\,r_x \quad r_y \quad r_z\,]^T$ be the unit vector of a rotation axis with respect to the reference frame O–xyz. In order to derive the rotation matrix $\boldsymbol{R}(\vartheta, \boldsymbol{r})$ expressing the rotation of an *angle* ϑ about *axis* $\boldsymbol{r}$, it is convenient to compose

[5] The ordered sequence of rotations XYZ about axes of the fixed frame is equivalent to the sequence ZYX about axes of the current frame.

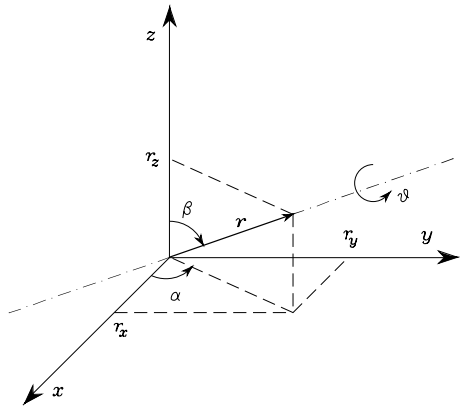

Fig. 2.10. Rotation of an angle about an axis

elementary rotations about the coordinate axes of the reference frame. The angle is taken to be positive if the rotation is made counter-clockwise about axis $\boldsymbol{r}$.

As shown in Fig. 2.10, a possible solution is to rotate first $\boldsymbol{r}$ by the angles necessary to align it with axis z, then to rotate by ϑ about z and finally to rotate by the angles necessary to align the unit vector with the initial direction. In detail, the sequence of rotations, to be made always with respect to axes of fixed frame, is the following:

- Align $\boldsymbol{r}$ with z, which is obtained as the sequence of a rotation by $-\alpha$ about z and a rotation by $-\beta$ about y.
- Rotate by ϑ about z.
- Realign with the initial direction of $\boldsymbol{r}$, which is obtained as the sequence of a rotation by β about y and a rotation by α about z.

In sum, the resulting rotation matrix is

$$\boldsymbol{R}(\vartheta, \boldsymbol{r}) = \boldsymbol{R}_z(\alpha)\boldsymbol{R}_y(\beta)\boldsymbol{R}_z(\vartheta)\boldsymbol{R}_y(-\beta)\boldsymbol{R}_z(-\alpha). \tag{2.24}$$

From the components of the unit vector $\boldsymbol{r}$ it is possible to extract the transcendental functions needed to compute the rotation matrix in (2.24), so as to eliminate the dependence from α and β; in fact, it is

$$\sin\alpha = \frac{r_y}{\sqrt{r_x^2 + r_y^2}} \qquad \cos\alpha = \frac{r_x}{\sqrt{r_x^2 + r_y^2}}$$

$$\sin\beta = \sqrt{r_x^2 + r_y^2} \qquad \cos\beta = r_z.$$

Then, it can be found that the rotation matrix corresponding to a given angle and axis is — see Problem 2.4 —

$$\boldsymbol{R}(\vartheta, \boldsymbol{r}) = \begin{bmatrix} r_x^2(1-c_\vartheta) + c_\vartheta & r_x r_y(1-c_\vartheta) - r_z s_\vartheta & r_x r_z(1-c_\vartheta) + r_y s_\vartheta \\ r_x r_y(1-c_\vartheta) + r_z s_\vartheta & r_y^2(1-c_\vartheta) + c_\vartheta & r_y r_z(1-c_\vartheta) - r_x s_\vartheta \\ r_x r_z(1-c_\vartheta) - r_y s_\vartheta & r_y r_z(1-c_\vartheta) + r_x s_\vartheta & r_z^2(1-c_\vartheta) + c_\vartheta \end{bmatrix}. \tag{2.25}$$

For this matrix, the following property holds:

$$\boldsymbol{R}(-\vartheta, -\boldsymbol{r}) = \boldsymbol{R}(\vartheta, \boldsymbol{r}), \tag{2.26}$$

i.e., a rotation by $-\vartheta$ about $-\boldsymbol{r}$ cannot be distinguished from a rotation by ϑ about $\boldsymbol{r}$; hence, such representation is not unique.

If it is desired to solve the *inverse problem* to compute the axis and angle corresponding to a given rotation matrix

$$\boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix},$$

the following result is useful:

$$\vartheta = \cos^{-1}\left(\frac{r_{11} + r_{22} + r_{33} - 1}{2}\right) \tag{2.27}$$

$$\boldsymbol{r} = \frac{1}{2\sin\vartheta}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}, \tag{2.28}$$

for $\sin\vartheta \neq 0$. Notice that the expressions (2.27), (2.28) describe the rotation in terms of four parameters; namely, the angle and the three components of the axis unit vector. However, it can be observed that the three components of $\boldsymbol{r}$ are not independent but are constrained by the condition

$$r_x^2 + r_y^2 + r_z^2 = 1. \tag{2.29}$$

If $\sin\vartheta = 0$, the expressions (2.27), (2.28) become meaningless. To solve the inverse problem, it is necessary to directly refer to the particular expressions attained by the rotation matrix $\boldsymbol{R}$ and find the solving formulae in the two cases $\vartheta = 0$ and $\vartheta = \pi$. Notice that, when $\vartheta = 0$ (null rotation), the unit vector $\boldsymbol{r}$ is arbitrary (singularity). See also Problem 2.5.

2.6 Unit Quaternion

The drawbacks of the angle/axis representation can be overcome by a different four-parameter representation; namely, the unit *quaternion*, viz. Euler parameters, defined as $\mathcal{Q} = \{\eta, \boldsymbol{\epsilon}\}$ where:

$$\eta = \cos\frac{\vartheta}{2} \tag{2.30}$$

$$\boldsymbol{\epsilon} = \sin\frac{\vartheta}{2}\,\boldsymbol{r}; \tag{2.31}$$

η is called the scalar part of the quaternion while $\boldsymbol{\epsilon} = [\,\epsilon_x \quad \epsilon_y \quad \epsilon_z\,]^T$ is called the vector part of the quaternion. They are constrained by the condition

$$\eta^2 + \epsilon_x^2 + \epsilon_y^2 + \epsilon_z^2 = 1, \tag{2.32}$$

hence, the name *unit* quaternion. It is worth remarking that, unlike the angle/axis representation, a rotation by $-\vartheta$ about $-\boldsymbol{r}$ gives the same quaternion as that associated with a rotation by ϑ about $\boldsymbol{r}$; this solves the above nonuniqueness problem. In view of (2.25), (2.30), (2.31), (2.32), the rotation matrix corresponding to a given quaternion takes on the form — see Problem 2.6 —

$$\boldsymbol{R}(\eta,\boldsymbol{\epsilon}) = \begin{bmatrix} 2(\eta^2+\epsilon_x^2)-1 & 2(\epsilon_x\epsilon_y-\eta\epsilon_z) & 2(\epsilon_x\epsilon_z+\eta\epsilon_y) \\ 2(\epsilon_x\epsilon_y+\eta\epsilon_z) & 2(\eta^2+\epsilon_y^2)-1 & 2(\epsilon_y\epsilon_z-\eta\epsilon_x) \\ 2(\epsilon_x\epsilon_z-\eta\epsilon_y) & 2(\epsilon_y\epsilon_z+\eta\epsilon_x) & 2(\eta^2+\epsilon_z^2)-1 \end{bmatrix}. \tag{2.33}$$

If it is desired to solve the *inverse problem* to compute the quaternion corresponding to a given rotation matrix

$$\boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix},$$

the following result is useful:

$$\eta = \frac{1}{2}\sqrt{r_{11}+r_{22}+r_{33}+1} \tag{2.34}$$

$$\boldsymbol{\epsilon} = \frac{1}{2}\begin{bmatrix} \mathrm{sgn}\,(r_{32}-r_{23})\sqrt{r_{11}-r_{22}-r_{33}+1} \\ \mathrm{sgn}\,(r_{13}-r_{31})\sqrt{r_{22}-r_{33}-r_{11}+1} \\ \mathrm{sgn}\,(r_{21}-r_{12})\sqrt{r_{33}-r_{11}-r_{22}+1} \end{bmatrix}, \tag{2.35}$$

where conventionally $\mathrm{sgn}\,(x) = 1$ for $x \geq 0$ and $\mathrm{sgn}\,(x) = -1$ for $x < 0$. Notice that in (2.34) it has been implicitly assumed $\eta \geq 0$; this corresponds to an angle $\vartheta \in [-\pi, \pi]$, and thus any rotation can be described. Also, compared to the inverse solution in (2.27), (2.28) for the angle and axis representation, no singularity occurs for (2.34), (2.35). See also Problem 2.8.

The quaternion extracted from $\boldsymbol{R}^{-1} = \boldsymbol{R}^T$ is denoted as $\mathcal{Q}^{-1}$, and can be computed as

$$\mathcal{Q}^{-1} = \{\eta, -\boldsymbol{\epsilon}\}. \tag{2.36}$$

Let $\mathcal{Q}_1 = \{\eta_1, \boldsymbol{\epsilon}_1\}$ and $\mathcal{Q}_2 = \{\eta_2, \boldsymbol{\epsilon}_2\}$ denote the quaternions corresponding to the rotation matrices $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$, respectively. The quaternion corresponding to the product $\boldsymbol{R}_1\boldsymbol{R}_2$ is given by

$$\mathcal{Q}_1 * \mathcal{Q}_2 = \{\eta_1\eta_2 - \boldsymbol{\epsilon}_1^T\boldsymbol{\epsilon}_2, \eta_1\boldsymbol{\epsilon}_2 + \eta_2\boldsymbol{\epsilon}_1 + \boldsymbol{\epsilon}_1 \times \boldsymbol{\epsilon}_2\} \tag{2.37}$$

where the quaternion product operator "$*$" has been formally introduced. It is easy to see that if $\mathcal{Q}_2 = \mathcal{Q}_1^{-1}$ then the quaternion $\{1, \boldsymbol{0}\}$ is obtained from (2.37) which is the identity element for the product. See also Problem 2.9.

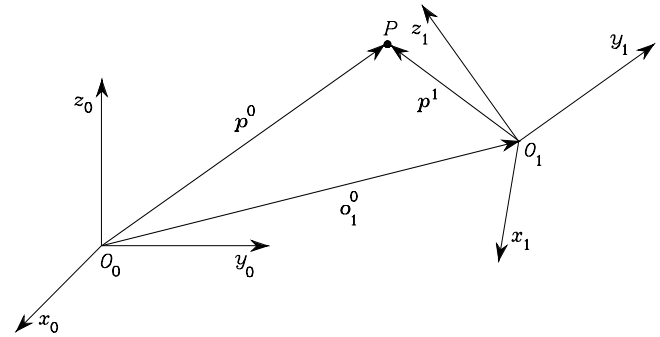

Fig. 2.11. Representation of a point P in different coordinate frames

2.7 Homogeneous Transformations

As illustrated at the beginning of the chapter, the position of a rigid body in space is expressed in terms of the position of a suitable point on the body with respect to a reference frame (translation), while its orientation is expressed in terms of the components of the unit vectors of a frame attached to the body — with origin in the above point — with respect to the same reference frame (rotation).

As shown in Fig. 2.11, consider an arbitrary point P in space. Let $\boldsymbol{p}^0$ be the vector of coordinates of P with respect to the reference frame O_0–$x_0y_0z_0$. Consider then another frame in space O_1–$x_1y_1z_1$. Let $\boldsymbol{o}_1^0$ be the vector describing the origin of Frame 1 with respect to Frame 0, and $\boldsymbol{R}_1^0$ be the rotation matrix of Frame 1 with respect to Frame 0. Let also $\boldsymbol{p}^1$ be the vector of coordinates of P with respect to Frame 1. On the basis of simple geometry, the position of point P with respect to the reference frame can be expressed as

$$\boldsymbol{p}^0 = \boldsymbol{o}_1^0 + \boldsymbol{R}_1^0\boldsymbol{p}^1. \tag{2.38}$$

Hence, (2.38) represents the *coordinate transformation* (*translation* + *rotation*) of a bound vector between two frames.

The inverse transformation can be obtained by premultiplying both sides of (2.38) by $\boldsymbol{R}_1^{0T}$; in view of(2.4), it follows that

$$\boldsymbol{p}^1 = -\boldsymbol{R}_1^{0T}\boldsymbol{o}_1^0 + \boldsymbol{R}_1^{0T}\boldsymbol{p}^0 \tag{2.39}$$

which, via (2.16), can be written as

$$\boldsymbol{p}^1 = -\boldsymbol{R}_0^1\boldsymbol{o}_1^0 + \boldsymbol{R}_0^1\boldsymbol{p}^0. \tag{2.40}$$

In order to achieve a compact representation of the relationship between the coordinates of the same point in two different frames, the *homogeneous representation* of a generic vector $\boldsymbol{p}$ can be introduced as the vector $\tilde{\boldsymbol{p}}$ formed by adding a fourth unit component, i.e.,

$$\widetilde{\boldsymbol{p}} = \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix}. \tag{2.41}$$

By adopting this representation for the vectors $\boldsymbol{p}^0$ and $\boldsymbol{p}^1$ in (2.38), the coordinate transformation can be written in terms of the (4×4) matrix

$$\boldsymbol{A}_1^0 = \begin{bmatrix} \boldsymbol{R}_1^0 & \boldsymbol{o}_1^0 \\ \boldsymbol{0}^T & 1 \end{bmatrix} \tag{2.42}$$

which, according to (2.41), is termed *homogeneous transformation matrix*. Since $\boldsymbol{o}_1^0 \in \mathbb{R}^3$ e $\boldsymbol{R}_1^0 \in SO(3)$, this matrix belongs to the *special Euclidean group* $SE(3) = \mathbb{R}^3 \times SO(3)$.

As can be easily seen from (2.42), the transformation of a vector from Frame 1 to Frame 0 is expressed by a single matrix containing the rotation matrix of Frame 1 with respect to Frame 0 and the translation vector from the origin of Frame 0 to the origin of Frame 1.[6] Therefore, the coordinate transformation (2.38) can be compactly rewritten as

$$\widetilde{\boldsymbol{p}}^0 = \boldsymbol{A}_1^0 \widetilde{\boldsymbol{p}}^1. \tag{2.43}$$

The coordinate transformation between Frame 0 and Frame 1 is described by the homogeneous transformation matrix $\boldsymbol{A}_0^1$ which satisfies the equation

$$\widetilde{\boldsymbol{p}}^1 = \boldsymbol{A}_0^1 \widetilde{\boldsymbol{p}}^0 = \left(\boldsymbol{A}_1^0\right)^{-1} \widetilde{\boldsymbol{p}}^0. \tag{2.44}$$

This matrix is expressed in a block-partitioned form as

$$\boldsymbol{A}_0^1 = \begin{bmatrix} \boldsymbol{R}_1^{0T} & -\boldsymbol{R}_1^{0T}\boldsymbol{o}_1^0 \\ \boldsymbol{0}^T & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_0^1 & -\boldsymbol{R}_0^1\boldsymbol{o}_1^0 \\ \boldsymbol{0}^T & 1 \end{bmatrix}, \tag{2.45}$$

which gives the homogeneous representation form of the result already established by (2.39), (2.40) — see Problem 2.10.

Notice that for the homogeneous transformation matrix the orthogonality property does not hold; hence, in general,

$$\boldsymbol{A}^{-1} \neq \boldsymbol{A}^T. \tag{2.46}$$

In sum, a homogeneous transformation matrix expresses the coordinate transformation between two frames in a compact form. If the frames have the same origin, it reduces to the rotation matrix previously defined. Instead, if the frames have distinct origins, it allows the notation with superscripts and subscripts to be kept which directly characterize the current frame and the fixed frame.

[6] It can be shown that in (2.42) non-null values of the first three elements of the fourth row of $\boldsymbol{A}$ produce a perspective effect, while values other than unity for the fourth element give a scaling effect.

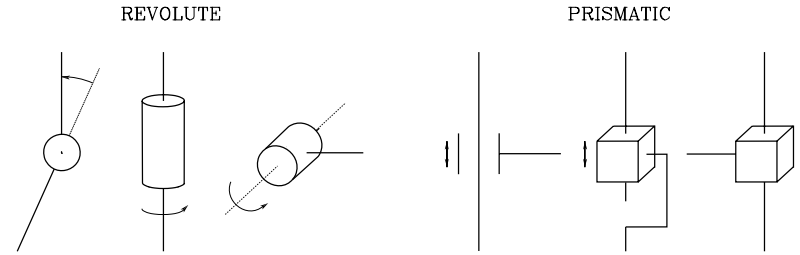

Fig. 2.12. Conventional representations of joints

Analogously to what presented for the rotation matrices, it is easy to verify that a sequence of coordinate transformations can be composed by the product

$$\widetilde{\boldsymbol{p}}^0 = \boldsymbol{A}_1^0 \boldsymbol{A}_2^1 \ldots \boldsymbol{A}_n^{n-1} \widetilde{\boldsymbol{p}}^n \tag{2.47}$$

where $\boldsymbol{A}_i^{i-1}$ denotes the homogeneous transformation relating the description of a point in Frame i to the description of the same point in Frame $i-1$.

2.8 Direct Kinematics

A manipulator consists of a series of rigid bodies (*links*) connected by means of kinematic pairs or *joints*. Joints can be essentially of two types: *revolute* and *prismatic*; conventional representations of the two types of joints are sketched in Fig. 2.12. The whole structure forms a *kinematic chain*. One end of the chain is constrained to a base. An *end-effector* (gripper, tool) is connected to the other end allowing manipulation of objects in space.

From a topological viewpoint, the kinematic chain is termed *open* when there is only one sequence of links connecting the two ends of the chain. Alternatively, a manipulator contains a *closed* kinematic chain when a sequence of links forms a loop.

The mechanical structure of a manipulator is characterized by a number of degrees of freedom (DOFs) which uniquely determine its *posture*.[7] Each DOF is typically associated with a joint articulation and constitutes a *joint variable*. The aim of *direct kinematics* is to compute the pose of the end-effector as a function of the joint variables.

[7] The term *posture* of a kinematic chain denotes the pose of all the rigid bodies composing the chain. Whenever the kinematic chain reduces to a single rigid body, then the posture coincides with the pose of the body.

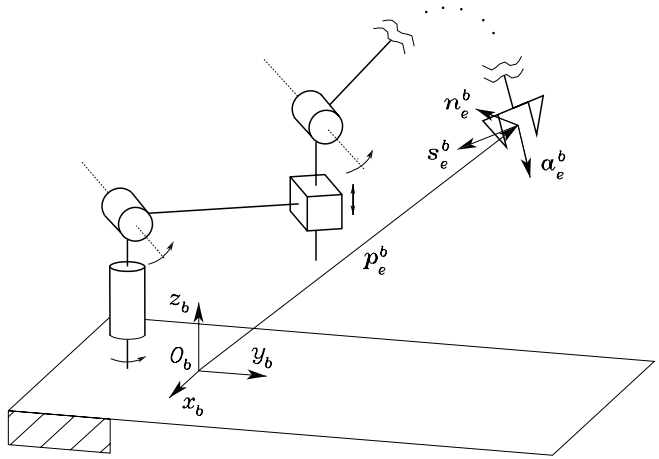

Fig. 2.13. Description of the position and orientation of the end-effector frame

It was previously illustrated that the pose of a body with respect to a reference frame is described by the position vector of the origin and the unit vectors of a frame attached to the body. Hence, with respect to a reference frame O_b–$x_by_bz_b$, the direct kinematics function is expressed by the homogeneous transformation matrix

$$\boldsymbol{T}_e^b(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{n}_e^b(\boldsymbol{q}) & \boldsymbol{s}_e^b(\boldsymbol{q}) & \boldsymbol{a}_e^b(\boldsymbol{q}) & \boldsymbol{p}_e^b(\boldsymbol{q}) \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.48}$$

where $\boldsymbol{q}$ is the $(n \times 1)$ vector of joint variables, $\boldsymbol{n}_e$, $\boldsymbol{s}_e$, $\boldsymbol{a}_e$ are the unit vectors of a frame attached to the end-effector, and $\boldsymbol{p}_e$ is the position vector of the origin of such a frame with respect to the origin of the base frame O_b–$x_by_bz_b$ (Fig. 2.13). Note that $\boldsymbol{n}_e$, $\boldsymbol{s}_e$, $\boldsymbol{a}_e$ and $\boldsymbol{p}_e$ are a function of $\boldsymbol{q}$.

The frame O_b–$x_by_bz_b$ is termed *base frame*. The frame attached to the end-effector is termed *end-effector frame* and is conveniently chosen according to the particular task geometry. If the end-effector is a gripper, the origin of the end-effector frame is located at the centre of the gripper, the unit vector $\boldsymbol{a}_e$ is chosen in the *approach* direction to the object, the unit vector $\boldsymbol{s}_e$ is chosen normal to $\boldsymbol{a}_e$ in the *sliding* plane of the jaws, and the unit vector $\boldsymbol{n}_e$ is chosen *normal* to the other two so that the frame $(\boldsymbol{n}_e, \boldsymbol{s}_e, \boldsymbol{a}_e)$ is right-handed.

A first way to compute direct kinematics is offered by a geometric analysis of the structure of the given manipulator.

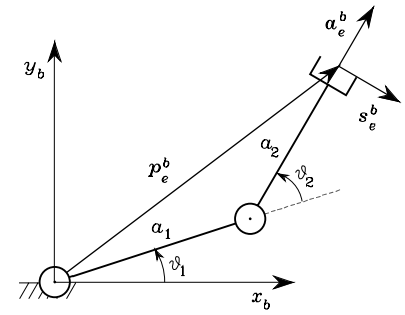

Fig. 2.14. Two-link planar arm

Example 2.4

Consider the two-link planar arm in Fig. 2.14. On the basis of simple trigonometry, the choice of the joint variables, the base frame, and the end-effector frame leads to[8]

$$\boldsymbol{T}_e^b(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{n}_e^b & \boldsymbol{s}_e^b & \boldsymbol{a}_e^b & \boldsymbol{p}_e^b \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & s_{12} & c_{12} & a_1c_1 + a_2c_{12} \\ 0 & -c_{12} & s_{12} & a_1s_1 + a_2s_{12} \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.49}$$

It is not difficult to infer that the effectiveness of a geometric approach to the direct kinematics problem is based first on a convenient choice of the relevant quantities and then on the ability and geometric intuition of the problem solver. Whenever the manipulator structure is complex and the number of joints increases, it is preferable to adopt a less direct solution, which, though, is based on a systematic, general procedure. The problem becomes even more complex when the manipulator contains one or more closed kinematic chains. In such a case, as it will be discussed later, there is no guarantee to obtain an analytical expression for the direct kinematics function in (2.48).

2.8.1 Open Chain

Consider an *open-chain* manipulator constituted by $n+1$ links connected by n joints, where Link 0 is conventionally fixed to the ground. It is assumed that each joint provides the mechanical structure with a single DOF, corresponding to the joint variable.

The construction of an operating procedure for the computation of direct kinematics is naturally derived from the typical open kinematic chain of the manipulator structure. In fact, since each joint connects two consecutive

[8] The notations $s_{i\ldots j}$, $c_{i\ldots j}$ denote respectively $\sin(q_i + \ldots + q_j)$, $\cos(q_i + \ldots + q_j)$.

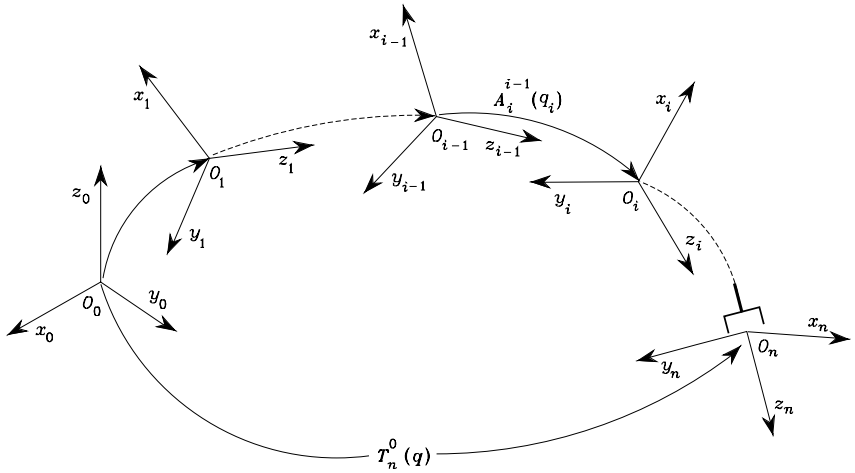

Fig. 2.15. Coordinate transformations in an open kinematic chain

links, it is reasonable to consider first the description of kinematic relationship between consecutive links and then to obtain the overall description of manipulator kinematics in a recursive fashion. To this purpose, it is worth defining a coordinate frame attached to each link, from Link 0 to Link n. Then, the coordinate transformation describing the position and orientation of Frame n with respect to Frame 0 (Fig. 2.15) is given by

$$\boldsymbol{T}_n^0(\boldsymbol{q}) = \boldsymbol{A}_1^0(q_1)\boldsymbol{A}_2^1(q_2)\dots\boldsymbol{A}_n^{n-1}(q_n). \tag{2.50}$$

As requested, the computation of the direct kinematics function is recursive and is obtained in a systematic manner by simple products of the homogeneous transformation matrices $\boldsymbol{A}_i^{i-1}(q_i)$ (for $i = 1, \dots, n$), each of which is a function of a single joint variable.

With reference to the direct kinematics equation in (2.49), the actual coordinate transformation describing the position and orientation of the end-effector frame with respect to the base frame can be obtained as

$$\boldsymbol{T}_e^b(\boldsymbol{q}) = \boldsymbol{T}_0^b\boldsymbol{T}_n^0(\boldsymbol{q})\boldsymbol{T}_e^n \tag{2.51}$$

where $\boldsymbol{T}_0^b$ and $\boldsymbol{T}_e^n$ are two (typically) constant homogeneous transformations describing the position and orientation of Frame 0 with respect to the base frame, and of the end-effector frame with respect to Frame n, respectively.

2.8.2 Denavit–Hartenberg Convention

In order to compute the direct kinematics equation for an open-chain manipulator according to the recursive expression in (2.50), a systematic, general

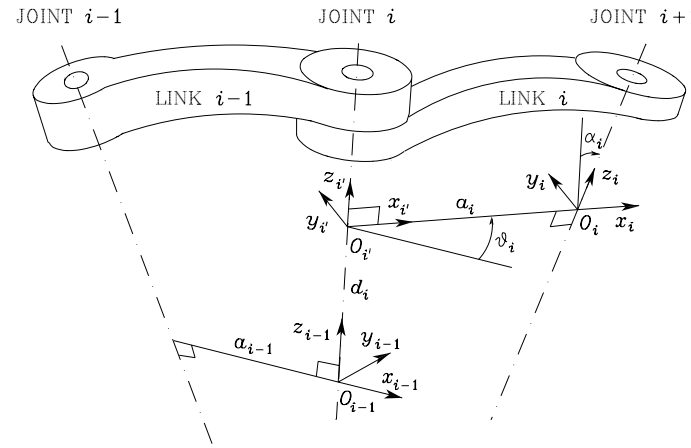

Fig. 2.16. Denavit–Hartenberg kinematic parameters

method is to be derived to define the relative position and orientation of two consecutive links; the problem is that to determine two frames attached to the two links and compute the coordinate transformations between them. In general, the frames can be arbitrarily chosen as long as they are attached to the link they are referred to. Nevertheless, it is convenient to set some rules also for the definition of the link frames.

With reference to Fig. 2.16, let Axis i denote the axis of the joint connecting Link $i-1$ to Link i; the so-called *Denavit–Hartenberg convention* (DH) is adopted to define link Frame i:

- Choose axis z_i along the axis of Joint $i+1$.
- Locate the origin O_i at the intersection of axis z_i with the common normal[9] to axes z_{i-1} and z_i. Also, locate $O_{i'}$ at the intersection of the common normal with axis z_{i-1}.
- Choose axis x_i along the common normal to axes z_{i-1} and z_i with direction from Joint i to Joint $i+1$.
- Choose axis y_i so as to complete a right-handed frame.

The Denavit–Hartenberg convention gives a nonunique definition of the link frame in the following cases:

- For Frame 0, only the direction of axis z_0 is specified; then O_0 and x_0 can be arbitrarily chosen.
- For Frame n, since there is no Joint $n+1$, z_n is not uniquely defined while x_n has to be normal to axis z_{n-1}. Typically, Joint n is revolute, and thus z_n is to be aligned with the direction of z_{n-1}.

[9] The common normal between two lines is the line containing the minimum distance segment between the two lines.

- When two consecutive axes are parallel, the common normal between them is not uniquely defined.
- When two consecutive axes intersect, the direction of x_i is arbitrary.
- When Joint i is prismatic, the direction of z_{i-1} is arbitrary.

In all such cases, the indeterminacy can be exploited to simplify the procedure; for instance, the axes of consecutive frames can be made parallel.

Once the link frames have been established, the position and orientation of Frame i with respect to Frame $i-1$ are completely specified by the following *parameters*:

a_i distance between O_i and $O_{i'}$,
d_i coordinate of $O_{i'}$ along z_{i-1},
α_i angle between axes z_{i-1} and z_i about axis x_i to be taken positive when rotation is made counter-clockwise,
ϑ_i angle between axes x_{i-1} and x_i about axis z_{i-1} to be taken positive when rotation is made counter-clockwise.

Two of the four parameters (a_i and α_i) are always constant and depend only on the geometry of connection between consecutive joints established by Link i. Of the remaining two parameters, only one is variable depending on the type of joint that connects Link $i-1$ to Link i. In particular:

- if Joint i is *revolute* the variable is ϑ_i,
- if Joint i is *prismatic* the variable is d_i.

At this point, it is possible to express the coordinate transformation between Frame i and Frame $i-1$ according to the following steps:

- Choose a frame aligned with Frame $i-1$.
- Translate the chosen frame by d_i along axis z_{i-1} and rotate it by ϑ_i about axis z_{i-1}; this sequence aligns the current frame with Frame i' and is described by the homogeneous transformation matrix

$$\boldsymbol{A}_{i'}^{i-1} = \begin{bmatrix} c_{\vartheta_i} & -s_{\vartheta_i} & 0 & 0 \\ s_{\vartheta_i} & c_{\vartheta_i} & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

- Translate the frame aligned with Frame i' by a_i along axis $x_{i'}$ and rotate it by α_i about axis $x_{i'}$; this sequence aligns the current frame with Frame i and is described by the homogeneous transformation matrix

$$\boldsymbol{A}_{i}^{i'} = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

- The resulting coordinate transformation is obtained by postmultiplication of the single transformations as

$$\boldsymbol{A}_{i}^{i-1}(q_i) = \boldsymbol{A}_{i'}^{i-1}\boldsymbol{A}_{i}^{i'} = \begin{bmatrix} c_{\vartheta_i} & -s_{\vartheta_i}c_{\alpha_i} & s_{\vartheta_i}s_{\alpha_i} & a_i c_{\vartheta_i} \\ s_{\vartheta_i} & c_{\vartheta_i}c_{\alpha_i} & -c_{\vartheta_i}s_{\alpha_i} & a_i s_{\vartheta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (2.52)$$

Notice that the transformation matrix from Frame i to Frame $i-1$ is a function only of the joint variable q_i, that is, ϑ_i for a revolute joint or d_i for a prismatic joint.

To summarize, the Denavit–Hartenberg convention allows the construction of the direct kinematics function by composition of the individual coordinate transformations expressed by (2.52) into one homogeneous transformation matrix as in (2.50). The procedure can be applied to any open kinematic chain and can be easily rewritten in an operating form as follows.

1. Find and number consecutively the joint axes; set the directions of axes $z_0, \ldots, z_{n-1}$.
2. Choose Frame 0 by locating the origin on axis z_0; axes x_0 and y_0 are chosen so as to obtain a right-handed frame. If feasible, it is worth choosing Frame 0 to coincide with the base frame.

Execute steps from **3** to **5** for $i = 1, \ldots, n-1$:

3. Locate the origin O_i at the intersection of z_i with the common normal to axes z_{i-1} and z_i. If axes z_{i-1} and z_i are parallel and Joint i is revolute, then locate O_i so that $d_i = 0$; if Joint i is prismatic, locate O_i at a reference position for the joint range, e.g., a mechanical limit.
4. Choose axis x_i along the common normal to axes z_{i-1} and z_i with direction from Joint i to Joint $i+1$.
5. Choose axis y_i so as to obtain a right-handed frame.

To complete:

6. Choose Frame n; if Joint n is revolute, then align z_n with z_{n-1}, otherwise, if Joint n is prismatic, then choose z_n arbitrarily. Axis x_n is set according to step **4**.
7. For $i = 1, \ldots, n$, form the table of parameters $a_i, d_i, \alpha_i, \vartheta_i$.
8. On the basis of the parameters in **7**, compute the homogeneous transformation matrices $\boldsymbol{A}_i^{i-1}(q_i)$ for $i = 1, \ldots, n$.
9. Compute the homogeneous transformation $\boldsymbol{T}_n^0(\boldsymbol{q}) = \boldsymbol{A}_1^0 \ldots \boldsymbol{A}_n^{n-1}$ that yields the position and orientation of Frame n with respect to Frame 0.
10. Given $\boldsymbol{T}_0^b$ and $\boldsymbol{T}_e^n$, compute the direct kinematics function as $\boldsymbol{T}_e^b(\boldsymbol{q}) = \boldsymbol{T}_0^b \boldsymbol{T}_n^0 \boldsymbol{T}_e^n$ that yields the position and orientation of the end-effector frame with respect to the base frame.

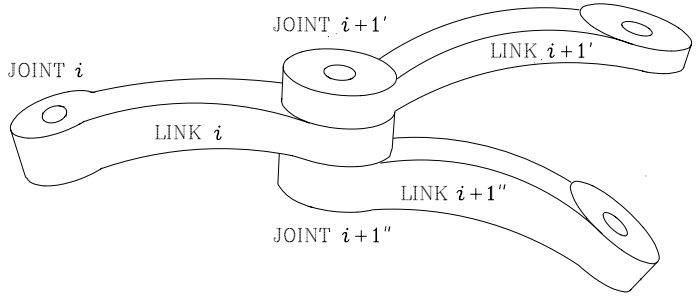

Fig. 2.17. Connection of a single link in the chain with two links

For what concerns the computational aspects of direct kinematics, it can be recognized that the heaviest load derives from the evaluation of transcendental functions. On the other hand, by suitably factorizing the transformation equations and introducing local variables, the number of flops (additions + multiplications) can be reduced. Finally, for computation of orientation it is convenient to evaluate the two unit vectors of the end-effector frame of simplest expression and derive the third one by vector product of the first two.

2.8.3 Closed Chain

The above direct kinematics method based on the DH convention exploits the inherently recursive feature of an open-chain manipulator. Nevertheless, the method can be extended to the case of manipulators containing closed kinematic chains according to the technique illustrated below.

Consider a *closed-chain* manipulator constituted by $n+1$ links. Because of the presence of a loop, the number of joints l must be greater than n; in particular, it can be understood that the number of closed loops is equal to $l-n$.

With reference to Fig. 2.17, Links 0 through i are connected successively through the first i joints as in an open kinematic chain. Then, Joint $i+1'$ connects Link i with Link $i+1'$ while Joint $i+1''$ connects Link i with Link $i+1''$; the axes of Joints $i+1'$ and $i+1''$ are assumed to be aligned. Although not represented in the figure, Links $i+1'$ and $i+1''$ are members of the closed kinematic chain. In particular, Link $i+1'$ is further connected to Link $i+2'$ via Joint $i+2'$ and so forth, until Link j via Joint j. Likewise, Link $i+1''$ is further connected to Link $i+2''$ via Joint $i+2''$ and so forth, until Link k via Joint k. Finally, Links j and k are connected together at Joint $j+1$ to form a closed chain. In general, $j \neq k$.

In order to attach frames to the various links and apply DH convention, one closed kinematic chain is taken into account. The closed chain can be virtually cut open at Joint $j+1$, i.e., the joint between Link j and Link k. An equivalent tree-structured open kinematic chain is obtained, and thus link

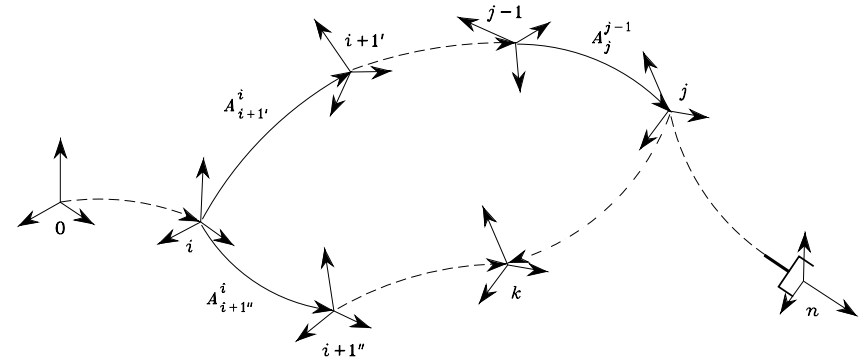

Fig. 2.18. Coordinate transformations in a closed kinematic chain

frames can be defined as in Fig. 2.18. Since Links 0 through i occur before the two branches of the tree, they are left out of the analysis. For the same reason, Links $j+1$ through n are left out as well. Notice that Frame i is to be chosen with axis z_i aligned with the axes of Joints $i+1'$ and $i+1''$.

It follows that the position and orientation of Frame j with respect to Frame i can be expressed by composing the homogeneous transformations as

$$\boldsymbol{A}_j^i(\boldsymbol{q}') = \boldsymbol{A}_{i+1'}^i(q_{i+1'}) \dots \boldsymbol{A}_j^{j-1}(q_j) \tag{2.53}$$

where $\boldsymbol{q}' = [\, q_{i+1'} \quad \dots \quad q_j \,]^T$. Likewise, the position and orientation of Frame k with respect to Frame i is given by

$$\boldsymbol{A}_k^i(\boldsymbol{q}'') = \boldsymbol{A}_{i+1''}^i(q_{i+1''}) \dots \boldsymbol{A}_k^{k-1}(q_k) \tag{2.54}$$

where $\boldsymbol{q}'' = [\, q_{i+1''} \quad \dots \quad q_k \,]^T$.

Since Links j and k are connected to each other through Joint $j+1$, it is worth analyzing the mutual position and orientation between Frames j and k, as illustrated in Fig. 2.19. Notice that, since Links j and k are connected to form a closed chain, axes z_j and z_k are aligned. Therefore, the following orientation constraint has to be imposed between Frames j and k:

$$\boldsymbol{z}_j^i(\boldsymbol{q}') = \boldsymbol{z}_k^i(\boldsymbol{q}''), \tag{2.55}$$

where the unit vectors of the two axes have been conveniently referred to Frame i.

Moreover, if Joint $j+1$ is prismatic, the angle ϑ_{jk} between axes x_j and x_k is fixed; hence, in addition to (2.55), the following constraint is obtained:

$$\boldsymbol{x}_j^{iT}(\boldsymbol{q}')\boldsymbol{x}_k^i(\boldsymbol{q}'') = \cos\vartheta_{jk}. \tag{2.56}$$

Obviously, there is no need to impose a similar constraint on axes y_j and y_k since that would be redundant.

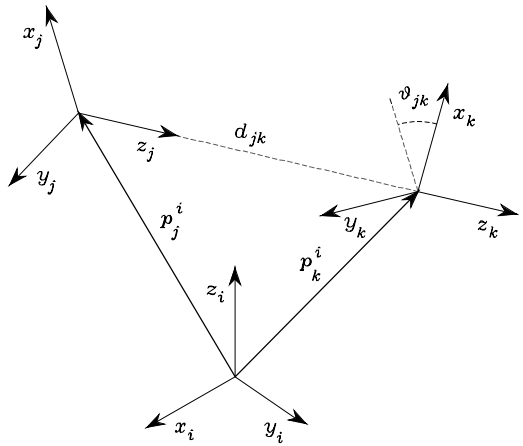

Fig. 2.19. Coordinate transformation at the cut joint

Regarding the position constraint between Frames j and k, let $\boldsymbol{p}_j^i$ and $\boldsymbol{p}_k^i$ respectively denote the positions of the origins of Frames j and k, when referred to Frame i. By projecting on Frame j the distance vector of the origin of Frame k from Frame j, the following constraint has to be imposed:

$$\boldsymbol{R}_i^j(\boldsymbol{q}')\left(\boldsymbol{p}_j^i(\boldsymbol{q}') - \boldsymbol{p}_k^i(\boldsymbol{q}'')\right) = [\,0 \quad 0 \quad d_{jk}\,]^T \tag{2.57}$$

where $\boldsymbol{R}_i^j = \boldsymbol{R}_j^{iT}$ denotes the orientation of Frame i with respect to Frame j. At this point, if Joint $j+1$ is revolute, then d_{jk} is a fixed offset along axis z_j; hence, the three equalities of (2.57) fully describe the position constraint. If, however, Joint $j+1$ is prismatic, then d_{jk} varies. Consequently, only the first two equalities of (2.57) describe the position constraint, i.e.,

$$\begin{bmatrix} \boldsymbol{x}_j^{iT}(\boldsymbol{q}') \\ \boldsymbol{y}_j^{iT}(\boldsymbol{q}') \end{bmatrix} \left(\boldsymbol{p}_j^i(\boldsymbol{q}') - \boldsymbol{p}_k^i(\boldsymbol{q}'')\right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{2.58}$$

where $\boldsymbol{R}_j^i = [\,\boldsymbol{x}_j^i \quad \boldsymbol{y}_j^i \quad \boldsymbol{z}_j^i\,]$.

In summary, if Joint $j+1$ is *revolute* the constraints are

$$\begin{cases} \boldsymbol{R}_i^j(\boldsymbol{q}')\left(\boldsymbol{p}_j^i(\boldsymbol{q}') - \boldsymbol{p}_k^i(\boldsymbol{q}'')\right) = [\,0 \quad 0 \quad d_{jk}\,]^T \\ \boldsymbol{z}_j^i(\boldsymbol{q}') = \boldsymbol{z}_k^i(\boldsymbol{q}''), \end{cases} \tag{2.59}$$

whereas if Joint $j+1$ is *prismatic* the constraints are

$$\begin{cases} \begin{bmatrix} \boldsymbol{x}_j^{iT}(\boldsymbol{q}') \\ \boldsymbol{y}_j^{iT}(\boldsymbol{q}') \end{bmatrix} \left(\boldsymbol{p}_j^i(\boldsymbol{q}') - \boldsymbol{p}_k^i(\boldsymbol{q}'')\right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ \boldsymbol{z}_j^i(\boldsymbol{q}') = \boldsymbol{z}_k^i(\boldsymbol{q}'') \\ \boldsymbol{x}_j^{iT}(\boldsymbol{q}')\boldsymbol{x}_k^i(\boldsymbol{q}'') = \cos\vartheta_{jk}. \end{cases} \tag{2.60}$$

In either case, there are six equalities that must be satisfied. Those should be solved for a reduced number of independent joint variables to be keenly chosen among the components of $\boldsymbol{q}'$ and $\boldsymbol{q}''$ which characterize the DOFs of the closed chain. These are the natural candidates to be the actuated joints, while the other joints in the chain (including the cut joint) are typically not actuated. Such independent variables, together with the remaining joint variables not involved in the above analysis, constitute the joint vector $\boldsymbol{q}$ that allows the direct kinematics equation to be computed as

$$\boldsymbol{T}_n^0(\boldsymbol{q}) = \boldsymbol{A}_i^0\boldsymbol{A}_j^i\boldsymbol{A}_n^j, \tag{2.61}$$

where the sequence of successive transformations after the closure of the chain has been conventionally resumed from Frame j.

In general, there is no guarantee to solve the constraints in closed form unless the manipulator has a simple kinematic structure. In other words, for a given manipulator with a specific geometry, e.g., a planar structure, some of the above equalities may become dependent. Hence, the number of independent equalities is less than six and it should likely be easier to solve them.

To conclude, it is worth sketching the operating form of the procedure to compute the direct kinematics function for a closed-chain manipulator using the Denavit–Hartenberg convention.

1. In the closed chain, select one joint that is not actuated. Assume that the joint is cut open so as to obtain an open chain in a tree structure.
2. Compute the homogeneous transformations according to DH convention.
3. Find the equality constraints for the two frames connected by the cut joint.
4. Solve the constraints for a reduced number of joint variables.
5. Express the homogeneous transformations in terms of the above joint variables and compute the direct kinematics function by composing the various transformations from the base frame to the end-effector frame.

2.9 Kinematics of Typical Manipulator Structures

This section contains several examples of computation of the direct kinematics function for typical manipulator structures that are often encountered in industrial robots.

With reference to the schematic representation of the kinematic chain, manipulators are usually illustrated in postures where the joint variables, defined according to the DH convention, are different from zero; such values might differ from the null references utilized for robot manipulator programming. Hence, it will be necessary to sum constant contributions (offsets) to the values of the joint variables measured by the robot sensory system, so as to match the references.

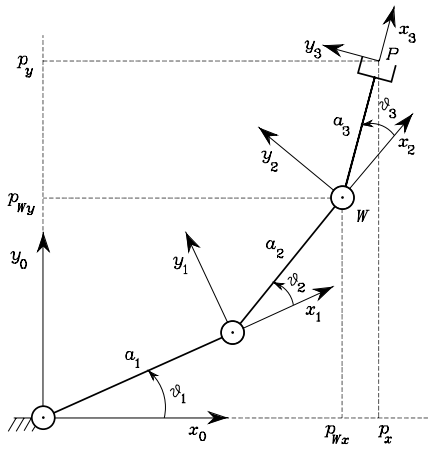

Fig. 2.20. Three-link planar arm

2.9.1 Three-link Planar Arm

Consider the three-link planar arm in Fig. 2.20, where the link frames have been illustrated. Since the revolute axes are all parallel, the simplest choice was made for all axes x_i along the direction of the relative links (the direction of x_0 is arbitrary) and all lying in the plane (x_0, y_0). In this way, all the parameters d_i are null and the angles between the axes x_i directly provide the joint variables. The DH parameters are specified in Table 2.1.

Table 2.1. DH parameters for the three-link planar arm

Link	a_i	α_i	d_i	ϑ_i
1	a_1	0	0	ϑ_1
2	a_2	0	0	ϑ_2
3	a_3	0	0	ϑ_3

Since all joints are revolute, the homogeneous transformation matrix defined in (2.52) has the same structure for each joint, i.e.,

$$\boldsymbol{A}_i^{i-1}(\vartheta_i) = \begin{bmatrix} c_i & -s_i & 0 & a_i c_i \\ s_i & c_i & 0 & a_i s_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad i = 1, 2, 3. \tag{2.62}$$

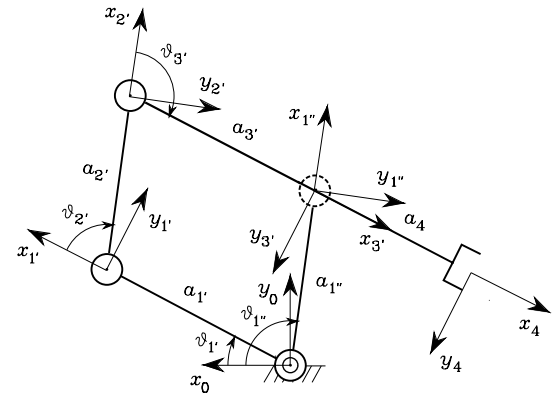

Fig. 2.21. Parallelogram arm

Computation of the direct kinematics function as in (2.50) yields

$$\boldsymbol{T}_3^0(\boldsymbol{q}) = \boldsymbol{A}_1^0 \boldsymbol{A}_2^1 \boldsymbol{A}_3^2 = \begin{bmatrix} c_{123} & -s_{123} & 0 & a_1 c_1 + a_2 c_{12} + a_3 c_{123} \\ s_{123} & c_{123} & 0 & a_1 s_1 + a_2 s_{12} + a_3 s_{123} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.63}$$

where $\boldsymbol{q} = [\,\vartheta_1 \quad \vartheta_2 \quad \vartheta_3\,]^T$. Notice that the unit vector $\boldsymbol{z}_3^0$ of Frame 3 is aligned with $\boldsymbol{z}_0 = [\,0 \quad 0 \quad 1\,]^T$, in view of the fact that all revolute joints are parallel to axis z_0. Obviously, $p_z = 0$ and all three joints concur to determine the end-effector position in the plane of the structure. It is worth pointing out that Frame 3 does not coincide with the end-effector frame (Fig. 2.13), since the resulting approach unit vector is aligned with $\boldsymbol{x}_3^0$ and not with $\boldsymbol{z}_3^0$. Thus, assuming that the two frames have the same origin, the constant transformation

$$\boldsymbol{T}_e^3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

is needed, having taken $\boldsymbol{n}$ aligned with $\boldsymbol{z}_0$.

2.9.2 Parallelogram Arm

Consider the parallelogram arm in Fig. 2.21. A closed chain occurs where the first two joints connect Link $1'$ and Link $1''$ to Link 0, respectively. Joint 4 was selected as the cut joint, and the link frames have been established accordingly. The DH parameters are specified in Table 2.2, where $a_{1'} = a_{3'}$ and $a_{2'} = a_{1''}$ in view of the parallelogram structure.

Notice that the parameters for Link 4 are all constant. Since the joints are revolute, the homogeneous transformation matrix defined in (2.52) has

Table 2.2. DH parameters for the parallelogram arm

Link	a_i	α_i	d_i	ϑ_i
$1'$	$a_{1'}$	0	0	$\vartheta_{1'}$
$2'$	$a_{2'}$	0	0	$\vartheta_{2'}$
$3'$	$a_{3'}$	0	0	$\vartheta_{3'}$
$1''$	$a_{1''}$	0	0	$\vartheta_{1''}$
4	a_4	0	0	0

the same structure for each joint, i.e., as in (2.62) for Joints $1'$, $2'$, $3'$ and $1''$. Therefore, the coordinate transformations for the two branches of the tree are respectively:

$$\boldsymbol{A}^0_{3'}(\boldsymbol{q}') = \boldsymbol{A}^0_{1'}\boldsymbol{A}^{1'}_{2'}\boldsymbol{A}^{2'}_{3'} = \begin{bmatrix} c_{1'2'3'} & -s_{1'2'3'} & 0 & a_{1'}c_{1'} + a_{2'}c_{1'2'} + a_{3'}c_{1'2'3'} \\ s_{1'2'3'} & c_{1'2'3'} & 0 & a_{1'}s_{1'} + a_{2'}s_{1'2'} + a_{3'}s_{1'2'3'} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $\boldsymbol{q}' = [\,\vartheta_{1'} \quad \vartheta_{2'} \quad \vartheta_{3'}\,]^T$, and

$$\boldsymbol{A}^0_{1''}(q'') = \begin{bmatrix} c_{1''} & -s_{1''} & 0 & a_{1''}c_{1''} \\ s_{1''} & c_{1''} & 0 & a_{1''}s_{1''} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $q'' = \vartheta_{1''}$. To complete, the constant homogeneous transformation for the last link is

$$\boldsymbol{A}^{3'}_4 = \begin{bmatrix} 1 & 0 & 0 & a_4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

With reference to (2.59), the position constraints are ($d_{3'1''} = 0$)

$$\boldsymbol{R}^{3'}_0(\boldsymbol{q}')\left(\boldsymbol{p}^0_{3'}(\boldsymbol{q}') - \boldsymbol{p}^0_{1''}(q'')\right) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

while the orientation constraints are satisfied independently of $\boldsymbol{q}'$ and q''. Since $a_{1'} = a_{3'}$ and $a_{2'} = a_{1''}$, two independent constraints can be extracted, i.e.,

$$\begin{aligned} a_{1'}(c_{1'} + c_{1'2'3'}) + a_{1''}(c_{1'2'} - c_{1''}) &= 0 \\ a_{1'}(s_{1'} + s_{1'2'3'}) + a_{1''}(s_{1'2'} - s_{1''}) &= 0. \end{aligned}$$

In order to satisfy them for any choice of $a_{1'}$ and $a_{1''}$, it must be

$$\begin{aligned} \vartheta_{2'} &= \vartheta_{1''} - \vartheta_{1'} \\ \vartheta_{3'} &= \pi - \vartheta_{2'} = \pi - \vartheta_{1''} + \vartheta_{1'} \end{aligned}$$

Therefore, the vector of joint variables is $\boldsymbol{q} = [\,\vartheta_{1'} \quad \vartheta_{1''}\,]^T$. These joints are natural candidates to be the actuated joints.[10] Substituting the expressions of $\vartheta_{2'}$ and $\vartheta_{3'}$ into the homogeneous transformation $\boldsymbol{A}^0_{3'}$ and computing the direct kinematics function as in (2.61) yields

$$\boldsymbol{T}^0_4(\boldsymbol{q}) = \boldsymbol{A}^0_{3'}(\boldsymbol{q})\boldsymbol{A}^{3'}_4 = \begin{bmatrix} -c_{1'} & s_{1'} & 0 & a_{1''}c_{1''} - a_4c_{1'} \\ -s_{1'} & -c_{1'} & 0 & a_{1''}s_{1''} - a_4s_{1'} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.64}$$

A comparison between (2.64) and (2.49) reveals that the parallelogram arm is kinematically equivalent to a two-link planar arm. The noticeable difference, though, is that the two actuated joints — providing the DOFs of the structure — are located at the base. This will greatly simplify the dynamic model of the structure, as will be seen in Sect. 7.3.3.

2.9.3 Spherical Arm

Consider the spherical arm in Fig. 2.22, where the link frames have been illustrated. Notice that the origin of Frame 0 was located at the intersection of z_0 with z_1 so that $d_1 = 0$; analogously, the origin of Frame 2 was located at the intersection between z_1 and z_2. The DH parameters are specified in Table 2.3.

Table 2.3. DH parameters for the spherical arm

Link	a_i	α_i	d_i	ϑ_i
1	0	$-\pi/2$	0	ϑ_1
2	0	$\pi/2$	d_2	ϑ_2
3	0	0	d_3	0

The homogeneous transformation matrices defined in (2.52) are for the single joints:

$$\boldsymbol{A}^0_1(\vartheta_1) = \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \boldsymbol{A}^1_2(\vartheta_2) = \begin{bmatrix} c_2 & 0 & s_2 & 0 \\ s_2 & 0 & -c_2 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}^2_3(d_3) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

[10] Notice that it is not possible to solve (2.64) for $\vartheta_{2'}$ and $\vartheta_{3'}$ since they are constrained by the condition $\vartheta_{2'} + \vartheta_{3'} = \pi$.

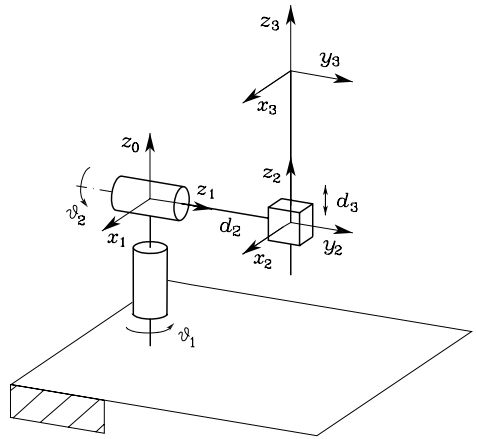

Fig. 2.22. Spherical arm

Computation of the direct kinematics function as in (2.50) yields

$$\boldsymbol{T}_3^0(\boldsymbol{q}) = \boldsymbol{A}_1^0\boldsymbol{A}_2^1\boldsymbol{A}_3^2 = \begin{bmatrix} c_1c_2 & -s_1 & c_1s_2 & c_1s_2d_3 - s_1d_2 \\ s_1c_2 & c_1 & s_1s_2 & s_1s_2d_3 + c_1d_2 \\ -s_2 & 0 & c_2 & c_2d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.65}$$

where $\boldsymbol{q} = [\,\vartheta_1 \quad \vartheta_2 \quad d_3\,]^T$. Notice that the third joint does not obviously influence the rotation matrix. Further, the orientation of the unit vector $\boldsymbol{y}_3^0$ is uniquely determined by the first joint, since the revolute axis of the second joint z_1 is parallel to axis y_3. Different from the previous structures, in this case Frame 3 can represent an end-effector frame of unit vectors $(\boldsymbol{n}_e, \boldsymbol{s}_e, \boldsymbol{a}_e)$, i.e., $\boldsymbol{T}_e^3 = \boldsymbol{I}_4$.

2.9.4 Anthropomorphic Arm

Consider the anthropomorphic arm in Fig. 2.23. Notice how this arm corresponds to a two-link planar arm with an additional rotation about an axis of the plane. In this respect, the parallelogram arm could be used in lieu of the two-link planar arm, as found in some industrial robots with an anthropomorphic structure.

The link frames have been illustrated in the figure. As for the previous structure, the origin of Frame 0 was chosen at the intersection of z_0 with z_1 ($d_1 = 0$); further, z_1 and z_2 are parallel and the choice of axes x_1 and x_2 was made as for the two-link planar arm. The DH parameters are specified in Table 2.4.

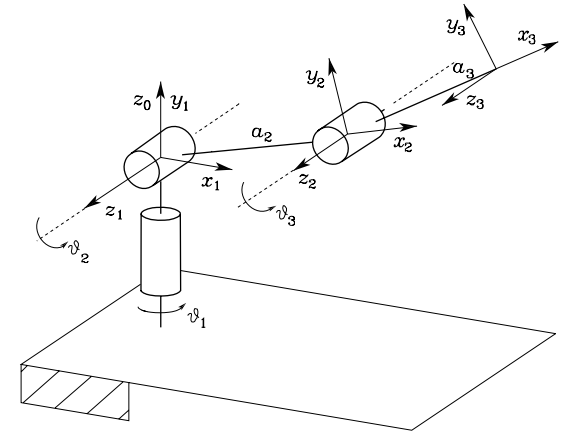

Fig. 2.23. Anthropomorphic arm

Table 2.4. DH parameters for the anthropomorphic arm

Link	a_i	α_i	d_i	ϑ_i
1	0	$\pi/2$	0	ϑ_1
2	a_2	0	0	ϑ_2
3	a_3	0	0	ϑ_3

The homogeneous transformation matrices defined in (2.52) are for the single joints:

$$\boldsymbol{A}_1^0(\vartheta_1) = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_i^{i-1}(\vartheta_i) = \begin{bmatrix} c_i & -s_i & 0 & a_ic_i \\ s_i & c_i & 0 & a_is_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad i = 2, 3.$$

Computation of the direct kinematics function as in (2.50) yields

$$\boldsymbol{T}_3^0(\boldsymbol{q}) = \boldsymbol{A}_1^0\boldsymbol{A}_2^1\boldsymbol{A}_3^2 = \begin{bmatrix} c_1c_{23} & -c_1s_{23} & s_1 & c_1(a_2c_2 + a_3c_{23}) \\ s_1c_{23} & -s_1s_{23} & -c_1 & s_1(a_2c_2 + a_3c_{23}) \\ s_{23} & c_{23} & 0 & a_2s_2 + a_3s_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.66}$$

where $\boldsymbol{q} = [\,\vartheta_1 \quad \vartheta_2 \quad \vartheta_3\,]^T$. Since z_3 is aligned with z_2, Frame 3 does not coincide with a possible end-effector frame as in Fig. 2.13, and a proper constant transformation would be needed.

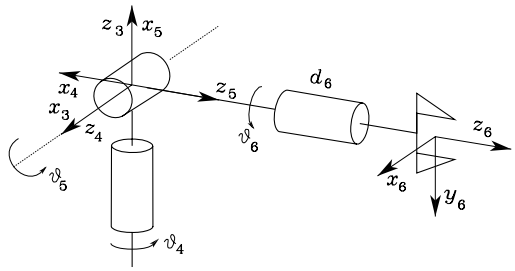

Fig. 2.24. Spherical wrist

2.9.5 Spherical Wrist

Consider a particular type of structure consisting just of the wrist of Fig. 2.24. Joint variables were numbered progressively starting from 4, since such a wrist is typically thought of as mounted on a three-DOF arm of a six-DOF manipulator. It is worth noticing that the wrist is spherical since all revolute axes intersect at a single point. Once z_3, z_4, z_5 have been established, and x_3 has been chosen, there is an indeterminacy on the directions of x_4 and x_5. With reference to the frames indicated in Fig. 2.24, the DH parameters are specified in Table 2.5.

Table 2.5. DH parameters for the spherical wrist

Link	a_i	α_i	d_i	ϑ_i
4	0	$-\pi/2$	0	ϑ_4
5	0	$\pi/2$	0	ϑ_5
6	0	0	d_6	ϑ_6

The homogeneous transformation matrices defined in (2.52) are for the single joints:

$$\boldsymbol{A}_4^3(\vartheta_4) = \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \boldsymbol{A}_5^4(\vartheta_5) = \begin{bmatrix} c_5 & 0 & s_5 & 0 \\ s_5 & 0 & -c_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{A}_6^5(\vartheta_6) = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

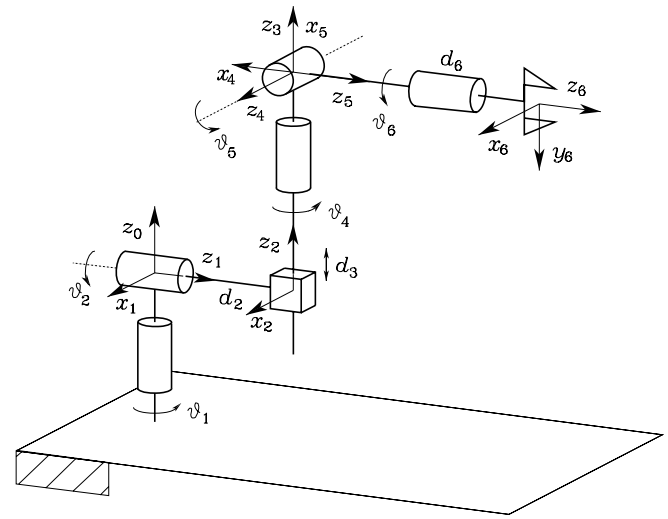

Fig. 2.25. Stanford manipulator

Computation of the direct kinematics function as in (2.50) yields

$$\boldsymbol{T}_6^3(\boldsymbol{q}) = \boldsymbol{A}_4^3\boldsymbol{A}_5^4\boldsymbol{A}_6^5 = \begin{bmatrix} c_4c_5c_6 - s_4s_6 & -c_4c_5s_6 - s_4c_6 & c_4s_5 & c_4s_5d_6 \\ s_4c_5c_6 + c_4s_6 & -s_4c_5s_6 + c_4c_6 & s_4s_5 & s_4s_5d_6 \\ -s_5c_6 & s_5s_6 & c_5 & c_5d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.67}$$

where $\boldsymbol{q} = [\,\vartheta_4 \quad \vartheta_5 \quad \vartheta_6\,]^T$. Notice that, as a consequence of the choice made for the coordinate frames, the block matrix $\boldsymbol{R}_6^3$ that can be extracted from $\boldsymbol{T}_6^3$ coincides with the rotation matrix of Euler angles (2.18) previously derived, that is, ϑ_4, ϑ_5, ϑ_6 constitute the set of ZYZ angles with respect to the reference frame O_3–$x_3y_3z_3$. Moreover, the unit vectors of Frame 6 coincide with the unit vectors of a possible end-effector frame according to Fig. 2.13.

2.9.6 Stanford Manipulator

The so-called Stanford manipulator is composed of a spherical arm and a spherical wrist (Fig. 2.25). Since Frame 3 of the spherical arm coincides with Frame 3 of the spherical wrist, the direct kinematics function can be obtained via simple composition of the transformation matrices (2.65), (2.67) of the previous examples, i.e.,

$$\boldsymbol{T}_6^0 = \boldsymbol{T}_3^0\boldsymbol{T}_6^3 = \begin{bmatrix} \boldsymbol{n}^0 & \boldsymbol{s}^0 & \boldsymbol{a}^0 & \boldsymbol{p}^0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Carrying out the products yields

$$\boldsymbol{p}_6^0 = \begin{bmatrix} c_1 s_2 d_3 - s_1 d_2 + \big(c_1(c_2c_4s_5 + s_2c_5) - s_1 s_4 s_5\big) d_6 \\ s_1 s_2 d_3 + c_1 d_2 + \big(s_1(c_2c_4s_5 + s_2c_5) + c_1 s_4 s_5\big) d_6 \\ c_2 d_3 + (-s_2c_4s_5 + c_2c_5) d_6 \end{bmatrix} \tag{2.68}$$

for the end-effector position, and

$$\begin{aligned} \boldsymbol{n}_6^0 &= \begin{bmatrix} c_1\big(c_2(c_4c_5c_6 - s_4s_6) - s_2s_5c_6\big) - s_1(s_4c_5c_6 + c_4s_6) \\ s_1\big(c_2(c_4c_5c_6 - s_4s_6) - s_2s_5c_6\big) + c_1(s_4c_5c_6 + c_4s_6) \\ -s_2(c_4c_5c_6 - s_4s_6) - c_2s_5c_6 \end{bmatrix} \\ \boldsymbol{s}_6^0 &= \begin{bmatrix} c_1\big(-c_2(c_4c_5s_6 + s_4c_6) + s_2s_5s_6\big) - s_1(-s_4c_5s_6 + c_4c_6) \\ s_1\big(-c_2(c_4c_5s_6 + s_4c_6) + s_2s_5s_6\big) + c_1(-s_4c_5s_6 + c_4c_6) \\ s_2(c_4c_5s_6 + s_4c_6) + c_2s_5s_6 \end{bmatrix} \\ \boldsymbol{a}_6^0 &= \begin{bmatrix} c_1(c_2c_4s_5 + s_2c_5) - s_1s_4s_5 \\ s_1(c_2c_4s_5 + s_2c_5) + c_1s_4s_5 \\ -s_2c_4s_5 + c_2c_5 \end{bmatrix} \end{aligned} \tag{2.69}$$

for the end-effector orientation.

A comparison of the vector $\boldsymbol{p}_6^0$ in (2.68) with the vector $\boldsymbol{p}_3^0$ in (2.65) relative to the sole spherical arm reveals the presence of additional contributions due to the choice of the origin of the end-effector frame at a distance d_6 from the origin of Frame 3 along the direction of $\boldsymbol{a}_6^0$. In other words, if it were $d_6 = 0$, the position vector would be the same. This feature is of fundamental importance for the solution of the inverse kinematics for this manipulator, as will be seen later.

2.9.7 Anthropomorphic Arm with Spherical Wrist

A comparison between Fig. 2.23 and Fig. 2.24 reveals that the direct kinematics function cannot be obtained by multiplying the transformation matrices $\boldsymbol{T}_3^0$ and $\boldsymbol{T}_6^3$, since Frame 3 of the anthropomorphic arm cannot coincide with Frame 3 of the spherical wrist.

Direct kinematics of the entire structure can be obtained in two ways. One consists of interposing a constant transformation matrix between $\boldsymbol{T}_3^0$ and $\boldsymbol{T}_6^3$ which allows the alignment of the two frames. The other refers to the Denavit–Hartenberg operating procedure with the frame assignment for the entire structure illustrated in Fig. 2.26. The DH parameters are specified in Table 2.6.

Since Rows 3 and 4 differ from the corresponding rows of the tables for the two single structures, the relative homogeneous transformation matrices $\boldsymbol{A}_3^2$ and $\boldsymbol{A}_4^3$ have to be modified into

$$\boldsymbol{A}_3^2(\vartheta_3) = \begin{bmatrix} c_3 & 0 & s_3 & 0 \\ s_3 & 0 & -c_3 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \boldsymbol{A}_4^3(\vartheta_4) = \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

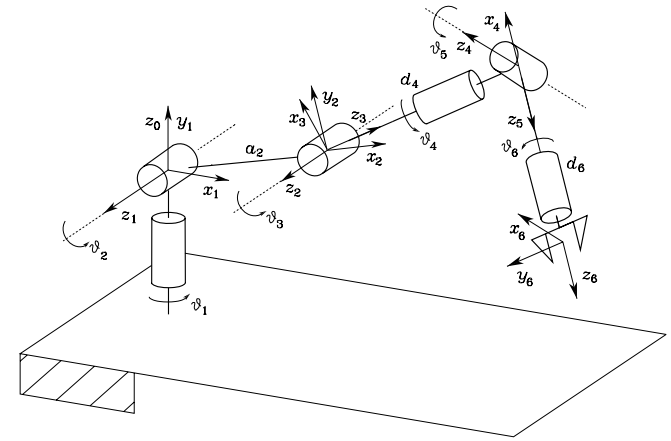

Fig. 2.26. Anthropomorphic arm with spherical wrist

Table 2.6. DH parameters for the anthropomorphic arm with spherical wrist

Link	a_i	α_i	d_i	ϑ_i
1	0	$\pi/2$	0	ϑ_1
2	a_2	0	0	ϑ_2
3	0	$\pi/2$	0	ϑ_3
4	0	$-\pi/2$	d_4	ϑ_4
5	0	$\pi/2$	0	ϑ_5
6	0	0	d_6	ϑ_6

while the other transformation matrices remain the same. Computation of the direct kinematics function leads to expressing the position and orientation of the end-effector frame as:

$$\boldsymbol{p}_6^0 = \begin{bmatrix} a_2c_1c_2 + d_4c_1s_{23} + d_6\big(c_1(c_{23}c_4s_5 + s_{23}c_5) + s_1s_4s_5\big) \\ a_2s_1c_2 + d_4s_1s_{23} + d_6\big(s_1(c_{23}c_4s_5 + s_{23}c_5) - c_1s_4s_5\big) \\ a_2s_2 - d_4c_{23} + d_6(s_{23}c_4s_5 - c_{23}c_5) \end{bmatrix} \tag{2.70}$$

and

$$\begin{aligned} \boldsymbol{n}_6^0 &= \begin{bmatrix} c_1\big(c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6\big) + s_1(s_4c_5c_6 + c_4s_6) \\ s_1\big(c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6\big) - c_1(s_4c_5c_6 + c_4s_6) \\ s_{23}(c_4c_5c_6 - s_4s_6) + c_{23}s_5c_6 \end{bmatrix} \\ \boldsymbol{s}_6^0 &= \begin{bmatrix} c_1\big(-c_{23}(c_4c_5s_6 + s_4c_6) + s_{23}s_5s_6\big) + s_1(-s_4c_5s_6 + c_4c_6) \\ s_1\big(-c_{23}(c_4c_5s_6 + s_4c_6) + s_{23}s_5s_6\big) - c_1(-s_4c_5s_6 + c_4c_6) \\ -s_{23}(c_4c_5s_6 + s_4c_6) - c_{23}s_5s_6 \end{bmatrix} \\ \boldsymbol{a}_6^0 &= \begin{bmatrix} c_1(c_{23}c_4s_5 + s_{23}c_5) + s_1s_4s_5 \\ s_1(c_{23}c_4s_5 + s_{23}c_5) - c_1s_4s_5 \\ s_{23}c_4s_5 - c_{23}c_5 \end{bmatrix}. \end{aligned} \tag{2.71}$$

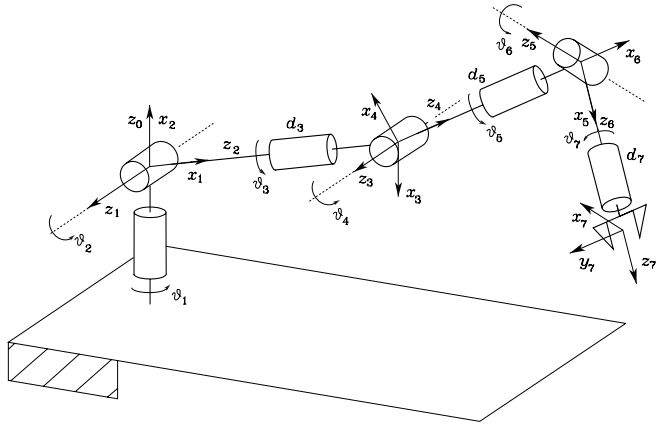

Fig. 2.27. DLR manipulator

By setting $d_6 = 0$, the position of the wrist axes intersection is obtained. In that case, the vector $\boldsymbol{p}^0$ in (2.70) corresponds to the vector $\boldsymbol{p}_3^0$ for the sole anthropomorphic arm in (2.66), because d_4 gives the length of the forearm (a_3) and axis x_3 in Fig. 2.26 is rotated by $\pi/2$ with respect to axis x_3 in Fig. 2.23.

2.9.8 DLR Manipulator

Consider the DLR manipulator, whose development is at the basis of the realization of the robot in Fig. 1.30; it is characterized by seven DOFs and as such it is inherently redundant. This manipulator has two possible configurations for the outer three joints (wrist). With reference to a spherical wrist similar to that introduced in Sect. 2.9.5, the resulting kinematic structure is illustrated in Fig. 2.27, where the frames attached to the links are evidenced.

As in the case of the spherical arm, notice that the origin of Frame 0 has been chosen so as to zero d_1. The DH parameters are specified in Table 2.7.

Table 2.7. DH parameters for the DLR manipulator

Link	a_i	α_i	d_i	ϑ_i
1	0	$\pi/2$	0	ϑ_1
2	0	$\pi/2$	0	ϑ_2
3	0	$\pi/2$	d_3	ϑ_3
4	0	$\pi/2$	0	ϑ_4
5	0	$\pi/2$	d_5	ϑ_5
6	0	$\pi/2$	0	ϑ_6
7	0	0	d_7	ϑ_7

The generic homogeneous transformation matrix defined in (2.52) is ($\alpha_i = \pi/2$)

$$\boldsymbol{A}_i^{i-1} = \begin{bmatrix} c_i & 0 & s_i & 0 \\ s_i & 0 & -c_i & 0 \\ 0 & 1 & 0 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad i = 1, \ldots, 6 \tag{2.72}$$

while, since $\alpha_7 = 0$, it is

$$\boldsymbol{A}_7^6 = \begin{bmatrix} c_7 & -s_7 & 0 & 0 \\ s_7 & c_7 & 0 & 0 \\ 0 & 0 & 1 & d_7 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.73}$$

The direct kinematics function, computed as in (2.50), leads to the following expressions for the end-effector frame

$$\boldsymbol{p}_7^0 = \begin{bmatrix} d_3 x_{d_3} + d_5 x_{d_5} + d_7 x_{d_7} \\ d_3 y_{d_3} + d_5 y_{d_5} + d_7 y_{d_7} \\ d_3 z_{d_3} + d_5 z_{d_5} + d_7 z_{d_7} \end{bmatrix} \tag{2.74}$$

with

$$\begin{aligned}
x_{d_3} &= c_1 s_2 \\
x_{d_5} &= c_1(c_2 c_3 s_4 - s_2 c_4) + s_1 s_3 s_4 \\
x_{d_7} &= c_1(c_2 k_1 + s_2 k_2) + s_1 k_3 \\
y_{d_3} &= s_1 s_2 \\
y_{d_5} &= s_1(c_2 c_3 s_4 - s_2 c_4) - c_1 s_3 s_4 \\
y_{d_7} &= s_1(c_2 k_1 + s_2 k_2) - c_1 k_3 \\
z_{d_3} &= -c_2 \\
z_{d_5} &= c_2 c_4 + s_2 c_3 s_4 \\
z_{d_7} &= s_2(c_3(c_4 c_5 s_6 - s_4 c_6) + s_3 s_5 s_6) - c_2 k_2,
\end{aligned}$$

where

$$\begin{aligned}
k_1 &= c_3(c_4 c_5 s_6 - s_4 c_6) + s_3 s_5 s_6 \\
k_2 &= s_4 c_5 s_6 + c_4 c_6 \\
k_3 &= s_3(c_4 c_5 s_6 - s_4 c_6) - c_3 s_5 s_6.
\end{aligned}$$

Furthermore, the end-effector frame orientation can be derived as

$$\boldsymbol{n}_7^0 = \begin{bmatrix} ((x_a c_5 + x_c s_5) c_6 + x_b s_6) c_7 + (x_a s_5 - x_c c_5) s_7 \\ ((y_a c_5 + y_c s_5) c_6 + y_b s_6) c_7 + (y_a s_5 - y_c c_5) s_7 \\ (z_a c_6 + z_c s_6) c_7 + z_b s_7 \end{bmatrix}$$

$$\boldsymbol{s}_7^0 = \begin{bmatrix} -((x_a c_5 + x_c s_5)c_6 + x_b s_6)s_7 + (x_a s_5 - x_c c_5)c_7 \\ -((y_a c_5 + y_c s_5)c_6 + y_b s_6)s_7 + (y_a s_5 - y_c c_5)c_7 \\ -(z_a c_6 + z_c s_6)s_7 + z_b c_7 \end{bmatrix} \qquad (2.75)$$

$$\boldsymbol{a}_7^0 = \begin{bmatrix} (x_a c_5 + x_c s_5)s_6 - x_b c_6 \\ (y_a c_5 + y_c s_5)s_6 - y_b c_6 \\ z_a s_6 - z_c c_6 \end{bmatrix},$$

where

$$\begin{aligned} x_a &= (c_1 c_2 c_3 + s_1 s_3)c_4 + c_1 s_2 s_4 \\ x_b &= (c_1 c_2 c_3 + s_1 s_3)s_4 - c_1 s_2 c_4 \\ x_c &= c_1 c_2 s_3 - s_1 c_3 \\ y_a &= (s_1 c_2 c_3 - c_1 s_3)c_4 + s_1 s_2 s_4 \\ y_b &= (s_1 c_2 c_3 - c_1 s_3)s_4 - s_1 s_2 c_4 \\ y_c &= s_1 c_2 s_3 + c_1 c_3 \\ z_a &= (s_2 c_3 c_4 - c_2 s_4)c_5 + s_2 s_3 s_5 \\ z_b &= (s_2 c_3 s_4 + c_2 c_4)s_5 - s_2 s_3 c_5 \\ z_c &= s_2 c_3 s_4 + c_2 c_4. \end{aligned} \qquad (2.76)$$

As in the case of the anthropomorphic arm with spherical wrist, it occurs that Frame 4 cannot coincide with the base frame of the wrist.

Finally, consider the possibility to mount a different type of spherical wrist, where Joint 7 is so that $\alpha_7 = \pi/2$. In such a case, the computation of the direct kinematics function changes, since the seventh row of the kinematic parameters table changes. In particular, notice that, since $d_7 = 0$, $a_7 \neq 0$, then

$$\boldsymbol{A}_7^6 = \begin{bmatrix} c_7 & 0 & s_7 & a_7 c_7 \\ s_7 & 0 & -c_7 & a_7 s_7 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \qquad (2.77)$$

It follows, however, that Frame 7 does not coincide with the end-effector frame, as already discussed for the three-link planar arm, since the approach unit vector $\boldsymbol{a}_7^0$ is aligned with x_7.

2.9.9 Humanoid Manipulator

The term humanoid refers to a robot showing a kinematic structure similar to that of the human body. It is commonly thought that the most relevant feature of humanoid robots is biped locomotion. However, in detail, a humanoid manipulator refers to an articulated structure with a kinematics analogous to

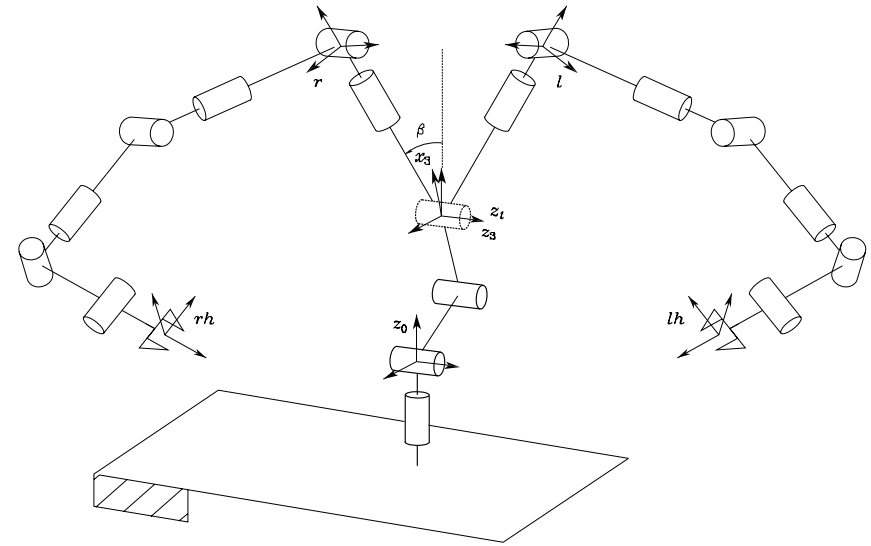

Fig. 2.28. Humanoid manipulator

that of the human body upper part: torso, arms, end-effectors similar to human hands and a 'head' which, eventually, includes an artificial vision system — see Chap. 10.

For the humanoid manipulator in Fig. 1.33, it is worth noticing the presence of two end-effectors (where the 'hands' are mounted), while the arms consist of two DLR manipulators, introduced in the previous section, each with seven DOFs. In particular, consider the configuration where the last joint is so that $\alpha_7 = \pi/2$.

To simplify, the kinematic structure allowing the articulation of the robot's head in Fig. 1.33. The torso can be modelled as an anthropomorphic arm (three DOFs), for a total of seventeen DOFs.

Further, a connecting device exists between the end-effector of the anthropomorphic torso and the base frames of the two manipulators. Such device permits keeping the 'chest' of the humanoid manipulator always orthogonal to the ground. With reference to Fig. 2.28, this device is represented by a further joint, located at the end of the torso. Hence, the corresponding parameter ϑ_4 does not constitute a DOF, yet it varies so as to compensate Joints 2 and 3 rotations of the anthropomorphic torso.

To compute the direct kinematics function, it is possible to resort to a DH parameters table for each of the two tree kinematic structures, which can be identified from the base of the manipulator to each of the two end-effectors. Similarly to the case of mounting a spherical wrist onto an anthropomorphic arm, this implies the change of some rows of the transformation matrices of

those manipulators, described in the previous sections, constituting the torso and the arms.

Alternatively, it is possible to consider intermediate transformation matrices between the relevant structures. In detail, as illustrated in Fig. 2.28, if t denotes the frame attached to the torso, r and l the base frames, respectively, of the right arm and the left arm, and rh and lh the frames attached to the two hands (end-effectors), it is possible to compute for the right arm and the left arm, respectively:

$$\boldsymbol{T}^0_{rh} = \boldsymbol{T}^0_3 \, \boldsymbol{T}^3_t \, \boldsymbol{T}^t_r \boldsymbol{T}^r_{rh} \tag{2.78}$$

$$\boldsymbol{T}^0_{lh} = \boldsymbol{T}^0_3 \, \boldsymbol{T}^3_t \, \boldsymbol{T}^t_l \boldsymbol{T}^l_{lh} \tag{2.79}$$

where the matrix $\boldsymbol{T}^3_t$ describes the transformation imposed by the motion of Joint 4 (dashed line in Fig. 2.28), located at the end-effector of the torso. Frame 4 coincides with Frame t in Fig. 2.27. In view of the property of parameter ϑ_4, it is $\vartheta_4 = -\vartheta_2 - \vartheta_3$, and thus

$$\boldsymbol{T}^3_t = \begin{bmatrix} c_{23} & s_{23} & 0 & 0 \\ -s_{23} & c_{23} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The matrix $\boldsymbol{T}^0_3$ is given by (2.66), whereas the matrices $\boldsymbol{T}^t_r$ and $\boldsymbol{T}^t_l$ relating the torso end-effector frame to the base frames of the two manipulators have constant values. With reference to Fig. 2.28, the elements of these matrices depend on the angle β and on the distances between the origin of Frame t and the origins of Frames r and l. Finally, the expressions of the matrices $\boldsymbol{T}^r_{rh}$ and $\boldsymbol{T}^l_{lh}$ must be computed by considering the change in the seventh row of the DH parameters table of the DLR manipulator, so as to account for the different kinematic structure of the wrist (see Problem 2.14).

2.10 Joint Space and Operational Space

As described in the previous sections, the direct kinematics equation of a manipulator allows the position and orientation of the end-effector frame to be expressed as a function of the joint variables with respect to the base frame.

If a task is to be specified for the end-effector, it is necessary to assign the end-effector position and orientation, eventually as a function of time (trajectory). This is quite easy for the position. On the other hand, specifying the orientation through the unit vector triplet $(\boldsymbol{n}_e, \boldsymbol{s}_e, \boldsymbol{a}_e)$[11] is quite difficult, since their nine components must be guaranteed to satisfy the orthonormality constraints imposed by (2.4) at each time instant. This problem will be resumed in Chap. 4.

[11] To simplify, the indication of the reference frame in the superscript is omitted.

The problem of describing end-effector orientation admits a natural solution if one of the above minimal representations is adopted. In this case, indeed, a motion trajectory can be assigned to the set of angles chosen to represent orientation.

Therefore, the position can be given by a minimal number of coordinates with regard to the geometry of the structure, and the orientation can be specified in terms of a minimal representation (Euler angles) describing the rotation of the end-effector frame with respect to the base frame. In this way, it is possible to describe the end-effector pose by means of the $(m \times 1)$ vector, with $m \leq n$,

$$\boldsymbol{x}_e = \begin{bmatrix} \boldsymbol{p}_e \\ \boldsymbol{\phi}_e \end{bmatrix} \tag{2.80}$$

where $\boldsymbol{p}_e$ describes the end-effector position and $\boldsymbol{\phi}_e$ its orientation.

This representation of position and orientation allows the description of an end-effector task in terms of a number of inherently independent parameters. The vector $\boldsymbol{x}_e$ is defined in the space in which the manipulator task is specified; hence, this space is typically called *operational space*. On the other hand, the *joint space* (configuration space) denotes the space in which the $(n \times 1)$ vector of joint variables

$$\boldsymbol{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}, \tag{2.81}$$

is defined; it is $q_i = \vartheta_i$ for a revolute joint and $q_i = d_i$ for a prismatic joint. Accounting for the dependence of position and orientation from the joint variables, the direct kinematics equation can be written in a form other than (2.50), i.e.,

$$\boldsymbol{x}_e = \boldsymbol{k}(\boldsymbol{q}). \tag{2.82}$$

The $(m \times 1)$ vector function $\boldsymbol{k}(\cdot)$ — nonlinear in general — allows computation of the operational space variables from the knowledge of the joint space variables.

It is worth noticing that the dependence of the orientation components of the function $\boldsymbol{k}(\boldsymbol{q})$ in (2.82) on the joint variables is not easy to express except for simple cases. In fact, in the most general case of a six-dimensional operational space ($m = 6$), the computation of the three components of the function $\boldsymbol{\phi}_e(\boldsymbol{q})$ cannot be performed in closed form but goes through the computation of the elements of the rotation matrix, i.e., $\boldsymbol{n}_e(\boldsymbol{q})$, $\boldsymbol{s}_e(\boldsymbol{q})$, $\boldsymbol{a}_e(\boldsymbol{q})$. The equations that allow the determination of the Euler angles from the triplet of unit vectors $\boldsymbol{n}_e$, $\boldsymbol{s}_e$, $\boldsymbol{a}_e$ were given in Sect. 2.4.

Example 2.5

Consider again the three-link planar arm in Fig. 2.20. The geometry of the structure suggests that the end-effector position is determined by the two coordinates p_x and p_y, while its orientation is determined by the angle ϕ formed by the end-effector with the axis x_0. Expressing these operational variables as a function of the joint variables, the two position coordinates are given by the first two elements of the fourth column of the homogeneous transformation matrix (2.63), while the orientation angle is simply given by the sum of joint variables. In sum, the direct kinematics equation can be written in the form

$$\boldsymbol{x}_e = \begin{bmatrix} p_x \\ p_y \\ \phi \end{bmatrix} = \boldsymbol{k}(\boldsymbol{q}) = \begin{bmatrix} a_1 c_1 + a_2 c_{12} + a_3 c_{123} \\ a_1 s_1 + a_2 s_{12} + a_3 s_{123} \\ \vartheta_1 + \vartheta_2 + \vartheta_3 \end{bmatrix}. \tag{2.83}$$

This expression shows that three joint space variables allow specification of at most three independent operational space variables. On the other hand, if orientation is of no concern, it is $\boldsymbol{x}_e = [\,p_x \quad p_y\,]^T$ and there is *kinematic redundancy* of DOFs with respect to a pure positioning end-effector task; this concept will be dealt with in detail afterwards.

2.10.1 Workspace

With reference to the operational space, an index of robot performance is the so-called *workspace*; this is the region described by the origin of the end-effector frame when all the manipulator joints execute all possible motions. It is often customary to distinguish between *reachable* workspace and *dexterous* workspace. The latter is the region that the origin of the end-effector frame can describe while attaining different orientations, while the former is the region that the origin of the end-effector frame can reach with at least one orientation. Obviously, the dexterous workspace is a subspace of the reachable workspace. A manipulator with less than six DOFs cannot take any arbitrary position and orientation in space.

The workspace is characterized by the manipulator geometry and the mechanical joint limits. For an n-DOF manipulator, the reachable workspace is the geometric locus of the points that can be achieved by considering the direct kinematics equation for the sole position part, i.e.,

$$\boldsymbol{p}_e = \boldsymbol{p}_e(\boldsymbol{q}) \qquad q_{im} \leq q_i \leq q_{iM} \quad i = 1, \ldots, n,$$

where q_{im} (q_{iM}) denotes the minimum (maximum) limit at Joint i. This volume is finite, closed, connected — $\boldsymbol{p}_e(\boldsymbol{q})$ is a continuous function — and thus is defined by its bordering surface. Since the joints are revolute or prismatic, it is easy to recognize that this surface is constituted by surface elements of planar, spherical, toroidal and cylindrical type. The manipulator workspace

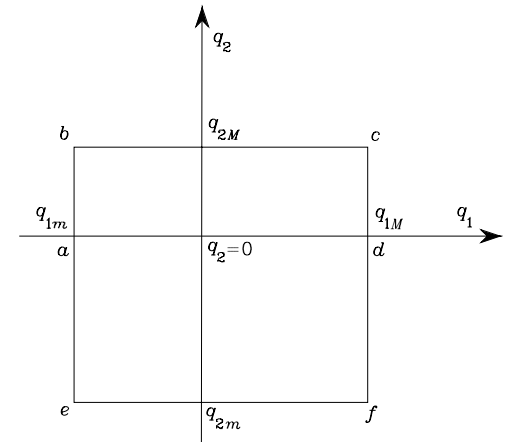

Fig. 2.29. Region of admissible configurations for a two-link arm

(without end-effector) is reported in the data sheet given by the robot manufacturer in terms of a top view and a side view. It represents a basic element to evaluate robot performance for a desired application.

Example 2.6

Consider the simple two-link planar arm. If the mechanical joint limits are known, the arm can attain all the joint space configurations corresponding to the points in the rectangle in Fig. 2.29.

The reachable workspace can be derived via a graphical construction of the image of the rectangle perimeter in the plane of the arm. To this purpose, it is worth considering the images of the segments ab, bc, cd, da, ae, ef, fd. Along the segments ab, bc, cd, ae, ef, fd a loss of mobility occurs due to a joint limit; a loss of mobility occurs also along the segment ad because the arm and forearm are aligned.[12] Further, a change of the arm posture occurs at points a and d: for $q_2 > 0$ the *elbow-down* posture is obtained, while for $q_2 < 0$ the arm is in the *elbow-up* posture.

In the plane of the arm, start drawing the arm in configuration A corresponding to q_{1m} and $q_2 = 0$ (a); then, the segment ab describing motion from $q_2 = 0$ to q_{2M} generates the arc AB; the subsequent arcs BC, CD, DA, AE, EF, FD are generated in a similar way (Fig. 2.30). The external contour of the area $CDAEFHC$ delimits the requested workspace. Further, the area $BCDAB$ is relative to elbow-down postures while the area $DAEFD$ is relative to elbow-up postures; hence, the points in the area $BADHB$ are reachable by the end-effector with both postures.

[12] In the following chapter, it will be seen that this configuration characterizes a kinematic *singularity* of the arm.

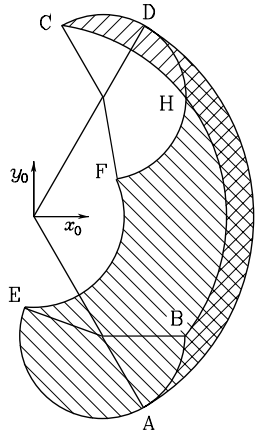

Fig. 2.30. Workspace of a two-link planar arm

In a real manipulator, for a given set of joint variables, the actual values of the operational space variables deviate from those computed via direct kinematics. The direct kinematics equation has indeed a dependence from the DH parameters which is not explicit in (2.82). If the mechanical dimensions of the structure differ from the corresponding parameter of the table because of mechanical tolerances, a deviation arises between the position reached in the assigned posture and the position computed via direct kinematics. Such a deviation is defined *accuracy*; this parameter attains typical values below one millimeter and depends on the structure as well as on manipulator dimensions. Accuracy varies with the end-effector position in the workspace and it is a relevant parameter when robot programming oriented environments are adopted, as will be seen in the last chapter.

Another parameter that is usually listed in the performance data sheet of an industrial robot is *repeatability* which gives a measure of the manipulator's ability to return to a previously reached position; this parameter is relevant for programming an industrial robot by the teaching–by–showing technique which will be presented in Chap. 6. Repeatability depends not only on the characteristics of the mechanical structure but also on the transducers and controller; it is expressed in metric units and is typically smaller than accuracy. For instance, for a manipulator with a maximum reach of 1.5 m, accuracy varies from 0.2 to 1 mm in the workspace, while repeatability varies from 0.02 to 0.2 mm.

2.10.2 Kinematic Redundancy

A manipulator is termed *kinematically redundant* when it has a number of DOFs which is greater than the number of variables that are necessary to describe a given task. With reference to the above-defined spaces, a manipulator is intrinsically redundant when the dimension of the operational space is smaller than the dimension of the joint space ($m < n$). Redundancy is, anyhow, a concept *relative* to the task assigned to the manipulator; a manipulator can be redundant with respect to a task and nonredundant with respect to another. Even in the case of $m = n$, a manipulator can be functionally redundant when only a number of r components of operational space are of concern for the specific task, with $r < m$.

Consider again the three-DOF planar arm of Sect. 2.9.1. If only the end-effector position (in the plane) is specified, that structure presents a functional redundancy ($n = m = 3$, $r = 2$); this is lost when also the end-effector orientation in the plane is specified ($n = m = r = 3$). On the other hand, a four-DOF planar arm is intrinsically redundant ($n = 4$, $m = 3$).

Yet, take the typical industrial robot with six DOFs; such manipulator is not intrinsically redundant ($n = m = 6$), but it can become functionally redundant with regard to the task to execute. Thus, for instance, in a laser-cutting task a functional redundancy will occur since the end-effector rotation about the approach direction is irrelevant to completion of the task ($r = 5$).

At this point, a question should arise spontaneously: Why to intentionally utilize a redundant manipulator? The answer is to recognize that redundancy can provide the manipulator with dexterity and versatility in its motion. The typical example is constituted by the human arm that has *seven* DOFs: three in the shoulder, one in the elbow and three in the wrist, without considering the DOFs in the fingers. This manipulator is intrinsically redundant; in fact, if the base and the hand position and orientation are both fixed — requiring six DOFs — the elbow can be moved, thanks to the additional available DOF. Then, for instance, it is possible to avoid obstacles in the workspace. Further, if a joint of a redundant manipulator reaches its mechanical limit, there might be other joints that allow execution of the prescribed end-effector motion.

A formal treatment of redundancy will be presented in the following chapter.

2.11 Kinematic Calibration

The Denavit–Hartenberg parameters for direct kinematics need to be computed as precisely as possible in order to improve manipulator accuracy. *Kinematic calibration* techniques are devoted to finding accurate estimates of DH parameters from a series of measurements on the manipulator's end-effector pose. Hence, they do not allow direct measurement of the geometric parameters of the structure.

Consider the direct kinematics equation in (2.82) which can be rewritten by emphasizing the dependence of the operational space variables on the fixed DH parameters, besides the joint variables. Let $\boldsymbol{a} = [\, a_1 \quad \dots \quad a_n \,]^T$, $\boldsymbol{\alpha} =$

$[\alpha_1 \quad \dots \quad \alpha_n]^T$, $\boldsymbol{d} = [d_1 \quad \dots \quad d_n]^T$, and $\boldsymbol{\vartheta} = [\theta_1 \quad \dots \quad \theta_n]^T$ denote the vectors of DH parameters for the whole structure; then (2.82) becomes

$$\boldsymbol{x}_e = \boldsymbol{k}(\boldsymbol{a}, \boldsymbol{\alpha}, \boldsymbol{d}, \boldsymbol{\vartheta}). \tag{2.84}$$

The manipulator's end-effector pose should be measured with high precision for the effectiveness of the kinematic calibration procedure. To this purpose a mechanical apparatus can be used that allows the end-effector to be constrained at given poses with a priori known precision. Alternatively, direct measurement systems of object position and orientation in the Cartesian space can be used which employ triangulation techniques.

Let $\boldsymbol{x}_m$ be the measured pose and $\boldsymbol{x}_n$ the nominal pose that can be computed via (2.84) with the nominal values of the parameters $\boldsymbol{a}$, $\boldsymbol{\alpha}$, $\boldsymbol{d}$, $\boldsymbol{\vartheta}$. The nominal values of the fixed parameters are set equal to the design data of the mechanical structure, whereas the nominal values of the joint variables are set equal to the data provided by the position transducers at the given manipulator posture. The deviation $\Delta\boldsymbol{x} = \boldsymbol{x}_m - \boldsymbol{x}_n$ gives a measure of accuracy at the given posture. On the assumption of small deviations, at first approximation, it is possible to derive the following relation from (2.84):

$$\Delta\boldsymbol{x} = \frac{\partial \boldsymbol{k}}{\partial \boldsymbol{a}}\Delta\boldsymbol{a} + \frac{\partial \boldsymbol{k}}{\partial \boldsymbol{\alpha}}\Delta\boldsymbol{\alpha} + \frac{\partial \boldsymbol{k}}{\partial \boldsymbol{d}}\Delta\boldsymbol{d} + \frac{\partial \boldsymbol{k}}{\partial \boldsymbol{\vartheta}}\Delta\boldsymbol{\vartheta} \tag{2.85}$$

where $\Delta\boldsymbol{a}$, $\Delta\boldsymbol{\alpha}$, $\Delta\boldsymbol{d}$, $\Delta\boldsymbol{\vartheta}$ denote the deviations between the values of the parameters of the real structure and the nominal ones. Moreover, $\partial\boldsymbol{k}/\partial\boldsymbol{a}$, $\partial\boldsymbol{k}/\partial\boldsymbol{\alpha}$, $\partial\boldsymbol{k}/\partial\boldsymbol{d}$, $\partial\boldsymbol{k}/\partial\boldsymbol{\vartheta}$ denote the $(m \times n)$ matrices whose elements are the partial derivatives of the components of the direct kinematics function with respect to the single parameters.[13]

Group the parameters in the $(4n \times 1)$ vector $\boldsymbol{\zeta} = [\boldsymbol{a}^T \quad \boldsymbol{\alpha}^T \quad \boldsymbol{d}^T \quad \boldsymbol{\vartheta}^T]^T$. Let $\Delta\boldsymbol{\zeta} = \boldsymbol{\zeta}_m - \boldsymbol{\zeta}_n$ denote the parameter variations with respect to the nominal values, and $\boldsymbol{\Phi} = [\partial\boldsymbol{k}/\partial\boldsymbol{a} \quad \partial\boldsymbol{k}/\partial\boldsymbol{\alpha} \quad \partial\boldsymbol{k}/\partial\boldsymbol{d} \quad \partial\boldsymbol{k}/\partial\boldsymbol{\vartheta}]$ the $(m \times 4n)$ *kinematic calibration matrix* computed for the nominal values of the parameters $\boldsymbol{\zeta}_n$. Then (2.85) can be compactly rewritten as

$$\Delta\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{\zeta}_n)\Delta\boldsymbol{\zeta}. \tag{2.86}$$

It is desired to compute $\Delta\boldsymbol{\zeta}$ starting from the knowledge of $\boldsymbol{\zeta}_n, \boldsymbol{x}_n$ and the measurement of $\boldsymbol{x}_m$. Since (2.86) constitutes a system of m equations into $4n$ unknowns with $m < 4n$, a sufficient number of end-effector pose measurements has to be performed so as to obtain a system of at least $4n$ equations. Therefore, if measurements are made for a number of l poses, (2.86) yields

$$\Delta\bar{\boldsymbol{x}} = \begin{bmatrix} \Delta\boldsymbol{x}_1 \\ \vdots \\ \Delta\boldsymbol{x}_l \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}_1 \\ \vdots \\ \boldsymbol{\Phi}_l \end{bmatrix} \Delta\boldsymbol{\zeta} = \bar{\boldsymbol{\Phi}}\Delta\boldsymbol{\zeta}. \tag{2.87}$$

[13] These matrices are the Jacobians of the transformations between the parameter space and the operational space.

As regards the nominal values of the parameters needed for the computation of the matrices $\boldsymbol{\Phi}_i$, it should be observed that the geometric parameters are constant whereas the joint variables depend on the manipulator configuration at pose i.

In order to avoid ill-conditioning of matrix $\bar{\boldsymbol{\Phi}}$, it is advisable to choose l so that $lm \gg 4n$ and then solve (2.87) with a least-squares technique; in this case the solution is of the form

$$\Delta\boldsymbol{\zeta} = (\bar{\boldsymbol{\Phi}}^T\bar{\boldsymbol{\Phi}})^{-1}\bar{\boldsymbol{\Phi}}^T\Delta\bar{\boldsymbol{x}} \tag{2.88}$$

where $(\bar{\boldsymbol{\Phi}}^T\bar{\boldsymbol{\Phi}})^{-1}\bar{\boldsymbol{\Phi}}^T$ is the *left pseudo-inverse* matrix of $\bar{\boldsymbol{\Phi}}$.[14] By computing $\bar{\boldsymbol{\Phi}}$ with the nominal values of the parameters $\boldsymbol{\zeta}_n$, the first parameter *estimate* is given by

$$\boldsymbol{\zeta}' = \boldsymbol{\zeta}_n + \Delta\boldsymbol{\zeta}. \tag{2.89}$$

This is a nonlinear parameter estimate problem and, as such, the procedure should be iterated until $\Delta\boldsymbol{\zeta}$ converges within a given threshold. At each iteration, the calibration matrix $\bar{\boldsymbol{\Phi}}$ is to be updated with the parameter estimates $\boldsymbol{\zeta}'$ obtained via (2.89) at the previous iteration. In a similar manner, the deviation $\Delta\bar{\boldsymbol{x}}$ is to be computed as the difference between the measured values for the l end-effector poses and the corresponding poses computed by the direct kinematics function with the values of the parameters at the previous iteration. As a result of the kinematic calibration procedure, more accurate estimates of the real manipulator geometric parameters as well as possible corrections to make on the joint transducers measurements are obtained.

Kinematic calibration is an operation that is performed by the robot manufacturer to guarantee the accuracy reported in the data sheet. There is another kind of calibration that is performed by the robot user which is needed for the measurement system *start-up* to guarantee that the position transducers data are consistent with the attained manipulator posture. For instance, in the case of incremental (nonabsolute) position transducers, such calibration consists of taking the mechanical structure into a given reference posture (*home*) and initializing the position transducers with the values at that posture.

2.12 Inverse Kinematics Problem

The direct kinematics equation, either in the form (2.50) or in the form (2.82), establishes the functional relationship between the joint variables and the end-effector position and orientation. The *inverse kinematics problem* consists of the determination of the joint variables corresponding to a given end-effector position and orientation. The solution to this problem is of fundamental importance in order to transform the motion specifications, assigned to the end-effector in the operational space, into the corresponding joint space motions that allow execution of the desired motion.

[14] See Sect. A.7 for the definition of the pseudo-inverse of a matrix.

As regards the direct kinematics equation in (2.50), the end-effector position and rotation matrix are computed in a unique manner, once the joint variables are known[15]. On the other hand, the inverse kinematics problem is much more complex for the following reasons:

- The equations to solve are in general nonlinear, and thus it is not always possible to find a *closed-form solution*.
- *Multiple solutions* may exist.
- *Infinite solutions* may exist, e.g., in the case of a kinematically redundant manipulator.
- There might be no *admissible* solutions, in view of the manipulator kinematic structure.

The existence of solutions is guaranteed only if the given end-effector position and orientation belong to the manipulator dexterous workspace.

On the other hand, the problem of multiple solutions depends not only on the number of DOFs but also on the number of non-null DH parameters; in general, the greater the number of non-null parameters, the greater the number of admissible solutions. For a six-DOF manipulator without mechanical joint limits, there are in general up to 16 admissible solutions. Such occurrence demands some criterion to choose among admissible solutions (e.g., the elbow-up/elbow-down case of Example 2.6). The existence of mechanical joint limits may eventually reduce the number of admissible multiple solutions for the real structure.

Computation of closed-form solutions requires either *algebraic intuition* to find those significant equations containing the unknowns or *geometric intuition* to find those significant points on the structure with respect to which it is convenient to express position and/or orientation as a function of a reduced number of unknowns. The following examples will point out the ability required to an inverse kinematics problem solver. On the other hand, in all those cases when there are no — or it is difficult to find — closed-form solutions, it might be appropriate to resort to *numerical solution techniques*; these clearly have the advantage of being applicable to any kinematic structure, but in general they do not allow computation of all admissible solutions. In the following chapter, it will be shown how suitable algorithms utilizing the manipulator Jacobian can be employed to solve the inverse kinematics problem.

2.12.1 Solution of Three-link Planar Arm

Consider the arm shown in Fig. 2.20 whose direct kinematics was given in (2.63). It is desired to find the joint variables ϑ_1, ϑ_2, ϑ_3 corresponding to a given end-effector position and orientation.

[15] In general, this cannot be said for (2.82) too, since the Euler angles are not uniquely defined.

As already pointed out, it is convenient to specify position and orientation in terms of a minimal number of parameters: the two coordinates p_x, p_y and the angle ϕ with axis x_0, in this case. Hence, it is possible to refer to the direct kinematics equation in the form (2.83).

A first *algebraic solution* technique is illustrated below. Having specified the orientation, the relation

$$\phi = \vartheta_1 + \vartheta_2 + \vartheta_3 \tag{2.90}$$

is one of the equations of the system to solve[16]. From (2.63) the following equations can be obtained:

$$p_{Wx} = p_x - a_3 c_\phi = a_1 c_1 + a_2 c_{12} \tag{2.91}$$

$$p_{Wy} = p_y - a_3 s_\phi = a_1 s_1 + a_2 s_{12} \tag{2.92}$$

which describe the position of point W, i.e., the origin of Frame 2; this depends only on the first two angles ϑ_1 and ϑ_2. Squaring and summing (2.91), (2.92) yields

$$p_{Wx}^2 + p_{Wy}^2 = a_1^2 + a_2^2 + 2a_1 a_2 c_2$$

from which

$$c_2 = \frac{p_{Wx}^2 + p_{Wy}^2 - a_1^2 - a_2^2}{2a_1 a_2}.$$

The existence of a solution obviously imposes that $-1 \leq c_2 \leq 1$, otherwise the given point would be outside the arm reachable workspace. Then, set

$$s_2 = \pm\sqrt{1 - c_2^2},$$

where the positive sign is relative to the elbow-down posture and the negative sign to the elbow-up posture. Hence, the angle ϑ_2 can be computed as

$$\vartheta_2 = \mathrm{Atan2}(s_2, c_2).$$

Having determined ϑ_2, the angle ϑ_1 can be found as follows. Substituting ϑ_2 into (2.91), (2.92) yields an algebraic system of two equations in the two unknowns s_1 and c_1, whose solution is

$$s_1 = \frac{(a_1 + a_2 c_2) p_{Wy} - a_2 s_2 p_{Wx}}{p_{Wx}^2 + p_{Wy}^2}$$

$$c_1 = \frac{(a_1 + a_2 c_2) p_{Wx} + a_2 s_2 p_{Wy}}{p_{Wx}^2 + p_{Wy}^2}.$$

In analogy to the above, it is

$$\vartheta_1 = \mathrm{Atan2}(s_1, c_1).$$

[16] If ϕ is not specified, then the arm is redundant and there exist infinite solutions to the inverse kinematics problem.

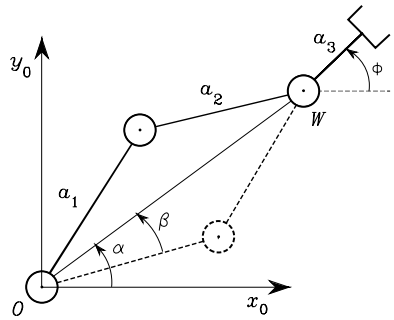

Fig. 2.31. Admissible postures for a two-link planar arm

In the case when $s_2 = 0$, it is obviously $\vartheta_2 = 0, \pi$; as will be shown in the following, in such a posture the manipulator is at a kinematic *singularity*. Yet, the angle ϑ_1 can be determined uniquely, unless $a_1 = a_2$ and it is required $p_{Wx} = p_{Wy} = 0$.

Finally, the angle ϑ_3 is found from (2.90) as

$$\vartheta_3 = \phi - \vartheta_1 - \vartheta_2.$$

An alternative *geometric solution* technique is presented below. As above, the orientation angle is given as in (2.90) and the coordinates of the origin of Frame 2 are computed as in (2.91), (2.92). The application of the cosine theorem to the triangle formed by links a_1, a_2 and the segment connecting points W and O gives

$$p_{Wx}^2 + p_{Wy}^2 = a_1^2 + a_2^2 - 2a_1a_2\cos(\pi - \vartheta_2);$$

the two admissible configurations of the triangle are shown in Fig. 2.31. Observing that $\cos(\pi - \vartheta_2) = -\cos\vartheta_2$ leads to

$$c_2 = \frac{p_{Wx}^2 + p_{Wy}^2 - a_1^2 - a_2^2}{2a_1a_2}.$$

For the existence of the triangle, it must be $\sqrt{p_{Wx}^2 + p_{Wy}^2} \leq a_1 + a_2$. This condition is not satisfied when the given point is outside the arm reachable workspace. Then, under the assumption of admissible solutions, it is

$$\vartheta_2 = \pm\cos^{-1}(c_2);$$

the elbow-up posture is obtained for $\vartheta_2 \in (-\pi, 0)$ while the elbow-down posture is obtained for $\vartheta_2 \in (0, \pi)$.

To find ϑ_1 consider the angles α and β in Fig. 2.31. Notice that the determination of α depends on the sign of p_{Wx} and p_{Wy}; then, it is necessary to compute α as

$$\alpha = \mathrm{Atan2}(p_{Wy}, p_{Wx}).$$

To compute β, applying again the cosine theorem yields

$$c_\beta\sqrt{p_{Wx}^2 + p_{Wy}^2} = a_1 + a_2c_2$$

and resorting to the expression of c_2 given above leads to

$$\beta = \cos^{-1}\left(\frac{p_{Wx}^2 + p_{Wy}^2 + a_1^2 - a_2^2}{2a_1\sqrt{p_{Wx}^2 + p_{Wy}^2}}\right)$$

with $\beta \in (0, \pi)$ so as to preserve the existence of triangles. Then, it is

$$\vartheta_1 = \alpha \pm \beta,$$

where the positive sign holds for $\vartheta_2 < 0$ and the negative sign for $\vartheta_2 > 0$. Finally, ϑ_3 is computed from (2.90).

It is worth noticing that, in view of the substantial equivalence between the two-link planar arm and the parallelogram arm, the above techniques can be formally applied to solve the inverse kinematics of the arm in Sect. 2.9.2.

2.12.2 Solution of Manipulators with Spherical Wrist

Most of the existing manipulators are kinematically simple, since they are typically formed by an arm, of the kind presented above, and a spherical wrist; see the manipulators in Sects. 2.9.6–2.9.8. This choice is partly motivated by the difficulty to find solutions to the inverse kinematics problem in the general case. In particular, a *six*-DOF kinematic structure has closed-form inverse kinematics solutions if:

- three consecutive revolute joint axes intersect at a common point, like for the spherical wrist;
- three consecutive revolute joint axes are parallel.

In any case, algebraic or geometric intuition is required to obtain closed-form solutions.

Inspired by the previous solution to a three-link planar arm, a suitable point along the structure can be found whose position can be expressed both as a function of the given end-effector position and orientation and as a function of a reduced number of joint variables. This is equivalent to articulating the inverse kinematics problem into two subproblems, since the solution for the *position* is *decoupled* from that for the *orientation*.

For a manipulator with spherical wrist, the natural choice is to locate such point W at the intersection of the three terminal revolute axes (Fig. 2.32). In fact, once the end-effector position and orientation are specified in terms of $\boldsymbol{p}_e$ and $\boldsymbol{R}_e = [\,\boldsymbol{n}_e \quad \boldsymbol{s}_e \quad \boldsymbol{a}_e\,]$, the wrist position can be found as

$$\boldsymbol{p}_W = \boldsymbol{p}_e - d_6\boldsymbol{a}_e \tag{2.93}$$

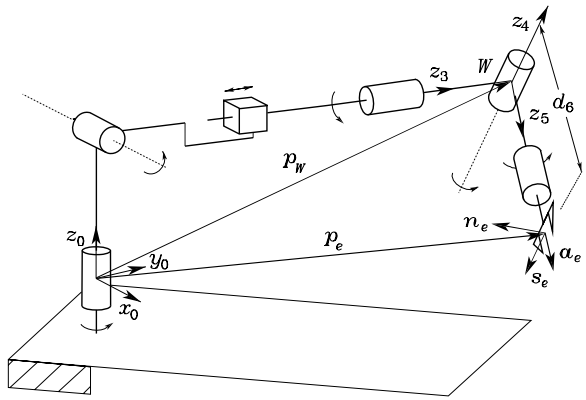

Fig. 2.32. Manipulator with spherical wrist

which is a function of the sole joint variables that determine the arm position[17]. Hence, in the case of a (nonredundant) three-DOF arm, the inverse kinematics can be solved according to the following steps:

- Compute the wrist position $\boldsymbol{p}_W(q_1, q_2, q_3)$ as in (2.93).
- Solve inverse kinematics for (q_1, q_2, q_3).
- Compute $\boldsymbol{R}_3^0(q_1, q_2, q_3)$.
- Compute $\boldsymbol{R}_6^3(\vartheta_4, \vartheta_5, \vartheta_6) = \boldsymbol{R}_3^{0T}\boldsymbol{R}$.
- Solve inverse kinematics for orientation $(\vartheta_4, \vartheta_5, \vartheta_6)$.

Therefore, on the basis of this kinematic decoupling, it is possible to solve the inverse kinematics for the arm separately from the inverse kinematics for the spherical wrist. Below are presented the solutions for two typical arms (spherical and anthropomorphic) as well as the solution for the spherical wrist.

2.12.3 Solution of Spherical Arm

Consider the spherical arm shown in Fig. 2.22, whose direct kinematics was given in (2.65). It is desired to find the joint variables ϑ_1, ϑ_2, d_3 corresponding to a given end-effector position $\boldsymbol{p}_W$.

In order to separate the variables on which $\boldsymbol{p}_W$ depends, it is convenient to express the position of $\boldsymbol{p}_W$ with respect to Frame 1; then, consider the matrix equation

$$(\boldsymbol{A}_1^0)^{-1}\boldsymbol{T}_3^0 = \boldsymbol{A}_2^1\boldsymbol{A}_3^2.$$

[17] Note that the same reasoning was implicitly adopted in Sect. 2.12.1 for the three-link planar arm; $\boldsymbol{p}_W$ described the one-DOF wrist position for the two-DOF arm obtained by considering only the first two links.

Equating the first three elements of the fourth columns of the matrices on both sides yields

$$\boldsymbol{p}_W^1 = \begin{bmatrix} p_{Wx}c_1 + p_{Wy}s_1 \\ -p_{Wz} \\ -p_{Wx}s_1 + p_{Wy}c_1 \end{bmatrix} = \begin{bmatrix} d_3 s_2 \\ -d_3 c_2 \\ d_2 \end{bmatrix} \tag{2.94}$$

which depends only on ϑ_2 and d_3. To solve this equation, set

$$t = \tan\frac{\vartheta_1}{2}$$

so that

$$c_1 = \frac{1-t^2}{1+t^2} \qquad s_1 = \frac{2t}{1+t^2}.$$

Substituting this equation in the third component on the left-hand side of (2.94) gives

$$(d_2 + p_{Wy})t^2 + 2p_{Wx}t + d_2 - p_{Wy} = 0,$$

whose solution is

$$t = \frac{-p_{Wx} \pm \sqrt{p_{Wx}^2 + p_{Wy}^2 - d_2^2}}{d_2 + p_{Wy}}.$$

The two solutions correspond to two different postures. Hence, it is

$$\vartheta_1 = 2\text{Atan2}\left(-p_{Wx} \pm \sqrt{p_{Wx}^2 + p_{Wy}^2 - d_2^2},\ d_2 + p_{Wy}\right).$$

Once ϑ_1 is known, squaring and summing the first two components of (2.94) yields

$$d_3 = \sqrt{(p_{Wx}c_1 + p_{Wy}s_1)^2 + p_{Wz}^2},$$

where only the solution with $d_3 \geq 0$ has been considered. Note that the same value of d_3 corresponds to both solutions for ϑ_1. Finally, if $d_3 \neq 0$, from the first two components of (2.94) it is

$$\frac{p_{Wx}c_1 + p_{Wy}s_1}{-p_{Wz}} = \frac{d_3 s_2}{-d_3 c_2},$$

from which

$$\vartheta_2 = \text{Atan2}(p_{Wx}c_1 + p_{Wy}s_1, p_{Wz}).$$

Notice that, if $d_3 = 0$, then ϑ_2 cannot be uniquely determined.

2.12.4 Solution of Anthropomorphic Arm

Consider the anthropomorphic arm shown in Fig. 2.23. It is desired to find the joint variables ϑ_1, ϑ_2, ϑ_3 corresponding to a given end-effector position $\boldsymbol{p}_W$. Notice that the direct kinematics for $\boldsymbol{p}_W$ is expressed by (2.66) which can

be obtained from (2.70) by setting $d_6 = 0$, $d_4 = a_3$ and replacing ϑ_3 with the angle $\vartheta_3 + \pi/2$ because of the misalignment of the Frames 3 for the structures in Fig. 2.23 and in Fig. 2.26, respectively. Hence, it follows

$$p_{Wx} = c_1(a_2c_2 + a_3c_{23}) \tag{2.95}$$
$$p_{Wy} = s_1(a_2c_2 + a_3c_{23}) \tag{2.96}$$
$$p_{Wz} = a_2s_2 + a_3s_{23}. \tag{2.97}$$

Proceeding as in the case of the two-link planar arm, it is worth squaring and summing (2.95)–(2.97) yielding

$$p_{Wx}^2 + p_{Wy}^2 + p_{Wz}^2 = a_2^2 + a_3^2 + 2a_2a_3c_3$$

from which

$$c_3 = \frac{p_{Wx}^2 + p_{Wy}^2 + p_{Wz}^2 - a_2^2 - a_3^2}{2a_2a_3} \tag{2.98}$$

where the admissibility of the solution obviously requires that $-1 \leq c_3 \leq 1$, or equivalently $|a_2 - a_3| \leq \sqrt{p_{Wx}^2 + p_{Wy}^2 + p_{Wz}^2} \leq a_2 + a_3$, otherwise the wrist point is outside the reachable workspace of the manipulator. Hence it is

$$s_3 = \pm\sqrt{1 - c_3^2} \tag{2.99}$$

and thus

$$\vartheta_3 = \text{Atan2}(s_3, c_3)$$

giving the two solutions, according to the sign of s_3,

$$\vartheta_{3,\text{I}} \in [-\pi, \pi] \tag{2.100}$$
$$\vartheta_{3,\text{II}} = -\vartheta_{3,I}. \tag{2.101}$$

Having determined ϑ_3, it is possible to compute ϑ_2 as follows. Squaring and summing (2.95), (2.96) gives

$$p_{Wx}^2 + p_{Wy}^2 = (a_2c_2 + a_3c_{23})^2$$

from which

$$a_2c_2 + a_3c_{23} = \pm\sqrt{p_{Wx}^2 + p_{Wy}^2}. \tag{2.102}$$

The system of the two Eqs. (2.102), (2.97), for each of the solutions (2.100), (2.101), admits the solutions:

$$c_2 = \frac{\pm\sqrt{p_{Wx}^2 + p_{Wy}^2}(a_2 + a_3c_3) + p_{Wz}a_3s_3}{a_2^2 + a_3^2 + 2a_2a_3c_3} \tag{2.103}$$

$$s_2 = \frac{p_{Wz}(a_2 + a_3c_3) \mp \sqrt{p_{Wx}^2 + p_{Wy}^2}a_3s_3}{a_2^2 + a_3^2 + 2a_2a_3c_3}. \tag{2.104}$$

From (2.103), (2.104) it follows

$$\vartheta_2 = \text{Atan2}(s_2, c_2)$$

which gives the four solutions for ϑ_2, according to the sign of s_3 in (2.99):

$$\vartheta_{2,\text{I}} = \text{Atan2}\Big((a_2 + a_3c_3)p_{Wz} - a_3s_3^+\sqrt{p_{Wx}^2 + p_{Wy}^2},\ (a_2 + a_3c_3)\sqrt{p_{Wx}^2 + p_{Wy}^2} + a_3s_3^+p_{Wz}\Big) \tag{2.105}$$

$$\vartheta_{2,\text{II}} = \text{Atan2}\Big((a_2 + a_3c_3)p_{Wz} + a_3s_3^+\sqrt{p_{Wx}^2 + p_{Wy}^2},\ -(a_2 + a_3c_3)\sqrt{p_{Wx}^2 + p_{Wy}^2} + a_3s_3^+p_{Wz}\Big) \tag{2.106}$$

corresponding to $s_3^+ = \sqrt{1 - c_3^2}$, and

$$\vartheta_{2,\text{III}} = \text{Atan2}\Big((a_2 + a_3c_3)p_{Wz} - a_3s_3^-\sqrt{p_{Wx}^2 + p_{Wy}^2},\ (a_2 + a_3c_3)\sqrt{p_{Wx}^2 + p_{Wy}^2} + a_3s_3^-p_{Wz}\Big) \tag{2.107}$$

$$\vartheta_{2,\text{IV}} = \text{Atan2}\Big((a_2 + a_3c_3)p_{Wz} + a_3s_3^-\sqrt{p_{Wx}^2 + p_{Wy}^2},\ -(a_2 + a_3c_3)\sqrt{p_{Wx}^2 + p_{Wy}^2} + a_3s_3^-p_{Wz}\Big) \tag{2.108}$$

corresponding to $s_3^- = -\sqrt{1 - c_3^2}$.

Finally, to compute ϑ_1, it is sufficient to rewrite (2.95), (2.96), using (2.102), as

$$p_{Wx} = \pm c_1\sqrt{p_{Wx}^2 + p_{Wy}^2}$$
$$p_{Wy} = \pm s_1\sqrt{p_{Wx}^2 + p_{Wy}^2}$$

which, once solved, gives the two solutions:

$$\vartheta_{1,\text{I}} = \text{Atan2}(p_{Wy}, p_{Wx}) \tag{2.109}$$
$$\vartheta_{1,\text{II}} = \text{Atan2}(-p_{Wy}, -p_{Wx}). \tag{2.110}$$

Notice that (2.110) gives[18]

$$\vartheta_{1,\text{II}} = \begin{cases} \text{Atan2}(p_{Wy}, p_{Wx}) - \pi & p_{Wy} \geq 0 \\ \text{Atan2}(p_{Wy}, p_{Wx}) + \pi & p_{Wy} < 0. \end{cases}$$

[18] It is easy to show that $\text{Atan2}(-y, -x) = -\text{Atan2}(y, -x)$ and

$$\text{Atan2}(y, -x) = \begin{cases} \pi - \text{Atan2}(y, x) & y \geq 0 \\ -\pi - \text{Atan2}(y, x) & y < 0. \end{cases}$$

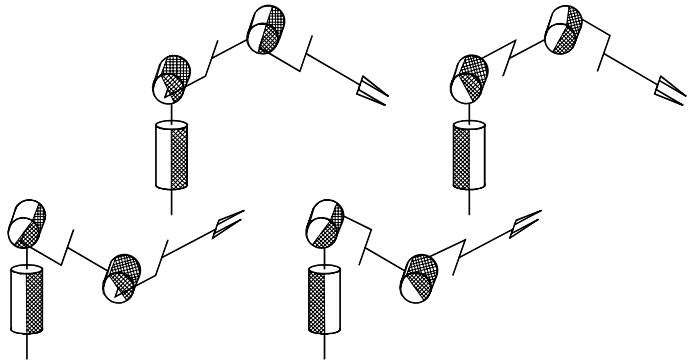

Fig. 2.33. The four configurations of an anthropomorphic arm compatible with a given wrist position

As can be recognized, there exist four solutions according to the values of ϑ_3 in (2.100), (2.101), ϑ_2 in (2.105)–(2.108) and ϑ_1 in (2.109), (2.110):

$$(\vartheta_{1,\mathrm{I}}, \vartheta_{2,\mathrm{I}}, \vartheta_{3,\mathrm{I}}) \quad (\vartheta_{1,\mathrm{I}}, \vartheta_{2,\mathrm{III}}, \vartheta_{3,\mathrm{II}}) \quad (\vartheta_{1,\mathrm{II}}, \vartheta_{2,\mathrm{II}}, \vartheta_{3,\mathrm{I}}) \quad (\vartheta_{1,\mathrm{II}}, \vartheta_{2,\mathrm{IV}}, \vartheta_{3,\mathrm{II}}),$$

which are illustrated in Fig. 2.33: shoulder–right/elbow–up, shoulder–left/elbow–up, shoulder–right/elbow–down, shoulder–left/elbow–down; obviously, the forearm orientation is different for the two pairs of solutions.

Notice finally how it is possible to find the solutions only if at least

$$p_{Wx} \neq 0 \qquad \text{or} \qquad p_{Wy} \neq 0.$$

In the case $p_{Wx} = p_{Wy} = 0$, an infinity of solutions is obtained, since it is possible to determine the joint variables ϑ_2 and ϑ_3 independently of the value of ϑ_1; in the following, it will be seen that the arm in such configuration is kinematically *singular* (see Problem 2.18).

2.12.5 Solution of Spherical Wrist

Consider the spherical wrist shown in Fig. 2.24, whose direct kinematics was given in (2.67). It is desired to find the joint variables ϑ_4, ϑ_5, ϑ_6 corresponding to a given end-effector orientation $\boldsymbol{R}_6^3$. As previously pointed out, these angles constitute a set of Euler angles ZYZ with respect to Frame 3. Hence, having computed the rotation matrix

$$\boldsymbol{R}_6^3 = \begin{bmatrix} n_x^3 & s_x^3 & a_x^3 \\ n_y^3 & s_y^3 & a_y^3 \\ n_z^3 & s_z^3 & a_z^3 \end{bmatrix},$$

from its expression in terms of the joint variables in (2.67), it is possible to compute the solutions directly as in (2.19), (2.20), i.e.,

$$\begin{aligned} \vartheta_4 &= \mathrm{Atan2}(a_y^3, a_x^3) \\ \vartheta_5 &= \mathrm{Atan2}\Big(\sqrt{(a_x^3)^2 + (a_y^3)^2}, a_z^3\Big) \\ \vartheta_6 &= \mathrm{Atan2}(s_z^3, -n_z^3) \end{aligned} \tag{2.111}$$

for $\vartheta_5 \in (0, \pi)$, and

$$\begin{aligned} \vartheta_4 &= \mathrm{Atan2}(-a_y^3, -a_x^3) \\ \vartheta_5 &= \mathrm{Atan2}\Big(-\sqrt{(a_x^3)^2 + (a_y^3)^2}, a_z^3\Big) \\ \vartheta_6 &= \mathrm{Atan2}(-s_z^3, n_z^3) \end{aligned} \tag{2.112}$$

for $\vartheta_5 \in (-\pi, 0)$.

Bibliography

The treatment of kinematics of robot manipulators can be found in several classical robotics texts, such as [180, 10, 200, 217]. Specific texts are [23, 6, 151].

For the descriptions of the orientation of a rigid body, see [187]. Quaternion algebra can be found in [46]; see [204] for the extraction of quaternions from rotation matrices.

The Denavit–Hartenberg convention was first introduced in [60]. A modified version is utilized in [53, 248, 111]. The use of homogeneous transformation matrices for the computation of open-chain manipulator direct kinematics is presented in [181], while in [183] sufficient conditions are given for the closed-form computation of the inverse kinematics problem. For kinematics of closed chains see [144, 111]. The design of the Stanford manipulator is due to [196].

The problem of kinematic calibration is considered in [188, 98]. Methods which do not require the use of external sensors for direct measurement of end-effector position and orientation are proposed in [68].

The kinematic decoupling deriving from the spherical wrist is utilized in [76, 99, 182]. Numerical methods for the solution of the inverse kinematics problem based on iterative algorithms are proposed in [232, 86].

Problems

2.1. Find the rotation matrix corresponding to the set of Euler angles ZXZ.

2.2. Discuss the inverse solution for the Euler angles ZYZ in the case $s_\vartheta = 0$.

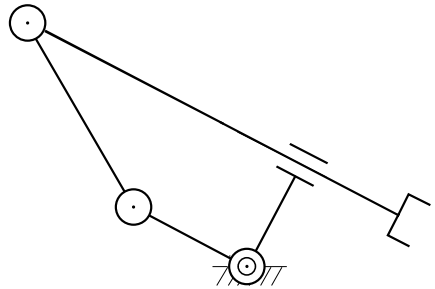

Fig. 2.34. Four-link closed-chain planar arm with prismatic joint

2.3. Discuss the inverse solution for the Roll–Pitch–Yaw angles in the case $c_\vartheta = 0$.

2.4. Verify that the rotation matrix corresponding to the rotation by an angle about an arbitrary axis is given by (2.25).

2.5. Prove that the angle and the unit vector of the axis corresponding to a rotation matrix are given by (2.27), (2.28). Find inverse formulae in the case of $\sin\vartheta = 0$.

2.6. Verify that the rotation matrix corresponding to the unit quaternion is given by (2.33).

2.7. Prove that the unit quaternion is invariant with respect to the rotation matrix and its transpose, i.e., $\boldsymbol{R}(\eta,\boldsymbol{\epsilon})\boldsymbol{\epsilon} = \boldsymbol{R}^T(\eta,\boldsymbol{\epsilon})\boldsymbol{\epsilon} = \boldsymbol{\epsilon}$.

2.8. Prove that the unit quaternion corresponding to a rotation matrix is given by (2.34), (2.35).

2.9. Prove that the quaternion product is expressed by (2.37).

2.10. By applying the rules for inverting a block-partitioned matrix, prove that matrix $\boldsymbol{A}_0^1$ is given by (2.45).

2.11. Find the direct kinematics equation of the four-link closed-chain planar arm in Fig. 2.34, where the two links connected by the prismatic joint are orthogonal to each other

2.12. Find the direct kinematics equation for the cylindrical arm in Fig. 2.35.

2.13. Find the direct kinematics equation for the SCARA manipulator in Fig. 2.36.

2.14. Find the complete direct kinematics equation for the humanoid manipulator in Fig. 2.28.

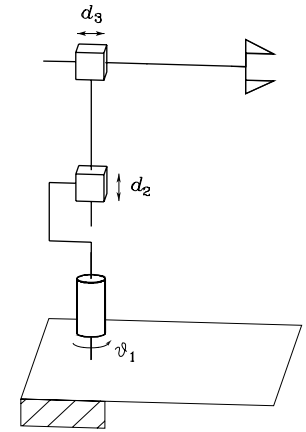

Fig. 2.35. Cylindrical arm

2.15. For the set of minimal representations of orientation $\boldsymbol{\phi}$, define the sum operation in terms of the composition of rotations. By means of an example, show that the commutative property does not hold for that operation.

2.16. Consider the elementary rotations about coordinate axes given by infinitesimal angles. Show that the rotation resulting from any two elementary rotations does not depend on the order of rotations. [*Hint*: for an infinitesimal angle $d\phi$, approximate $\cos(d\phi) \approx 1$ and $\sin(d\phi) \approx d\phi \ldots$]. Further, define $\boldsymbol{R}(d\phi_x, d\phi_y, d\phi_z) = \boldsymbol{R}_x(d\phi_x)\boldsymbol{R}_y(d\phi_y)\boldsymbol{R}_z(d\phi_z)$; show that

$$\boldsymbol{R}(d\phi_x, d\phi_y, d\phi_z)\boldsymbol{R}(d\phi'_x, d\phi'_y, d\phi'_z) = \boldsymbol{R}(d\phi_x + d\phi'_x, d\phi_y + d\phi'_y, d\phi_z + d\phi'_z).$$

2.17. Draw the workspace of the three-link planar arm in Fig. 2.20 with the data:

$$a_1 = 0.5 \qquad a_2 = 0.3 \qquad a_3 = 0.2$$

$$-\pi/3 \le q_1 \le \pi/3 \qquad -2\pi/3 \le q_2 \le 2\pi/3 \qquad -\pi/2 \le q_3 \le \pi/2.$$

2.18. With reference to the inverse kinematics of the anthropomorphic arm in Sect. 2.12.4, discuss the number of solutions in the singular cases of $s_3 = 0$ and $p_{Wx} = p_{Wy} = 0$.

2.19. Solve the inverse kinematics for the cylindrical arm in Fig. 2.35.

2.20. Solve the inverse kinematics for the SCARA manipulator in Fig. 2.36.

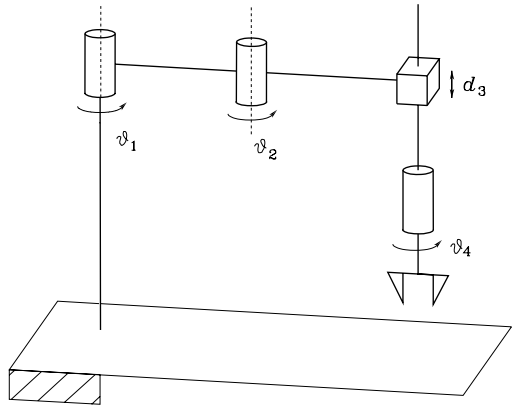

Fig. 2.36. SCARA manipulator

3

Differential Kinematics and Statics

In the previous chapter, direct and inverse kinematics equations establishing the relationship between the joint variables and the end-effector pose were derived. In this chapter, *differential kinematics* is presented which gives the relationship between the joint velocities and the corresponding end-effector linear and angular velocity. This mapping is described by a matrix, termed *geometric Jacobian*, which depends on the manipulator configuration. Alternatively, if the end-effector pose is expressed with reference to a minimal representation in the operational space, it is possible to compute the Jacobian matrix via differentiation of the direct kinematics function with respect to the joint variables. The resulting Jacobian, termed *analytical Jacobian*, in general differs from the geometric one. The Jacobian constitutes one of the most important tools for manipulator characterization; in fact, it is useful for finding *singularities*, analyzing *redundancy*, determining *inverse kinematics algorithms*, describing the mapping between forces applied to the end-effector and resulting torques at the joints (*statics*) and, as will be seen in the following chapters, deriving dynamic equations of motion and designing operational space control schemes. Finally, the *kineto-statics duality* concept is illustrated, which is at the basis of the definition of velocity and force *manipulability ellipsoids*.

3.1 Geometric Jacobian

Consider an n-DOF manipulator. The direct kinematics equation can be written in the form

$$\boldsymbol{T}_e(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{R}_e(\boldsymbol{q}) & \boldsymbol{p}_e(\boldsymbol{q}) \\ \boldsymbol{0}^T & 1 \end{bmatrix} \tag{3.1}$$

where $\boldsymbol{q} = [\, q_1 \quad \dots \quad q_n \,]^T$ is the vector of joint variables. Both end-effector position and orientation vary as $\boldsymbol{q}$ varies.

The goal of the differential kinematics is to find the relationship between the joint velocities and the end-effector linear and angular velocities. In other words, it is desired to express the end-effector linear velocity $\dot{\boldsymbol{p}}_e$ and angular velocity $\boldsymbol{\omega}_e$ as a function of the joint velocities $\dot{\boldsymbol{q}}$. As will be seen afterwards, the sought relations are both linear in the joint velocities, i.e.,

$$\dot{\boldsymbol{p}}_e = \boldsymbol{J}_P(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3.2}$$

$$\boldsymbol{\omega}_e = \boldsymbol{J}_O(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{3.3}$$

In (3.2) $\boldsymbol{J}_P$ is the $(3 \times n)$ matrix relating the contribution of the joint velocities $\dot{\boldsymbol{q}}$ to the end-effector *linear* velocity $\dot{\boldsymbol{p}}_e$, while in (3.3) $\boldsymbol{J}_O$ is the $(3 \times n)$ matrix relating the contribution of the joint velocities $\dot{\boldsymbol{q}}$ to the end-effector *angular* velocity $\boldsymbol{\omega}_e$. In compact form, (3.2), (3.3) can be written as

$$\boldsymbol{v}_e = \begin{bmatrix} \dot{\boldsymbol{p}}_e \\ \boldsymbol{\omega}_e \end{bmatrix} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3.4}$$

which represents the manipulator *differential kinematics equation*. The $(6 \times n)$ matrix $\boldsymbol{J}$ is the manipulator *geometric Jacobian*

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_P \\ \boldsymbol{J}_O \end{bmatrix}, \tag{3.5}$$

which in general is a function of the joint variables.

In order to compute the geometric Jacobian, it is worth recalling a number of properties of rotation matrices and some important results of rigid body kinematics.

3.1.1 Derivative of a Rotation Matrix

The manipulator direct kinematics equation in (3.1) describes the end-effector pose, as a function of the joint variables, in terms of a position vector and a rotation matrix. Since the aim is to characterize the end-effector linear and angular velocities, it is worth considering first the *derivative of a rotation matrix* with respect to time.

Consider a time-varying rotation matrix $\boldsymbol{R} = \boldsymbol{R}(t)$. In view of the orthogonality of $\boldsymbol{R}$, one has the relation

$$\boldsymbol{R}(t)\boldsymbol{R}^T(t) = \boldsymbol{I}$$

which, differentiated with respect to time, gives the identity

$$\dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t) + \boldsymbol{R}(t)\dot{\boldsymbol{R}}^T(t) = \boldsymbol{O}.$$

Set

$$\boldsymbol{S}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t); \tag{3.6}$$

the (3×3) matrix $\boldsymbol{S}$ is *skew-symmetric* since

$$\boldsymbol{S}(t) + \boldsymbol{S}^T(t) = \boldsymbol{O}. \tag{3.7}$$

Postmultiplying both sides of (3.6) by $\boldsymbol{R}(t)$ gives

$$\dot{\boldsymbol{R}}(t) = \boldsymbol{S}(t)\boldsymbol{R}(t) \tag{3.8}$$

that allows the time derivative of $\boldsymbol{R}(t)$ to be expressed as a function of $\boldsymbol{R}(t)$ itself.

Equation (3.8) relates the rotation matrix $\boldsymbol{R}$ to its derivative by means of the skew-symmetric operator $\boldsymbol{S}$ and has a meaningful physical interpretation. Consider a constant vector $\boldsymbol{p}'$ and the vector $\boldsymbol{p}(t) = \boldsymbol{R}(t)\boldsymbol{p}'$. The time derivative of $\boldsymbol{p}(t)$ is

$$\dot{\boldsymbol{p}}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{p}',$$

which, in view of (3.8), can be written as

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{S}(t)\boldsymbol{R}(t)\boldsymbol{p}'.$$

If the vector $\boldsymbol{\omega}(t)$ denotes the *angular velocity* of frame $\boldsymbol{R}(t)$ with respect to the reference frame at time t, it is known from mechanics that

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{\omega}(t) \times \boldsymbol{R}(t)\boldsymbol{p}'.$$

Therefore, the matrix operator $\boldsymbol{S}(t)$ describes the vector product between the vector $\boldsymbol{\omega}$ and the vector $\boldsymbol{R}(t)\boldsymbol{p}'$. The matrix $\boldsymbol{S}(t)$ is so that its symmetric elements with respect to the main diagonal represent the components of the vector $\boldsymbol{\omega}(t) = [\,\omega_x \quad \omega_y \quad \omega_z\,]^T$ in the form

$$\boldsymbol{S} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}, \tag{3.9}$$

which justifies the expression $\boldsymbol{S}(t) = \boldsymbol{S}(\boldsymbol{\omega}(t))$. Hence, (3.8) can be rewritten as

$$\dot{\boldsymbol{R}} = \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}. \tag{3.10}$$

Furthermore, if $\boldsymbol{R}$ denotes a rotation matrix, it can be shown that the following relation holds:

$$\boldsymbol{R}\boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}^T = \boldsymbol{S}(\boldsymbol{R}\boldsymbol{\omega}) \tag{3.11}$$

which will be useful later (see Problem 3.1).

Example 3.1

Consider the elementary rotation matrix about axis z given in (2.6). If α is a function of time, by computing the time derivative of $\boldsymbol{R}_z(\alpha(t))$, (3.6) becomes

$$\boldsymbol{S}(t) = \begin{bmatrix} -\dot{\alpha}\sin\alpha & -\dot{\alpha}\cos\alpha & 0 \\ \dot{\alpha}\cos\alpha & -\dot{\alpha}\sin\alpha & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} 0 & -\dot{\alpha} & 0 \\ \dot{\alpha} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \boldsymbol{S}(\boldsymbol{\omega}(t)).$$

According to (3.9), it is

$$\boldsymbol{\omega} = [\,0 \quad 0 \quad \dot{\alpha}\,]^T$$

that expresses the angular velocity of the frame about axis z.

With reference to Fig. 2.11, consider the coordinate transformation of a point P from Frame 1 to Frame 0; in view of (2.38), this is given by

$$\boldsymbol{p}^0 = \boldsymbol{o}_1^0 + \boldsymbol{R}_1^0 \boldsymbol{p}^1. \tag{3.12}$$

Differentiating (3.12) with respect to time gives

$$\dot{\boldsymbol{p}}^0 = \dot{\boldsymbol{o}}_1^0 + \boldsymbol{R}_1^0 \dot{\boldsymbol{p}}^1 + \dot{\boldsymbol{R}}_1^0 \boldsymbol{p}^1; \tag{3.13}$$

utilizing the expression of the derivative of a rotation matrix (3.8) and specifying the dependence on the angular velocity gives

$$\dot{\boldsymbol{p}}^0 = \dot{\boldsymbol{o}}_1^0 + \boldsymbol{R}_1^0 \dot{\boldsymbol{p}}^1 + \boldsymbol{S}(\boldsymbol{\omega}_1^0)\boldsymbol{R}_1^0 \boldsymbol{p}^1.$$

Further, denoting the vector $\boldsymbol{R}_1^0 \boldsymbol{p}^1$ by $\boldsymbol{r}_1^0$, it is

$$\dot{\boldsymbol{p}}^0 = \dot{\boldsymbol{o}}_1^0 + \boldsymbol{R}_1^0 \dot{\boldsymbol{p}}^1 + \boldsymbol{\omega}_1^0 \times \boldsymbol{r}_1^0 \tag{3.14}$$

which is the known form of the velocity composition rule.

Notice that, if $\boldsymbol{p}^1$ is *fixed* in Frame 1, then it is

$$\dot{\boldsymbol{p}}^0 = \dot{\boldsymbol{o}}_1^0 + \boldsymbol{\omega}_1^0 \times \boldsymbol{r}_1^0 \tag{3.15}$$

since $\dot{\boldsymbol{p}}^1 = \boldsymbol{0}$.

3.1.2 Link Velocities

Consider the generic Link i of a manipulator with an open kinematic chain. According to the Denavit–Hartenberg convention adopted in the previous chapter, Link i connects Joints i and $i+1$; Frame i is attached to Link i

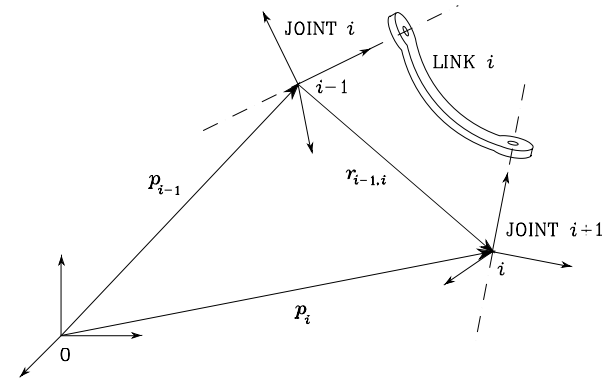

Fig. 3.1. Characterization of generic Link i of a manipulator

and has origin along Joint $i+1$ axis, while Frame $i-1$ has origin along Joint i axis (Fig. 3.1).

Let $\boldsymbol{p}_{i-1}$ and $\boldsymbol{p}_i$ be the position vectors of the origins of Frames $i-1$ and i, respectively. Also, let $\boldsymbol{r}_{i-1,i}^{i-1}$ denote the position of the origin of Frame i with respect to Frame $i-1$ expressed in Frame $i-1$. According to the coordinate transformation (3.10), one can write[1]

$$\boldsymbol{p}_i = \boldsymbol{p}_{i-1} + \boldsymbol{R}_{i-1} \boldsymbol{r}_{i-1,i}^{i-1}.$$

Then, by virtue of (3.14), it is

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{R}_{i-1}\dot{\boldsymbol{r}}_{i-1,i}^{i-1} + \boldsymbol{\omega}_{i-1} \times \boldsymbol{R}_{i-1}\boldsymbol{r}_{i-1,i}^{i-1} = \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{v}_{i-1,i} + \boldsymbol{\omega}_{i-1} \times \boldsymbol{r}_{i-1,i} \tag{3.16}$$

which gives the expression of the linear velocity of Link i as a function of the translational and rotational velocities of Link $i-1$. Note that $\boldsymbol{v}_{i-1,i}$ denotes the velocity of the origin of Frame i with respect to the origin of Frame $i-1$.

Concerning link angular velocity, it is worth starting from the rotation composition

$$\boldsymbol{R}_i = \boldsymbol{R}_{i-1}\boldsymbol{R}_i^{i-1};$$

from (3.8), its time derivative can be written as

$$\boldsymbol{S}(\boldsymbol{\omega}_i)\boldsymbol{R}_i = \boldsymbol{S}(\boldsymbol{\omega}_{i-1})\boldsymbol{R}_i + \boldsymbol{R}_{i-1}\boldsymbol{S}(\boldsymbol{\omega}_{i-1,i}^{i-1})\boldsymbol{R}_i^{i-1} \tag{3.17}$$

where $\boldsymbol{\omega}_{i-1,i}^{i-1}$ denotes the angular velocity of Frame i with respect to Frame $i-1$ expressed in Frame $i-1$. From (2.4), the second term on the right-hand side of (3.17) can be rewritten as

$$\boldsymbol{R}_{i-1}\boldsymbol{S}(\boldsymbol{\omega}_{i-1,i}^{i-1})\boldsymbol{R}_i^{i-1} = \boldsymbol{R}_{i-1}\boldsymbol{S}(\boldsymbol{\omega}_{i-1,i}^{i-1})\boldsymbol{R}_{i-1}^T\boldsymbol{R}_{i-1}\boldsymbol{R}_i^{i-1};$$

[1] Hereafter, the indication of superscript '0' is omitted for quantities referred to Frame 0. Also, without loss of generality, Frame 0 and Frame n are taken as the base frame and the end-effector frame, respectively.

in view of property (3.11), it is

$$\boldsymbol{R}_{i-1}\boldsymbol{S}(\boldsymbol{\omega}_{i-1,i}^{i-1})\boldsymbol{R}_i^{i-1} = \boldsymbol{S}(\boldsymbol{R}_{i-1}\boldsymbol{\omega}_{i-1,i}^{i-1})\boldsymbol{R}_i.$$

Then, (3.17) becomes

$$\boldsymbol{S}(\boldsymbol{\omega}_i)\boldsymbol{R}_i = \boldsymbol{S}(\boldsymbol{\omega}_{i-1})\boldsymbol{R}_i + \boldsymbol{S}(\boldsymbol{R}_{i-1}\boldsymbol{\omega}_{i-1,i}^{i-1})\boldsymbol{R}_i$$

leading to the result

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \boldsymbol{R}_{i-1}\boldsymbol{\omega}_{i-1,i}^{i-1} = \boldsymbol{\omega}_{i-1} + \boldsymbol{\omega}_{i-1,i}, \tag{3.18}$$

which gives the expression of the angular velocity of Link i as a function of the angular velocities of Link $i-1$ and of Link i with respect to Link $i-1$.

The relations (3.16), (3.18) attain different expressions depending on the type of Joint i (*prismatic* or *revolute*).

Prismatic joint

Since orientation of Frame i with respect to Frame $i-1$ does not vary by moving Joint i, it is

$$\boldsymbol{\omega}_{i-1,i} = \boldsymbol{0}. \tag{3.19}$$

Further, the linear velocity is

$$\boldsymbol{v}_{i-1,i} = \dot{d}_i\boldsymbol{z}_{i-1} \tag{3.20}$$

where $\boldsymbol{z}_{i-1}$ is the unit vector of Joint i axis. Hence, the expressions of angular velocity (3.18) and linear velocity (3.16) respectively become

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} \tag{3.21}$$

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \dot{d}_i\boldsymbol{z}_{i-1} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i}, \tag{3.22}$$

where the relation $\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1}$ has been exploited to derive (3.22).

Revolute joint

For the angular velocity it is obviously

$$\boldsymbol{\omega}_{i-1,i} = \dot{\vartheta}_i\boldsymbol{z}_{i-1}, \tag{3.23}$$

while for the linear velocity it is

$$\boldsymbol{v}_{i-1,i} = \boldsymbol{\omega}_{i-1,i} \times \boldsymbol{r}_{i-1,i} \tag{3.24}$$

due to the rotation of Frame i with respect to Frame $i-1$ induced by the motion of Joint i. Hence, the expressions of angular velocity (3.18) and linear velocity (3.16) respectively become

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{\vartheta}_i\boldsymbol{z}_{i-1} \tag{3.25}$$

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i}, \tag{3.26}$$

where (3.18) has been exploited to derive (3.26).

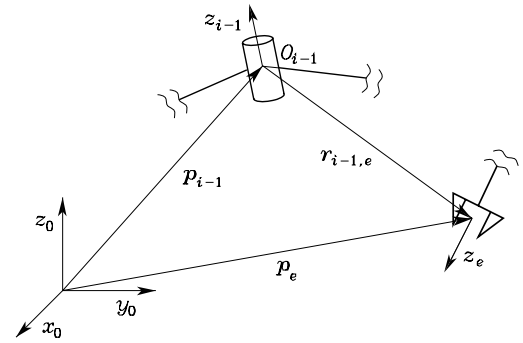

Fig. 3.2. Representation of vectors needed for the computation of the velocity contribution of a revolute joint to the end-effector linear velocity

3.1.3 Jacobian Computation

In order to compute the Jacobian, it is convenient to proceed separately for the linear velocity and the angular velocity.

For the contribution to the *linear velocity*, the time derivative of $\boldsymbol{p}_e(\boldsymbol{q})$ can be written as

$$\dot{\boldsymbol{p}}_e = \sum_{i=1}^{n} \frac{\partial \boldsymbol{p}_e}{\partial q_i}\dot{q}_i = \sum_{i=1}^{n} \boldsymbol{\jmath}_{Pi}\dot{q}_i. \tag{3.27}$$

This expression shows how $\dot{\boldsymbol{p}}_e$ can be obtained as the sum of the terms $\dot{q}_i\boldsymbol{\jmath}_{Pi}$. Each term represents the contribution of the velocity of single Joint i to the end-effector linear velocity when all the other joints are still.

Therefore, by distinguishing the case of a *prismatic* joint ($q_i = d_i$) from the case of a *revolute* joint ($q_i = \vartheta_i$), it is:

- If Joint i is *prismatic*, from (3.20) it is

  $$\dot{q}_i\boldsymbol{\jmath}_{Pi} = \dot{d}_i\boldsymbol{z}_{i-1}$$

  and then

  $$\boldsymbol{\jmath}_{Pi} = \boldsymbol{z}_{i-1}.$$

- If Joint i is *revolute*, observing that the contribution to the linear velocity is to be computed with reference to the origin of the end-effector frame (Fig. 3.2), it is

  $$\dot{q}_i\boldsymbol{\jmath}_{Pi} = \boldsymbol{\omega}_{i-1,i} \times \boldsymbol{r}_{i-1,e} = \dot{\vartheta}_i\boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1})$$

  and then

  $$\boldsymbol{\jmath}_{Pi} = \boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1}).$$

For the contribution to the *angular velocity*, in view of (3.18), it is

$$\boldsymbol{\omega}_e = \boldsymbol{\omega}_n = \sum_{i=1}^{n} \boldsymbol{\omega}_{i-1,i} = \sum_{i=1}^{n} \boldsymbol{\jmath}_{Oi} \dot{q}_i, \tag{3.28}$$

where (3.19) and (3.23) have been utilized to characterize the terms $\dot{q}_i \boldsymbol{\jmath}_{Oi}$, and thus in detail:

- If Joint i is *prismatic*, from (3.19) it is

$$\dot{q}_i \boldsymbol{\jmath}_{Oi} = \mathbf{0}$$

and then

$$\boldsymbol{\jmath}_{Oi} = \mathbf{0}.$$

- If Joint i is *revolute*, from (3.23) it is

$$\dot{q}_i \boldsymbol{\jmath}_{Oi} = \dot{\vartheta}_i \boldsymbol{z}_{i-1}$$

and then

$$\boldsymbol{\jmath}_{Oi} = \boldsymbol{z}_{i-1}.$$

In summary, the Jacobian in (3.5) can be partitioned into the (3×1) column vectors $\boldsymbol{\jmath}_{Pi}$ and $\boldsymbol{\jmath}_{Oi}$ as

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{\jmath}_{P1} & \cdots & \boldsymbol{\jmath}_{Pn} \\ \boldsymbol{\jmath}_{O1} & & \boldsymbol{\jmath}_{On} \end{bmatrix}, \tag{3.29}$$

where

$$\begin{bmatrix} \boldsymbol{\jmath}_{Pi} \\ \boldsymbol{\jmath}_{Oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} \boldsymbol{z}_{i-1} \\ \mathbf{0} \end{bmatrix} & \text{for a } \textit{prismatic} \text{ joint} \\ \begin{bmatrix} \boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1}) \\ \boldsymbol{z}_{i-1} \end{bmatrix} & \text{for a } \textit{revolute} \text{ joint.} \end{cases} \tag{3.30}$$

The expressions in (3.30) allow Jacobian computation in a simple, systematic way on the basis of direct kinematics relations. In fact, the vectors $\boldsymbol{z}_{i-1}$, $\boldsymbol{p}_e$ and $\boldsymbol{p}_{i-1}$ are all functions of the joint variables. In particular:

- $\boldsymbol{z}_{i-1}$ is given by the third column of the rotation matrix $\boldsymbol{R}^0_{i-1}$, i.e.,

$$\boldsymbol{z}_{i-1} = \boldsymbol{R}^0_1(q_1) \dots \boldsymbol{R}^{i-2}_{i-1}(q_{i-1}) \boldsymbol{z}_0 \tag{3.31}$$

where $\boldsymbol{z}_0 = [\,0 \quad 0 \quad 1\,]^T$ allows the selection of the third column.

- $\boldsymbol{p}_e$ is given by the first three elements of the fourth column of the transformation matrix $\boldsymbol{T}^0_e$, i.e., by expressing $\widetilde{\boldsymbol{p}}_e$ in the (4×1) homogeneous form

$$\widetilde{\boldsymbol{p}}_e = \boldsymbol{A}^0_1(q_1) \dots \boldsymbol{A}^{n-1}_n(q_n) \widetilde{\boldsymbol{p}}_0 \tag{3.32}$$

where $\widetilde{\boldsymbol{p}}_0 = [\,0 \quad 0 \quad 0 \quad 1\,]^T$ allows the selection of the fourth column.

- $\boldsymbol{p}_{i-1}$ is given by the first three elements of the fourth column of the transformation matrix $\boldsymbol{T}^0_{i-1}$, i.e., it can be extracted from

$$\widetilde{\boldsymbol{p}}_{i-1} = \boldsymbol{A}^0_1(q_1) \dots \boldsymbol{A}^{i-2}_{i-1}(q_{i-1}) \widetilde{\boldsymbol{p}}_0. \tag{3.33}$$

The above equations can be conveniently used to compute the translational and rotational velocities of any point along the manipulator structure, as long as the direct kinematics functions relative to that point are known.

Finally, notice that the Jacobian matrix depends on the frame in which the end-effector velocity is expressed. The above equations allow computation of the geometric Jacobian with respect to the base frame. If it is desired to represent the Jacobian in a different Frame u, it is sufficient to know the relative rotation matrix $\boldsymbol{R}^u$. The relationship between velocities in the two frames is

$$\begin{bmatrix} \dot{\boldsymbol{p}}^u_e \\ \boldsymbol{\omega}^u_e \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^u & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}^u \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{p}}_e \\ \boldsymbol{\omega}_e \end{bmatrix},$$

which, substituted in (3.4), gives

$$\begin{bmatrix} \dot{\boldsymbol{p}}^u_e \\ \boldsymbol{\omega}^u_e \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^u & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}^u \end{bmatrix} \boldsymbol{J} \dot{\boldsymbol{q}}$$

and then

$$\boldsymbol{J}^u = \begin{bmatrix} \boldsymbol{R}^u & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}^u \end{bmatrix} \boldsymbol{J}, \tag{3.34}$$

where $\boldsymbol{J}^u$ denotes the geometric Jacobian in Frame u, which has been assumed to be time-invariant.

3.2 Jacobian of Typical Manipulator Structures

In the following, the Jacobian is computed for some of the typical manipulator structures presented in the previous chapter.

3.2.1 Three-link Planar Arm

In this case, from (3.30) the Jacobian is

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{z}_0 \times (\boldsymbol{p}_3 - \boldsymbol{p}_0) & \boldsymbol{z}_1 \times (\boldsymbol{p}_3 - \boldsymbol{p}_1) & \boldsymbol{z}_2 \times (\boldsymbol{p}_3 - \boldsymbol{p}_2) \\ \boldsymbol{z}_0 & \boldsymbol{z}_1 & \boldsymbol{z}_2 \end{bmatrix}.$$

Computation of the position vectors of the various links gives

$$\boldsymbol{p}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{p}_1 = \begin{bmatrix} a_1 c_1 \\ a_1 s_1 \\ 0 \end{bmatrix} \quad \boldsymbol{p}_2 = \begin{bmatrix} a_1 c_1 + a_2 c_{12} \\ a_1 s_1 + a_2 s_{12} \\ 0 \end{bmatrix}$$

$$\boldsymbol{p}_3 = \begin{bmatrix} a_1c_1 + a_2c_{12} + a_3c_{123} \\ a_1s_1 + a_2s_{12} + a_3s_{123} \\ 0 \end{bmatrix}$$

while computation of the unit vectors of revolute joint axes gives

$$\boldsymbol{z}_0 = \boldsymbol{z}_1 = \boldsymbol{z}_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

since they are all parallel to axis z_0. From (3.29) it is

$$\boldsymbol{J} = \begin{bmatrix} -a_1s_1 - a_2s_{12} - a_3s_{123} & -a_2s_{12} - a_3s_{123} & -a_3s_{123} \\ a_1c_1 + a_2c_{12} + a_3c_{123} & a_2c_{12} + a_3c_{123} & a_3c_{123} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}. \tag{3.35}$$

In the Jacobian (3.35), only the three non-null rows are relevant (the rank of the matrix is at most 3); these refer to the two components of linear velocity along axes x_0, y_0 and the component of angular velocity about axis z_0. This result can be derived by observing that three DOFs allow specification of at most three end-effector variables; v_z, ω_x, ω_y are always null for this kinematic structure. If orientation is of no concern, the (2×3) Jacobian for the positional part can be derived by considering just the first two rows, i.e.,

$$\boldsymbol{J}_P = \begin{bmatrix} -a_1s_1 - a_2s_{12} - a_3s_{123} & -a_2s_{12} - a_3s_{123} & -a_3s_{123} \\ a_1c_1 + a_2c_{12} + a_3c_{123} & a_2c_{12} + a_3c_{123} & a_3c_{123} \end{bmatrix}. \tag{3.36}$$

3.2.2 Anthropomorphic Arm

In this case, from (3.30) the Jacobian is

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{z}_0 \times (\boldsymbol{p}_3 - \boldsymbol{p}_0) & \boldsymbol{z}_1 \times (\boldsymbol{p}_3 - \boldsymbol{p}_1) & \boldsymbol{z}_2 \times (\boldsymbol{p}_3 - \boldsymbol{p}_2) \\ \boldsymbol{z}_0 & \boldsymbol{z}_1 & \boldsymbol{z}_2 \end{bmatrix}.$$

Computation of the position vectors of the various links gives

$$\boldsymbol{p}_0 = \boldsymbol{p}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{p}_2 = \begin{bmatrix} a_2c_1c_2 \\ a_2s_1c_2 \\ a_2s_2 \end{bmatrix}$$

$$\boldsymbol{p}_3 = \begin{bmatrix} c_1(a_2c_2 + a_3c_{23}) \\ s_1(a_2c_2 + a_3c_{23}) \\ a_2s_2 + a_3s_{23} \end{bmatrix}$$

while computation of the unit vectors of revolute joint axes gives

$$\boldsymbol{z}_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \boldsymbol{z}_1 = \boldsymbol{z}_2 = \begin{bmatrix} s_1 \\ -c_1 \\ 0 \end{bmatrix}.$$

From (3.29) it is

$$\boldsymbol{J} = \begin{bmatrix} -s_1(a_2c_2 + a_3c_{23}) & -c_1(a_2s_2 + a_3s_{23}) & -a_3c_1s_{23} \\ c_1(a_2c_2 + a_3c_{23}) & -s_1(a_2s_2 + a_3s_{23}) & -a_3s_1s_{23} \\ 0 & a_2c_2 + a_3c_{23} & a_3c_{23} \\ 0 & s_1 & s_1 \\ 0 & -c_1 & -c_1 \\ 1 & 0 & 0 \end{bmatrix}. \tag{3.37}$$

Only three of the six rows of the Jacobian (3.37) are linearly independent. Having 3 DOFs only, it is worth considering the upper (3×3) block of the Jacobian

$$\boldsymbol{J}_P = \begin{bmatrix} -s_1(a_2c_2 + a_3c_{23}) & -c_1(a_2s_2 + a_3s_{23}) & -a_3c_1s_{23} \\ c_1(a_2c_2 + a_3c_{23}) & -s_1(a_2s_2 + a_3s_{23}) & -a_3s_1s_{23} \\ 0 & a_2c_2 + a_3c_{23} & a_3c_{23} \end{bmatrix} \tag{3.38}$$

that describes the relationship between the joint velocities and the end-effector linear velocity. This structure does not allow an arbitrary angular velocity $\boldsymbol{\omega}$ to be obtained; in fact, the two components ω_x and ω_y are not independent $(s_1\omega_y = -c_1\omega_x)$.

3.2.3 Stanford Manipulator

In this case, from (3.30) it is

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{z}_0 \times (\boldsymbol{p}_6 - \boldsymbol{p}_0) & \boldsymbol{z}_1 \times (\boldsymbol{p}_6 - \boldsymbol{p}_1) & \boldsymbol{z}_2 & & & \\ \boldsymbol{z}_0 & \boldsymbol{z}_1 & \boldsymbol{0} & & & \\ & & & \boldsymbol{z}_3 \times (\boldsymbol{p}_6 - \boldsymbol{p}_3) & \boldsymbol{z}_4 \times (\boldsymbol{p}_6 - \boldsymbol{p}_4) & \boldsymbol{z}_5 \times (\boldsymbol{p}_6 - \boldsymbol{p}_5) \\ & & & \boldsymbol{z}_3 & \boldsymbol{z}_4 & \boldsymbol{z}_5 \end{bmatrix}.$$

Computation of the position vectors of the various links gives

$$\boldsymbol{p}_0 = \boldsymbol{p}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{p}_3 = \boldsymbol{p}_4 = \boldsymbol{p}_5 = \begin{bmatrix} c_1s_2d_3 - s_1d_2 \\ s_1s_2d_3 + c_1d_2 \\ c_2d_3 \end{bmatrix}$$

$$\boldsymbol{p}_6 = \begin{bmatrix} c_1s_2d_3 - s_1d_2 + \big(c_1(c_2c_4s_5 + s_2c_5) - s_1s_4s_5\big)d_6 \\ s_1s_2d_3 + c_1d_2 + \big(s_1(c_2c_4s_5 + s_2c_5) + c_1s_4s_5\big)d_6 \\ c_2d_3 + (-s_2c_4s_5 + c_2c_5)d_6 \end{bmatrix},$$

while computation of the unit vectors of joint axes gives

$$\boldsymbol{z}_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \boldsymbol{z}_1 = \begin{bmatrix} -s_1 \\ c_1 \\ 0 \end{bmatrix} \quad \boldsymbol{z}_2 = \boldsymbol{z}_3 = \begin{bmatrix} c_1 s_2 \\ s_1 s_2 \\ c_2 \end{bmatrix}$$

$$\boldsymbol{z}_4 = \begin{bmatrix} -c_1 c_2 s_4 - s_1 c_4 \\ -s_1 c_2 s_4 + c_1 c_4 \\ s_2 s_4 \end{bmatrix} \quad \boldsymbol{z}_5 = \begin{bmatrix} c_1(c_2 c_4 s_5 + s_2 c_5) - s_1 s_4 s_5 \\ s_1(c_2 c_4 s_5 + s_2 c_5) + c_1 s_4 s_5 \\ -s_2 c_4 s_5 + c_2 c_5 \end{bmatrix}.$$

The sought Jacobian can be obtained by developing the computations as in (3.29), leading to expressing end-effector linear and angular velocity as a function of joint velocities.

3.3 Kinematic Singularities

The Jacobian in the differential kinematics equation of a manipulator defines a linear mapping

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3.39}$$

between the vector $\dot{\boldsymbol{q}}$ of joint velocities and the vector $\boldsymbol{v}_e = [\,\dot{\boldsymbol{p}}_e^T \quad \boldsymbol{\omega}_e^T\,]^T$ of end-effector velocity. The Jacobian is, in general, a function of the configuration $\boldsymbol{q}$; those configurations at which $\boldsymbol{J}$ is rank-deficient are termed *kinematic singularities*. To find the singularities of a manipulator is of great interest for the following reasons:

a) Singularities represent configurations at which mobility of the structure is reduced, i.e., it is not possible to impose an arbitrary motion to the end-effector.
b) When the structure is at a singularity, infinite solutions to the inverse kinematics problem may exist.
c) In the neighbourhood of a singularity, small velocities in the operational space may cause large velocities in the joint space.

Singularities can be classified into:

- *Boundary* singularities that occur when the manipulator is either outstretched or retracted. It may be understood that these singularities do not represent a true drawback, since they can be avoided on condition that the manipulator is not driven to the boundaries of its reachable workspace.
- *Internal* singularities that occur inside the reachable workspace and are generally caused by the alignment of two or more axes of motion, or else by the attainment of particular end-effector configurations. Unlike the above, these singularities constitute a serious problem, as they can be encountered anywhere in the reachable workspace for a planned path in the operational space.

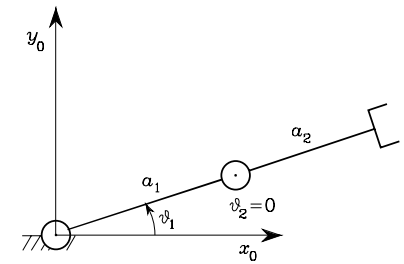

Fig. 3.3. Two-link planar arm at a boundary singularity

Example 3.2

To illustrate the behaviour of a manipulator at a singularity, consider a two-link planar arm. In this case, it is worth considering only the components $\dot{p}_x$ and $\dot{p}_y$ of the linear velocity in the plane. Thus, the Jacobian is the (2×2) matrix

$$\boldsymbol{J} = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} & -a_2 s_{12} \\ a_1 c_1 + a_2 c_{12} & a_2 c_{12} \end{bmatrix}. \tag{3.40}$$

To analyze matrix rank, consider its determinant given by

$$\det(\boldsymbol{J}) = a_1 a_2 s_2. \tag{3.41}$$

For $a_1, a_2 \neq 0$, it is easy to find that the determinant in (3.41) vanishes whenever

$$\vartheta_2 = 0 \qquad \vartheta_2 = \pi,$$

ϑ_1 being irrelevant for the determination of singular configurations. These occur when the arm tip is located either on the outer ($\vartheta_2 = 0$) or on the inner ($\vartheta_2 = \pi$) boundary of the reachable workspace. Figure 3.3 illustrates the arm posture for $\vartheta_2 = 0$.

By analyzing the differential motion of the structure in such configuration, it can be observed that the two column vectors $[\,-(a_1 + a_2)s_1 \quad (a_1 + a_2)c_1\,]^T$ and $[\,-a_2 s_1 \quad a_2 c_1\,]^T$ of the Jacobian become parallel, and thus the Jacobian rank becomes one; this means that the tip velocity components are not independent (see point **a)** above).

3.3.1 Singularity Decoupling

Computation of internal singularities via the Jacobian determinant may be tedious and of no easy solution for complex structures. For manipulators having a spherical wrist, by analogy with what has already been seen for inverse kinematics, it is possible to split the problem of singularity computation into two separate problems:

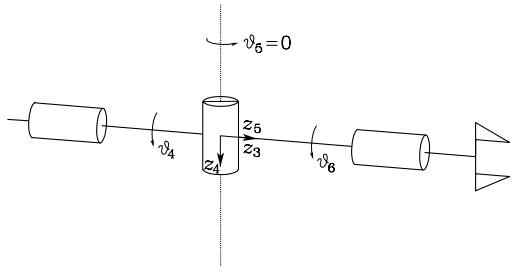

Fig. 3.4. Spherical wrist at a singularity

- computation of *arm singularities* resulting from the motion of the first 3 or more links,
- computation of *wrist singularities* resulting from the motion of the wrist joints.

For the sake of simplicity, consider the case $n = 6$; the Jacobian can be partitioned into (3×3) blocks as follows:

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_{11} & \boldsymbol{J}_{12} \\ \boldsymbol{J}_{21} & \boldsymbol{J}_{22} \end{bmatrix} \tag{3.42}$$

where, since the outer 3 joints are all revolute, the expressions of the two right blocks are respectively

$$\boldsymbol{J}_{12} = \begin{bmatrix} \boldsymbol{z}_3 \times (\boldsymbol{p}_e - \boldsymbol{p}_3) & \boldsymbol{z}_4 \times (\boldsymbol{p}_e - \boldsymbol{p}_4) & \boldsymbol{z}_5 \times (\boldsymbol{p}_e - \boldsymbol{p}_5) \end{bmatrix}$$

$$\boldsymbol{J}_{22} = \begin{bmatrix} \boldsymbol{z}_3 & \boldsymbol{z}_4 & \boldsymbol{z}_5 \end{bmatrix}. \tag{3.43}$$

As singularities are typical of the mechanical structure and do not depend on the frames chosen to describe kinematics, it is convenient to choose the origin of the end-effector frame at the intersection of the wrist axes (see Fig. 2.32). The choice $\boldsymbol{p} = \boldsymbol{p}_W$ leads to

$$\boldsymbol{J}_{12} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix},$$

since all vectors $\boldsymbol{p}_W - \boldsymbol{p}_i$ are parallel to the unit vectors $\boldsymbol{z}_i$, for $i = 3, 4, 5$, no matter how Frames $3, 4, 5$ are chosen according to DH convention. In view of this choice, the overall Jacobian becomes a block lower-triangular matrix. In this case, computation of the determinant is greatly simplified, as this is given by the product of the determinants of the two blocks on the diagonal, i.e.,

$$\det(\boldsymbol{J}) = \det(\boldsymbol{J}_{11})\det(\boldsymbol{J}_{22}). \tag{3.44}$$

In turn, a true *singularity decoupling* has been achieved; the condition

$$\det(\boldsymbol{J}_{11}) = 0$$

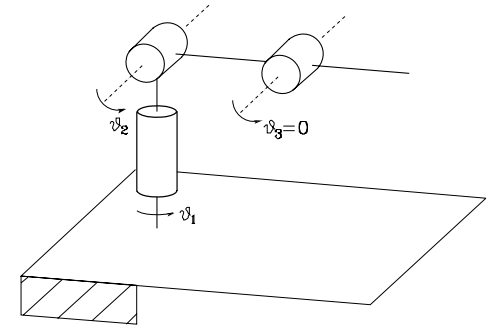

Fig. 3.5. Anthropomorphic arm at an elbow singularity

leads to determining the *arm singularities*, while the condition

$$\det(\boldsymbol{J}_{22}) = 0$$

leads to determining the *wrist singularities*.

Notice, however, that this form of Jacobian does not provide the relationship between the joint velocities and the end-effector velocity, but it leads to simplifying singularity computation. Below the two types of singularities are analyzed in detail.

3.3.2 Wrist Singularities

On the basis of the above singularity decoupling, wrist singularities can be determined by inspecting the block $\boldsymbol{J}_{22}$ in (3.43). It can be recognized that the wrist is at a singular configuration whenever the unit vectors $\boldsymbol{z}_3$, $\boldsymbol{z}_4$, $\boldsymbol{z}_5$ are linearly dependent. The wrist kinematic structure reveals that a singularity occurs when $\boldsymbol{z}_3$ and $\boldsymbol{z}_5$ are aligned, i.e., whenever

$$\vartheta_5 = 0 \qquad \vartheta_5 = \pi.$$

Taking into consideration only the first configuration (Fig. 3.4), the loss of mobility is caused by the fact that rotations of equal magnitude about opposite directions on ϑ_4 and ϑ_6 do not produce any end-effector rotation. Further, the wrist is not allowed to rotate about the axis orthogonal to $\boldsymbol{z}_4$ and $\boldsymbol{z}_3$, (see point **a)** above). This singularity is naturally described in the joint space and can be encountered anywhere inside the manipulator reachable workspace; as a consequence, special care is to be taken in programming an end-effector motion.

3.3.3 Arm Singularities

Arm singularities are characteristic of a specific manipulator structure; to illustrate their determination, consider the anthropomorphic arm (Fig. 2.23),

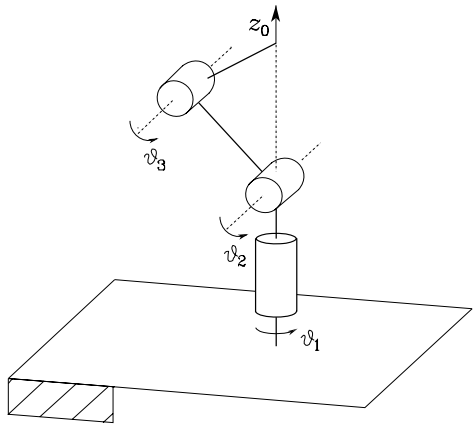

Fig. 3.6. Anthropomorphic arm at a shoulder singularity

whose Jacobian for the linear velocity part is given by (3.38). Its determinant is

$$\det(\boldsymbol{J}_P) = -a_2 a_3 s_3 (a_2 c_2 + a_3 c_{23}).$$

Like in the case of the planar arm of Example 3.2, the determinant does not depend on the first joint variable.

For $a_2, a_3 \neq 0$, the determinant vanishes if $s_3 = 0$ and/or $(a_2 c_2 + a_3 c_{23}) = 0$. The first situation occurs whenever

$$\vartheta_3 = 0 \qquad \vartheta_3 = \pi$$

meaning that the elbow is outstretched (Fig. 3.5) or retracted, and is termed *elbow singularity*. Notice that this type of singularity is conceptually equivalent to the singularity found for the two-link planar arm.

By recalling the direct kinematics equation in (2.66), it can be observed that the second situation occurs when the wrist point lies on axis z_0 (Fig. 3.6); it is thus characterized by

$$p_x = p_y = 0$$

and is termed *shoulder singularity*.

Notice that the whole axis z_0 describes a continuum of singular configurations; a rotation of ϑ_1 does not cause any translation of the wrist position (the first column of $\boldsymbol{J}_P$ is always null at a shoulder singularity), and then the kinematics equation admits infinite solutions; moreover, motions starting from the singular configuration that take the wrist along the z_1 direction are not allowed (see point **b)** above).

If a spherical wrist is connected to an anthropomorphic arm (Fig. 2.26), the arm direct kinematics is different. In this case the Jacobian to consider represents the block $\boldsymbol{J}_{11}$ of the Jacobian in (3.42) with $\boldsymbol{p} = \boldsymbol{p}_W$. Analyzing its

determinant leads to finding the same singular configurations, which are relative to different values of the third joint variables, though — compare (2.66) and (2.70).

Finally, it is important to remark that, unlike the wrist singularities, the arm singularities are well identified in the operational space, and thus they can be suitably avoided in the end-effector trajectory planning stage.

3.4 Analysis of Redundancy

The concept of *kinematic redundancy* has been introduced in Sect. 2.10.2; redundancy is related to the number n of DOFs of the structure, the number m of operational space variables, and the number r of operational space variables necessary to specify a given task.

In order to perform a systematic analysis of redundancy, it is worth considering differential kinematics in lieu of direct kinematics (2.82). To this end, (3.39) is to be interpreted as the differential kinematics mapping relating the n components of the joint velocity vector to the $r \leq m$ components of the velocity vector $\boldsymbol{v}_e$ of concern for the specific task. To clarify this point, consider the case of a 3-link planar arm; that is not intrinsically redundant ($n = m = 3$) and its Jacobian (3.35) has 3 null rows accordingly. If the task does not specify ω_z ($r = 2$), the arm becomes functionally redundant and the Jacobian to consider for redundancy analysis is the one in (3.36).

A different case is that of the anthropomorphic arm for which only position variables are of concern ($n = m = 3$). The relevant Jacobian is the one in (3.38). The arm is neither intrinsically redundant nor can become functionally redundant if it is assigned a planar task; in that case, indeed, the task would set constraints on the 3 components of end-effector linear velocity.

Therefore, the differential kinematics equation to consider can be formally written as in (3.39), i.e.,

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{3.45}$$

where now $\boldsymbol{v}_e$ is meant to be the $(r \times 1)$ vector of end-effector velocity of concern for the specific task and $\boldsymbol{J}$ is the corresponding $(r \times n)$ Jacobian matrix that can be extracted from the geometric Jacobian; $\dot{\boldsymbol{q}}$ is the $(n \times 1)$ vector of joint velocities. If $r < n$, the manipulator is kinematically redundant and there exist $(n - r)$ *redundant DOFs*.

The Jacobian describes the linear mapping from the joint velocity space to the end-effector velocity space. In general, it is a function of the configuration. In the context of differential kinematics, however, the Jacobian has to be regarded as a constant matrix, since the instantaneous velocity mapping is of interest for a given posture. The mapping is schematically illustrated in Fig. 3.7 with a typical notation from set theory.

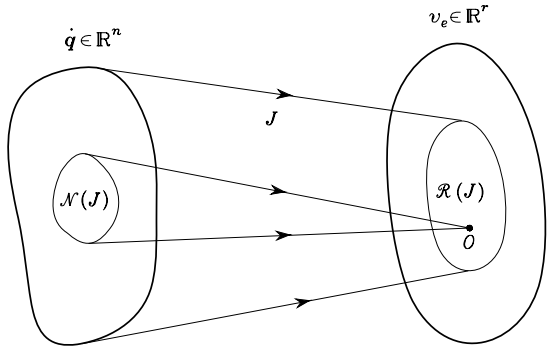

Fig. 3.7. Mapping between the joint velocity space and the end-effector velocity space

The differential kinematics equation in (3.45) can be characterized in terms of the *range* and *null* spaces of the mapping;[2] specifically, one has that:

- The *range* space of $\boldsymbol{J}$ is the subspace $\mathcal{R}(\boldsymbol{J})$ in $\mathbb{R}^r$ of the end-effector velocities that can be generated by the joint velocities, in the given manipulator posture.
- The *null* space of $\boldsymbol{J}$ is the subspace $\mathcal{N}(\boldsymbol{J})$ in $\mathbb{R}^n$ of joint velocities that do not produce any end-effector velocity, in the given manipulator posture.

If the Jacobian has *full rank*, one has

$$\dim\big(\mathcal{R}(\boldsymbol{J})\big) = r \qquad \dim\big(\mathcal{N}(\boldsymbol{J})\big) = n - r$$

and the range of $\boldsymbol{J}$ spans the entire space $\mathbb{R}^r$. Instead, if the Jacobian degenerates at a *singularity*, the dimension of the range space decreases while the dimension of the null space increases, since the following relation holds:

$$\dim\big(\mathcal{R}(\boldsymbol{J})\big) + \dim\big(\mathcal{N}(\boldsymbol{J})\big) = n$$

independently of the rank of the matrix $\boldsymbol{J}$.

The existence of a subspace $\mathcal{N}(\boldsymbol{J}) \neq \emptyset$ for a redundant manipulator allows determination of systematic techniques for handling redundant DOFs. To this end, if $\dot{\boldsymbol{q}}^*$ denotes a solution to (3.45) and $\boldsymbol{P}$ is an $(n \times n)$ matrix so that

$$\mathcal{R}(\boldsymbol{P}) \equiv \mathcal{N}(\boldsymbol{J}),$$

the joint velocity vector

$$\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}^* + \boldsymbol{P}\dot{\boldsymbol{q}}_0, \tag{3.46}$$

with arbitrary $\dot{\boldsymbol{q}}_0$, is also a solution to (3.45). In fact, premultiplying both sides of (3.46) by $\boldsymbol{J}$ yields

$$\boldsymbol{J}\dot{\boldsymbol{q}} = \boldsymbol{J}\dot{\boldsymbol{q}}^* + \boldsymbol{J}\boldsymbol{P}\dot{\boldsymbol{q}}_0 = \boldsymbol{J}\dot{\boldsymbol{q}}^* = \boldsymbol{v}_e$$

[2] See Sect. A.4 for the linear mappings.

since $\boldsymbol{J}\boldsymbol{P}\dot{\boldsymbol{q}}_0 = \boldsymbol{0}$ for any $\dot{\boldsymbol{q}}_0$. This result is of fundamental importance for redundancy resolution; a solution of the kind (3.46) points out the possibility of choosing the vector of arbitrary joint velocities $\dot{\boldsymbol{q}}_0$ so as to exploit advantageously the redundant DOFs. In fact, the effect of $\dot{\boldsymbol{q}}_0$ is to generate *internal motions* of the structure that do not change the end-effector position and orientation but may allow, for instance, manipulator reconfiguration into more dexterous postures for execution of a given task.

3.5 Inverse Differential Kinematics

In Sect. 2.12 it was shown how the inverse kinematics problem admits closed-form solutions only for manipulators having a simple kinematic structure. Problems arise whenever the end-effector attains a particular position and/or orientation in the operational space, or the structure is complex and it is not possible to relate the end-effector pose to different sets of joint variables, or else the manipulator is redundant. These limitations are caused by the highly nonlinear relationship between joint space variables and operational space variables.

On the other hand, the differential kinematics equation represents a linear mapping between the joint velocity space and the operational velocity space, although it varies with the current configuration. This fact suggests the possibility to utilize the differential kinematics equation to tackle the inverse kinematics problem.

Suppose that a motion trajectory is assigned to the end-effector in terms of $\boldsymbol{v}_e$ and the initial conditions on position and orientation. The aim is to determine a feasible joint trajectory $(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))$ that reproduces the given trajectory.

By considering (3.45) with $n = r$, the joint velocities can be obtained via simple inversion of the Jacobian matrix

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q})\boldsymbol{v}_e. \tag{3.47}$$

If the initial manipulator posture $\boldsymbol{q}(0)$ is known, joint positions can be computed by integrating velocities over time, i.e.,

$$\boldsymbol{q}(t) = \int_0^t \dot{\boldsymbol{q}}(\varsigma)d\varsigma + \boldsymbol{q}(0).$$

The integration can be performed in discrete time by resorting to numerical techniques. The simplest technique is based on the Euler integration method; given an integration interval Δt, if the joint positions and velocities at time t_k are known, the joint positions at time $t_{k+1} = t_k + \Delta t$ can be computed as

$$\boldsymbol{q}(t_{k+1}) = \boldsymbol{q}(t_k) + \dot{\boldsymbol{q}}(t_k)\Delta t. \tag{3.48}$$

This technique for inverting kinematics is independent of the solvability of the kinematic structure. Nonetheless, it is necessary that the *Jacobian* be *square* and of *full rank*; this demands further insight into the cases of *redundant* manipulators and kinematic *singularity* occurrence.

3.5.1 Redundant Manipulators

When the manipulator is *redundant* ($r < n$), the Jacobian matrix has more columns than rows and infinite solutions exist to (3.45). A viable solution method is to formulate the problem as a constrained linear optimization problem.

In detail, once the end-effector velocity $\boldsymbol{v}_e$ and Jacobian $\boldsymbol{J}$ are given (for a given configuration $\boldsymbol{q}$), it is desired to find the solutions $\dot{\boldsymbol{q}}$ that satisfy the linear equation in (3.45) and *minimize* the quadratic cost functional of joint velocities[3]

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

where $\boldsymbol{W}$ is a suitable $(n \times n)$ symmetric positive definite weighting matrix.

This problem can be solved with the *method of Lagrange multipliers*. Consider the modified cost functional

$$g(\dot{\boldsymbol{q}}, \boldsymbol{\lambda}) = \frac{1}{2}\dot{\boldsymbol{q}}^T \boldsymbol{W} \dot{\boldsymbol{q}} + \boldsymbol{\lambda}^T (\boldsymbol{v}_e - \boldsymbol{J}\dot{\boldsymbol{q}}),$$

where $\boldsymbol{\lambda}$ is an $(r \times 1)$ vector of unknown multipliers that allows the incorporation of the constraint (3.45) in the functional to minimize. The requested solution has to satisfy the necessary conditions:

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

From the first one, it is $\boldsymbol{W}\dot{\boldsymbol{q}} - \boldsymbol{J}^T\boldsymbol{\lambda} = \boldsymbol{0}$ and thus

$$\dot{\boldsymbol{q}} = \boldsymbol{W}^{-1}\boldsymbol{J}^T\boldsymbol{\lambda} \tag{3.49}$$

where the inverse of $\boldsymbol{W}$ exists. Notice that the solution (3.49) is a minimum, since $\partial^2 g / \partial \dot{\boldsymbol{q}}^2 = \boldsymbol{W}$ is positive definite. From the second condition above, the constraint

$$\boldsymbol{v}_e = \boldsymbol{J}\dot{\boldsymbol{q}}$$

is recovered. Combining the two conditions gives

$$\boldsymbol{v}_e = \boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^T\boldsymbol{\lambda};$$

under the assumption that $\boldsymbol{J}$ has full rank, $\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^T$ is an $(r \times r)$ square matrix of rank r and thus can be inverted. Solving for $\boldsymbol{\lambda}$ yields

$$\boldsymbol{\lambda} = (\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^T)^{-1}\boldsymbol{v}_e$$

[3] Quadratic forms and the relative operations are recalled in Sect. A.6.

which, substituted into (3.49), gives the sought optimal solution

$$\dot{\boldsymbol{q}} = \boldsymbol{W}^{-1}\boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{W}^{-1}\boldsymbol{J}^T)^{-1}\boldsymbol{v}_e. \tag{3.50}$$

Premultiplying both sides of (3.50) by $\boldsymbol{J}$, it is easy to verify that this solution satisfies the differential kinematics equation in (3.45).

A particular case occurs when the weighting matrix $\boldsymbol{W}$ is the identity matrix $\boldsymbol{I}$ and the solution simplifies into

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^\dagger \boldsymbol{v}_e; \tag{3.51}$$

the matrix

$$\boldsymbol{J}^\dagger = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T)^{-1} \tag{3.52}$$

is the *right pseudo-inverse* of $\boldsymbol{J}$.[4] The obtained solution locally minimizes the norm of joint velocities.

It was pointed out above that if $\dot{\boldsymbol{q}}^*$ is a solution to (3.45), $\dot{\boldsymbol{q}}^* + \boldsymbol{P}\dot{\boldsymbol{q}}_0$ is also a solution, where $\dot{\boldsymbol{q}}_0$ is a vector of arbitrary joint velocities and $\boldsymbol{P}$ is a projector in the null space of $\boldsymbol{J}$. Therefore, in view of the presence of redundant DOFs, the solution (3.51) can be modified by the introduction of another term of the kind $\boldsymbol{P}\dot{\boldsymbol{q}}_0$. In particular, $\dot{\boldsymbol{q}}_0$ can be specified so as to satisfy an additional constraint to the problem.

In that case, it is necessary to consider a new cost functional in the form

$$g'(\dot{\boldsymbol{q}}) = \frac{1}{2}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_0)^T(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_0);$$

this choice is aimed at minimizing the norm of vector $\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_0$; in other words, solutions are sought which satisfy the constraint (3.45) and are as close as possible to $\dot{\boldsymbol{q}}_0$. In this way, the objective specified through $\dot{\boldsymbol{q}}_0$ becomes unavoidably a secondary objective to satisfy with respect to the primary objective specified by the constraint (3.45).

Proceeding in a way similar to the above yields

$$g'(\dot{\boldsymbol{q}}, \boldsymbol{\lambda}) = \frac{1}{2}(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_0)^T(\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_0) + \boldsymbol{\lambda}^T(\boldsymbol{v}_e - \boldsymbol{J}\dot{\boldsymbol{q}});$$

from the first necessary condition it is

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^T\boldsymbol{\lambda} + \dot{\boldsymbol{q}}_0 \tag{3.53}$$

which, substituted into (3.45), gives

$$\boldsymbol{\lambda} = (\boldsymbol{J}\boldsymbol{J}^T)^{-1}(\boldsymbol{v}_e - \boldsymbol{J}\dot{\boldsymbol{q}}_0).$$

Finally, substituting $\boldsymbol{\lambda}$ back in (3.53) gives

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^\dagger \boldsymbol{v}_e + (\boldsymbol{I}_n - \boldsymbol{J}^\dagger \boldsymbol{J})\dot{\boldsymbol{q}}_0. \tag{3.54}$$

[4] See Sect. A.7 for the definition of the pseudo-inverse of a matrix.

As can be easily recognized, the obtained solution is composed of two terms. The first is relative to minimum norm joint velocities. The second, termed *homogeneous solution*, attempts to satisfy the additional constraint to specify via $\dot{\boldsymbol{q}}_0$;[5] the matrix $(\boldsymbol{I}-\boldsymbol{J}^\dagger\boldsymbol{J})$ is one of those matrices $\boldsymbol{P}$ introduced in (3.46) which allows the projection of the vector $\dot{\boldsymbol{q}}_0$ in the null space of $\boldsymbol{J}$, so as not to violate the constraint (3.45). A direct consequence is that, in the case $\boldsymbol{v}_e=\boldsymbol{0}$, is is possible to generate *internal motions* described by $(\boldsymbol{I}-\boldsymbol{J}^\dagger\boldsymbol{J})\dot{\boldsymbol{q}}_0$ that reconfigure the manipulator structure without changing the end-effector position and orientation.

Finally, it is worth discussing the way to specify the vector $\dot{\boldsymbol{q}}_0$ for a convenient utilization of redundant DOFs. A typical choice is

$$\dot{\boldsymbol{q}}_0 = k_0 \left(\frac{\partial w(\boldsymbol{q})}{\partial \boldsymbol{q}}\right)^T \tag{3.55}$$

where $k_0>0$ and $w(\boldsymbol{q})$ is a (secondary) objective function of the joint variables. Since the solution moves along the direction of the gradient of the objective function, it attempts to *maximize* it *locally* compatible to the primary objective (kinematic constraint). Typical objective functions are:

- The *manipulability measure*, defined as

$$w(\boldsymbol{q}) = \sqrt{\det\big(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})\big)} \tag{3.56}$$

  which vanishes at a singular configuration; thus, by maximizing this measure, redundancy is exploited to move away from singularities.[6]
- The *distance from mechanical joint limits*, defined as

$$w(\boldsymbol{q}) = -\frac{1}{2n}\sum_{i=1}^{n}\left(\frac{q_i-\bar{q}_i}{q_{iM}-q_{im}}\right)^2 \tag{3.57}$$

  where q_{iM} (q_{im}) denotes the maximum (minimum) joint limit and $\bar{q}_i$ the middle value of the joint range; thus, by maximizing this distance, redundancy is exploited to keep the joint variables as close as possible to the centre of their ranges.
- The *distance from an obstacle*, defined as

$$w(\boldsymbol{q}) = \min_{\boldsymbol{p},\boldsymbol{o}} \|\boldsymbol{p}(\boldsymbol{q})-\boldsymbol{o}\| \tag{3.58}$$

  where $\boldsymbol{o}$ is the position vector of a suitable point on the obstacle (its centre, for instance, if the obstacle is modelled as a sphere) and $\boldsymbol{p}$ is the position vector of a generic point along the structure; thus, by maximizing this distance, redundancy is exploited to avoid collision of the manipulator with an obstacle (see also Problem 3.9).[7]

[5] It should be recalled that the additional constraint has secondary priority with respect to the primary kinematic constraint.

[6] The manipulability measure is given by the product of the singular values of the Jacobian (see Problem 3.8).

3.5.2 Kinematic Singularities

Both solutions (3.47) and (3.51) can be computed only when the Jacobian has full rank. Hence, they become meaningless when the manipulator is at a singular configuration; in such a case, the system $\boldsymbol{v}_e=\boldsymbol{J}\dot{\boldsymbol{q}}$ contains linearly dependent equations.

It is possible to find a solution $\dot{\boldsymbol{q}}$ by extracting all the linearly independent equations only if $\boldsymbol{v}_e\in\mathcal{R}(\boldsymbol{J})$. The occurrence of this situation means that the assigned path is physically executable by the manipulator, even though it is at a singular configuration. If instead $\boldsymbol{v}_e\notin\mathcal{R}(\boldsymbol{J})$, the system of equations has no solution; this means that the operational space path cannot be executed by the manipulator at the given posture.

It is important to underline that the inversion of the Jacobian can represent a serious inconvenience not only at a singularity but also in the neighbourhood of a singularity. For instance, for the Jacobian inverse it is well known that its computation requires the computation of the determinant; in the neighbourhood of a singularity, the determinant takes on a relatively small value which can cause large joint velocities (see point **c)** in Sect. 3.3). Consider again the above example of the shoulder singularity for the anthropomorphic arm. If a path is assigned to the end-effector which passes nearby the base rotation axis (geometric locus of singular configurations), the base joint is forced to make a rotation of about π in a relatively short time to allow the end-effector to keep tracking the imposed trajectory.

A more rigorous analysis of the solution features in the neighbourhood of singular configurations can be developed by resorting to the singular value decomposition (SVD) of matrix $\boldsymbol{J}$.[8]

An alternative solution overcoming the problem of inverting differential kinematics in the neighbourhood of a singularity is provided by the so-called *damped least-squares (DLS) inverse*

$$\boldsymbol{J}^\star = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T+k^2\boldsymbol{I})^{-1} \tag{3.59}$$

where k is a damping factor that renders the inversion better conditioned from a numerical viewpoint. It can be shown that such a solution can be

[7] If an obstacle occurs along the end-effector path, it is opportune to invert the order of priority between the kinematic constraint and the additional constraint; in this way the obstacle may be avoided, but one gives up tracking the desired path.

[8] See Sect. A.8.

obtained by reformulating the problem in terms of the minimization of the cost functional

$$g''(\dot{\boldsymbol{q}}) = \frac{1}{2}(\boldsymbol{v}_e - \boldsymbol{J}\dot{\boldsymbol{q}})^T(\boldsymbol{v}_e - \boldsymbol{J}\dot{\boldsymbol{q}}) + \frac{1}{2}k^2\dot{\boldsymbol{q}}^T\dot{\boldsymbol{q}},$$

where the introduction of the first term allows a finite inversion error to be tolerated, with the advantage of norm-bounded velocities. The factor k establishes the relative weight between the two objectives, and there exist techniques for selecting optimal values for the damping factor (see Problem 3.10).

3.6 Analytical Jacobian

The above sections have shown the way to compute the end-effector velocity in terms of the velocity of the end-effector frame. The Jacobian is computed according to a *geometric technique* in which the contributions of each joint velocity to the components of end-effector linear and angular velocity are determined.

If the end-effector pose is specified in terms of a minimal number of parameters in the operational space as in (2.80), it is natural to ask whether it is possible to compute the Jacobian via differentiation of the direct kinematics function with respect to the joint variables. To this end, an *analytical technique* is presented below to compute the Jacobian, and the existing relationship between the two Jacobians is found.

The translational velocity of the end-effector frame can be expressed as the time derivative of vector $\boldsymbol{p}_e$, representing the origin of the end-effector frame with respect to the base frame, i.e.,

$$\dot{\boldsymbol{p}}_e = \frac{\partial \boldsymbol{p}_e}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} = \boldsymbol{J}_P(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{3.60}$$

For what concerns the rotational velocity of the end-effector frame, the minimal representation of orientation in terms of three variables $\boldsymbol{\phi}_e$ can be considered. Its time derivative $\dot{\boldsymbol{\phi}}_e$ in general differs from the angular velocity vector defined above. In any case, once the function $\boldsymbol{\phi}_e(\boldsymbol{q})$ is known, it is formally correct to consider the Jacobian obtained as

$$\dot{\boldsymbol{\phi}}_e = \frac{\partial \boldsymbol{\phi}_e}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} = \boldsymbol{J}_\phi(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{3.61}$$

Computing the Jacobian $\boldsymbol{J}_\phi(\boldsymbol{q})$ as $\partial\boldsymbol{\phi}_e/\partial\boldsymbol{q}$ is not straightforward, since the function $\boldsymbol{\phi}_e(\boldsymbol{q})$ is not usually available in direct form, but requires computation of the elements of the relative rotation matrix.

Upon these premises, the differential kinematics equation can be obtained as the time derivative of the direct kinematics equation in (2.82), i.e.,

$$\dot{\boldsymbol{x}}_e = \begin{bmatrix} \dot{\boldsymbol{p}}_e \\ \dot{\boldsymbol{\phi}}_e \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_P(\boldsymbol{q}) \\ \boldsymbol{J}_\phi(\boldsymbol{q}) \end{bmatrix} \dot{\boldsymbol{q}} = \boldsymbol{J}_A(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3.62}$$

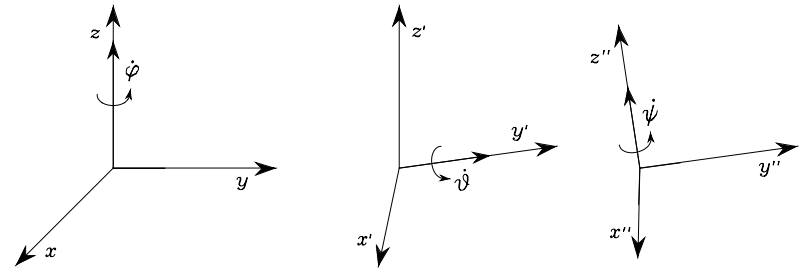

Fig. 3.8. Rotational velocities of Euler angles ZYZ in current frame

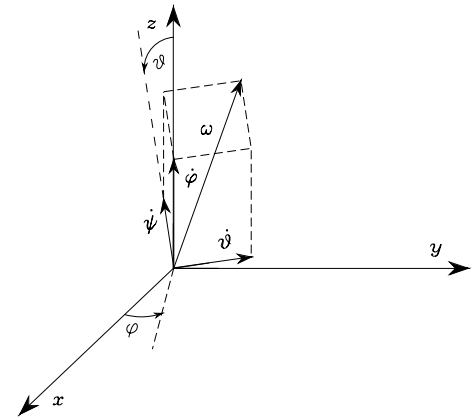

Fig. 3.9. Composition of elementary rotational velocities for computing angular velocity

where the *analytical Jacobian*

$$\boldsymbol{J}_A(\boldsymbol{q}) = \frac{\partial \boldsymbol{k}(\boldsymbol{q})}{\partial \boldsymbol{q}} \tag{3.63}$$

is different from the geometric Jacobian $\boldsymbol{J}$, since the end-effector angular velocity $\boldsymbol{\omega}_e$ with respect to the base frame is not given by $\dot{\boldsymbol{\phi}}_e$.

It is possible to find the relationship between the angular velocity $\boldsymbol{\omega}_e$ and the rotational velocity $\dot{\boldsymbol{\phi}}_e$ for a given set of orientation angles. For instance, consider the Euler angles ZYZ defined in Sect. 2.4.1; in Fig. 3.8, the vectors corresponding to the rotational velocities $\dot{\varphi}$, $\dot{\vartheta}$, $\dot{\psi}$ have been represented with reference to the current frame. Figure 3.9 illustrates how to compute the contributions of each rotational velocity to the components of angular velocity about the axes of the reference frame:

- as a result of $\dot{\varphi}$: $[\,\omega_x \quad \omega_y \quad \omega_z\,]^T = \dot{\varphi}\,[\,0 \quad 0 \quad 1\,]^T$
- as a result of $\dot{\vartheta}$: $[\,\omega_x \quad \omega_y \quad \omega_z\,]^T = \dot{\vartheta}\,[\,-s_\varphi \quad c_\varphi \quad 0\,]^T$

- as a result of $\dot{\psi}$: $[\omega_x \quad \omega_y \quad \omega_z]^T = \dot{\psi}\,[c_\varphi s_\vartheta \quad s_\varphi s_\vartheta \quad c_\vartheta]^T$,

and then the equation relating the angular velocity $\boldsymbol{\omega}_e$ to the time derivative of the Euler angles $\dot{\boldsymbol{\phi}}_e$ is[9]

$$\boldsymbol{\omega}_e = \boldsymbol{T}(\boldsymbol{\phi}_e)\dot{\boldsymbol{\phi}}_e, \tag{3.64}$$

where, in this case,

$$\boldsymbol{T} = \begin{bmatrix} 0 & -s_\varphi & c_\varphi s_\vartheta \\ 0 & c_\varphi & s_\varphi s_\vartheta \\ 1 & 0 & c_\vartheta \end{bmatrix}.$$

The determinant of matrix $\boldsymbol{T}$ is $-s_\vartheta$, which implies that the relationship cannot be inverted for $\vartheta = 0, \pi$. This means that, even though all rotational velocities of the end-effector frame can be expressed by means of a suitable angular velocity vector $\boldsymbol{\omega}_e$, there exist angular velocities which cannot be expressed by means of $\dot{\boldsymbol{\phi}}_e$ when the orientation of the end-effector frame causes $s_\vartheta = 0$.[10] In fact, in this situation, the angular velocities that can be described by $\dot{\boldsymbol{\phi}}_e$ should have linearly dependent components in the directions orthogonal to axis z ($\omega_x^2 + \omega_y^2 = \dot{\vartheta}^2$). An orientation for which the determinant of the transformation matrix vanishes is termed *representation singularity* of $\boldsymbol{\phi}_e$.

From a physical viewpoint, the meaning of $\boldsymbol{\omega}_e$ is more intuitive than that of $\dot{\boldsymbol{\phi}}_e$. The three components of $\boldsymbol{\omega}_e$ represent the components of angular velocity with respect to the base frame. Instead, the three elements of $\dot{\boldsymbol{\phi}}_e$ represent nonorthogonal components of angular velocity defined with respect to the axes of a frame that varies as the end-effector orientation varies. On the other hand, while the integral of $\dot{\boldsymbol{\phi}}_e$ over time gives $\boldsymbol{\phi}_e$, the integral of $\boldsymbol{\omega}_e$ does not admit a clear physical interpretation, as can be seen in the following example.

Example 3.3

Consider an object whose orientation with respect to a reference frame is known at time $t = 0$. Assign the following time profiles to $\boldsymbol{\omega}$:

- $\boldsymbol{\omega} = [\pi/2 \quad 0 \quad 0]^T \quad 0 \le t \le 1 \qquad \boldsymbol{\omega} = [0 \quad \pi/2 \quad 0]^T \quad 1 < t \le 2,$
- $\boldsymbol{\omega} = [0 \quad \pi/2 \quad 0]^T \quad 0 \le t \le 1 \qquad \boldsymbol{\omega} = [\pi/2 \quad 0 \quad 0]^T \quad 1 < t \le 2.$

The integral of $\boldsymbol{\omega}$ gives the same result in the two cases

$$\int_0^2 \boldsymbol{\omega} dt = [\pi/2 \quad \pi/2 \quad 0]^T$$

but the final object orientation corresponding to the second timing law is clearly different from the one obtained with the first timing law (Fig. 3.10).

---

[9] This relation can also be obtained from the rotation matrix associated with the three angles (see Problem 3.11).

[10] In Sect. 2.4.1, it was shown that for this orientation the inverse solution of the Euler angles degenerates.

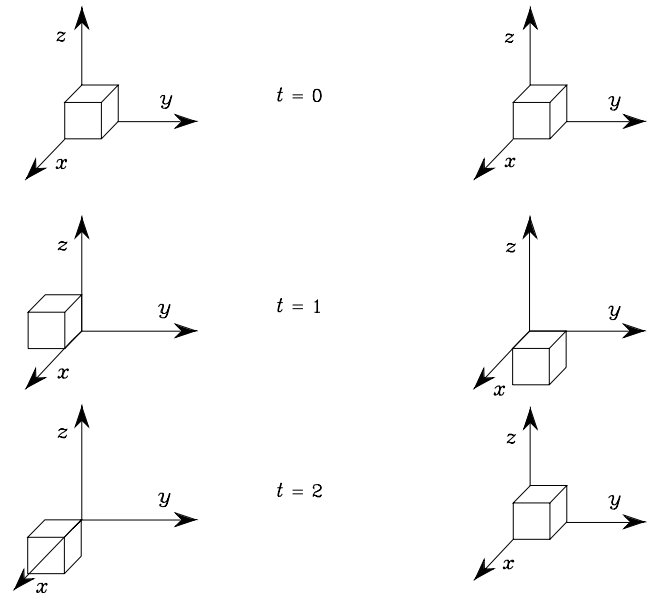

Fig. 3.10. Nonuniqueness of orientation computed as the integral of angular velocity

Once the transformation $\boldsymbol{T}$ between $\boldsymbol{\omega}_e$ and $\dot{\boldsymbol{\phi}}_e$ is given, the analytical Jacobian can be related to the geometric Jacobian as

$$\boldsymbol{v}_e = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{T}(\boldsymbol{\phi}_e) \end{bmatrix} \dot{\boldsymbol{x}}_e = \boldsymbol{T}_A(\boldsymbol{\phi}_e)\dot{\boldsymbol{x}}_e \tag{3.65}$$

which, in view of (3.4), (3.62), yields

$$\boldsymbol{J} = \boldsymbol{T}_A(\boldsymbol{\phi})\boldsymbol{J}_A. \tag{3.66}$$

This relationship shows that $\boldsymbol{J}$ and $\boldsymbol{J}_A$, in general, differ. Regarding the use of either one or the other in all those problems where the influence of the Jacobian matters, it is anticipated that the geometric Jacobian will be adopted whenever it is necessary to refer to quantities of clear physical meaning, while the analytical Jacobian will be adopted whenever it is necessary to refer to differential quantities of variables defined in the operational space.

For certain manipulator geometries, it is possible to establish a substantial equivalence between $\boldsymbol{J}$ and $\boldsymbol{J}_A$. In fact, when the DOFs cause rotations of the end-effector all about the same fixed axis in space, the two Jacobians are essentially the same. This is the case of the above three-link planar arm. Its geometric Jacobian (3.35) reveals that only rotations about axis z_0 are permitted. The (3×3) analytical Jacobian that can be derived by considering the end-effector position components in the plane of the structure and defining

the end-effector orientation as $\phi = \vartheta_1 + \vartheta_2 + \vartheta_3$ coincides with the matrix that is obtained by eliminating the three null rows of the geometric Jacobian.

3.7 Inverse Kinematics Algorithms

In Sect. 3.5 it was shown how to invert kinematics by using the differential kinematics equation. In the numerical implementation of (3.48), computation of joint velocities is obtained by using the inverse of the Jacobian evaluated with the joint variables at the previous instant of time

$$\boldsymbol{q}(t_{k+1}) = \boldsymbol{q}(t_k) + \boldsymbol{J}^{-1}(\boldsymbol{q}(t_k))\boldsymbol{v}_e(t_k)\Delta t.$$

It follows that the computed joint velocities $\dot{\boldsymbol{q}}$ do not coincide with those satisfying (3.47) in the continuous time. Therefore, reconstruction of joint variables $\boldsymbol{q}$ is entrusted to a numerical integration which involves *drift* phenomena of the solution; as a consequence, the end-effector pose corresponding to the computed joint variables differs from the desired one.

This inconvenience can be overcome by resorting to a solution scheme that accounts for the *operational space error* between the desired and the actual end-effector position and orientation. Let

$$\boldsymbol{e} = \boldsymbol{x}_d - \boldsymbol{x}_e \tag{3.67}$$

be the expression of such error.

Consider the time derivative of (3.67), i.e.,

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}_e \tag{3.68}$$

which, according to differential kinematics (3.62), can be written as

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_d - \boldsymbol{J}_A(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{3.69}$$

Notice in (3.69) that the use of operational space quantities has naturally lead to using the analytical Jacobian in lieu of the geometric Jacobian. For this equation to lead to an *inverse kinematics algorithm*, it is worth relating the computed joint velocity vector $\dot{\boldsymbol{q}}$ to the error $\boldsymbol{e}$ so that (3.69) gives a differential equation describing error evolution over time. Nonetheless, it is necessary to choose a relationship between $\dot{\boldsymbol{q}}$ and $\boldsymbol{e}$ that ensures convergence of the error to zero.

Having formulated inverse kinematics in algorithmic terms implies that the joint variables $\boldsymbol{q}$ corresponding to a given end-effector pose $\boldsymbol{x}_d$ are accurately computed only when the error $\boldsymbol{x}_d - \boldsymbol{k}(\boldsymbol{q})$ is reduced within a given threshold; such settling time depends on the dynamic characteristics of the error differential equation. The choice of $\dot{\boldsymbol{q}}$ as a function of $\boldsymbol{e}$ permits finding inverse kinematics algorithms with different features.

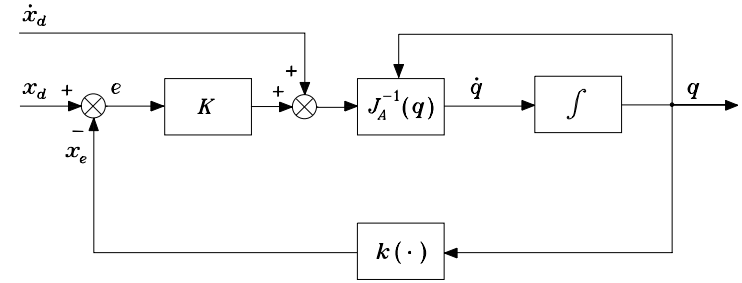

Fig. 3.11. Inverse kinematics algorithm with Jacobian inverse

3.7.1 Jacobian (Pseudo-)inverse

On the assumption that matrix $\boldsymbol{J}_A$ is square and nonsingular, the choice

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_A^{-1}(\boldsymbol{q})(\dot{\boldsymbol{x}}_d + \boldsymbol{K}\boldsymbol{e}) \tag{3.70}$$

leads to the equivalent linear system

$$\dot{\boldsymbol{e}} + \boldsymbol{K}\boldsymbol{e} = \boldsymbol{0}. \tag{3.71}$$

If $\boldsymbol{K}$ is a positive definite (usually diagonal) matrix, the system (3.71) is *asymptotically stable*. The error tends to zero along the trajectory with a convergence rate that depends on the eigenvalues of matrix $\boldsymbol{K}$;[11] the larger the eigenvalues, the faster the convergence. Since the scheme is practically implemented as a discrete-time system, it is reasonable to predict that an upper bound exists on the eigenvalues; depending on the sampling time, there will be a limit for the maximum eigenvalue of $\boldsymbol{K}$ under which asymptotic stability of the error system is guaranteed.

The block scheme corresponding to the inverse kinematics algorithm in (3.70) is illustrated in Fig. 3.11, where $\boldsymbol{k}(\cdot)$ indicates the direct kinematics function in (2.82). This scheme can be revisited in terms of the usual feedback control schemes. Specifically, it can observed that the nonlinear block $\boldsymbol{k}(\cdot)$ is needed to compute $\boldsymbol{x}$ and thus the tracking error $\boldsymbol{e}$, while the block $\boldsymbol{J}_A^{-1}(\boldsymbol{q})$ has been introduced to compensate for $\boldsymbol{J}_A(\boldsymbol{q})$ and making the system linear. The block scheme shows the presence of a string of integrators on the forward loop and then, for a constant reference ($\dot{\boldsymbol{x}}_d = \boldsymbol{0}$), guarantees a null steady-state error. Further, the *feedforward* action provided by $\dot{\boldsymbol{x}}_d$ for a time-varying reference ensures that the error is kept to zero (in the case $\boldsymbol{e}(0) = \boldsymbol{0}$) along the whole trajectory, independently of the type of desired reference $\boldsymbol{x}_d(t)$.

Finally, notice that (3.70), for $\dot{\boldsymbol{x}}_d = \boldsymbol{0}$, corresponds to the Newton method for solving a system of nonlinear equations. Given a constant end-effector pose $\boldsymbol{x}_d$, the algorithm can be keenly applied to compute one of the admissible

[11] See Sect. A.5.

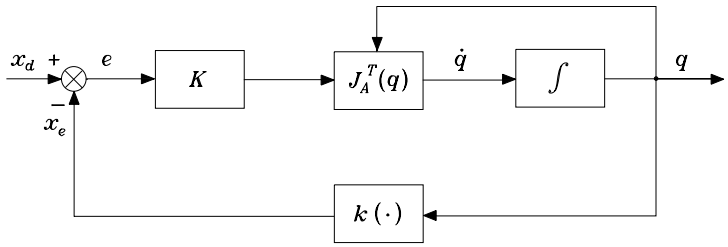

Fig. 3.12. Block scheme of the inverse kinematics algorithm with Jacobian transpose

solutions to the inverse kinematics problem, whenever that does not admit closed-form solutions, as discussed in Sect. 2.12. Such a method is also useful in practice at the start-up of the manipulator for a given task, to compute the corresponding joint configuration.

In the case of a *redundant manipulator*, solution (3.70) can be generalized into

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_A^{\dagger}(\dot{\boldsymbol{x}}_d + \boldsymbol{K}\boldsymbol{e}) + (\boldsymbol{I}_n - \boldsymbol{J}_A^{\dagger}\boldsymbol{J}_A)\dot{\boldsymbol{q}}_0, \tag{3.72}$$

which represents the algorithmic version of solution (3.54).

The structure of the inverse kinematics algorithm can be conceptually adopted for a simple robot control technique, known under the name of *kinematic control*. As will be seen in Chap. 7, a manipulator is actually an electromechanical system actuated by motor torques, while in Chaps. 8–10 dynamic control techniques will be presented which will properly account for the nonlinear and coupling effects of the dynamic model.

At first approximation, however, it is possible to consider a kinematic command as system input, typically a velocity. This is possible in view of the presence of a low-level control loop, which 'ideally' imposes any specified reference velocity. On the other hand, such a loop already exists in a 'closed' control unit, where the user can also intervene with kinematic commands. In other words, the scheme in Fig. 3.11 can implement a kinematic control, provided that the integrator is regarded as a simplified model of the robot, thanks to the presence of single joint local servos, which ensure a more or less accurate reproduction of the velocity commands. Nevertheless, it is worth underlining that such a kinematic control technique yields satisfactory performance only when one does not require too fast motions or rapid accelerations. The performance of the independent joint control will be analyzed in Sect. 8.3.

3.7.2 Jacobian Transpose

A computationally simpler algorithm can be derived by finding a relationship between $\dot{\boldsymbol{q}}$ and $\boldsymbol{e}$ that ensures error convergence to zero, without requiring linearization of (3.69). As a consequence, the error dynamics is governed by a

nonlinear differential equation. The Lyapunov direct method can be utilized to determine a dependence $\dot{\boldsymbol{q}}(\boldsymbol{e})$ that ensures asymptotic stability of the error system. Choose as Lyapunov function candidate the positive definite quadratic form[12]

$$V(\boldsymbol{e}) = \frac{1}{2}\boldsymbol{e}^T\boldsymbol{K}\boldsymbol{e}, \tag{3.73}$$

where $\boldsymbol{K}$ is a symmetric positive definite matrix. This function is so that

$$V(\boldsymbol{e}) > 0 \quad \forall \boldsymbol{e} \neq \boldsymbol{0}, \qquad V(\boldsymbol{0}) = 0.$$

Differentiating (3.73) with respect to time and accounting for (3.68) gives

$$\dot{V} = \boldsymbol{e}^T\boldsymbol{K}\dot{\boldsymbol{x}}_d - \boldsymbol{e}^T\boldsymbol{K}\dot{\boldsymbol{x}}_e. \tag{3.74}$$

In view of (3.62), it is

$$\dot{V} = \boldsymbol{e}^T\boldsymbol{K}\dot{\boldsymbol{x}}_d - \boldsymbol{e}^T\boldsymbol{K}\boldsymbol{J}_A(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{3.75}$$

At this point, the choice of joint velocities as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{K}\boldsymbol{e} \tag{3.76}$$

leads to

$$\dot{V} = \boldsymbol{e}^T\boldsymbol{K}\dot{\boldsymbol{x}}_d - \boldsymbol{e}^T\boldsymbol{K}\boldsymbol{J}_A(\boldsymbol{q})\boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{K}\boldsymbol{e}. \tag{3.77}$$

Consider the case of a constant reference ($\dot{\boldsymbol{x}}_d = \boldsymbol{0}$). The function in (3.77) is negative definite, under the assumption of full rank for $\boldsymbol{J}_A(\boldsymbol{q})$. The condition $\dot{V} < 0$ with $V > 0$ implies that the system trajectories uniformly converge to $\boldsymbol{e} = \boldsymbol{0}$, i.e., the system is *asymptotically stable*. When $\mathcal{N}(\boldsymbol{J}_A^T) \neq \emptyset$, the function in (3.77) is only negative semi-definite, since $\dot{V} = 0$ for $\boldsymbol{e} \neq \boldsymbol{0}$ with $\boldsymbol{K}\boldsymbol{e} \in \mathcal{N}(\boldsymbol{J}_A^T)$. In this case, the algorithm can get stuck at $\dot{\boldsymbol{q}} = \boldsymbol{0}$ with $\boldsymbol{e} \neq \boldsymbol{0}$. However, the example that follows will show that this situation occurs only if the assigned end-effector position is not actually reachable from the current configuration.

The resulting block scheme is illustrated in Fig. 3.12, which shows the notable feature of the algorithm to require *computation only of direct kinematics functions* $\boldsymbol{k}(\boldsymbol{q})$, $\boldsymbol{J}_A^T(\boldsymbol{q})$.

It can be recognized that (3.76) corresponds to the gradient method for the solution of a system on nonlinear equations. As in the case of the Jacobian inverse solution, for a given constant end-effector pose $\boldsymbol{x}_d$, the Jacobian transpose algorithm can be keenly employed to solve the inverse kinematics problem, or more simply to initialize the values of the manipulator joint variables.

The case when $\boldsymbol{x}_d$ is a time-varying function ($\dot{\boldsymbol{x}}_d \neq \boldsymbol{0}$) deserves a separate analysis. In order to obtain $\dot{V} < 0$ also in this case, it would be sufficient to choose a $\dot{\boldsymbol{q}}$ that depends on the (pseudo-)inverse of the Jacobian as in (3.70),

[12] See Sect. C.3 for the presentation of the Lyapunov direct method.

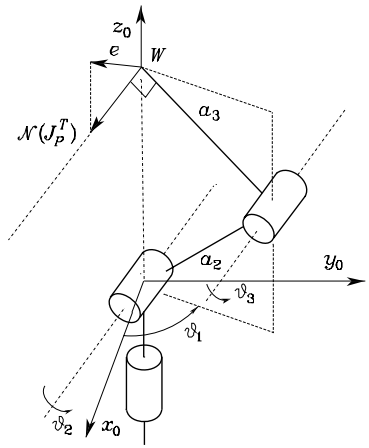

Fig. 3.13. Characterization of the anthropomorphic arm at a shoulder singularity for the admissible solutions of the Jacobian transpose algorithm

recovering the asymptotic stability result derived above.[13] For the inversion scheme based on the transpose, the first term on the right-hand side of (3.77) is not cancelled any more and nothing can be said about its sign. This implies that asymptotic stability along the trajectory cannot be achieved. The tracking error $\boldsymbol{e}(t)$ is, anyhow, norm-bounded; the larger the norm of $\boldsymbol{K}$, the smaller the norm of $\boldsymbol{e}$.[14] In practice, since the inversion scheme is to be implemented in discrete-time, there is an upper bound on the norm of $\boldsymbol{K}$ with reference to the adopted sampling time.

Example 3.4

Consider the anthropomorphic arm; a shoulder singularity occurs whenever $a_2c_2 + a_3c_{23} = 0$ (Fig. 3.6). In this configuration, the transpose of the Jacobian in (3.38) is

$$\boldsymbol{J}_P^T = \begin{bmatrix} 0 & 0 & 0 \\ -c_1(a_2s_2 + a_3s_{23}) & -s_1(a_2s_2 + a_3s_{23}) & 0 \\ -a_3c_1s_{23} & -a_3s_1s_{23} & a_3c_{23} \end{bmatrix}.$$

By computing the null space of $\boldsymbol{J}_P^T$, if ν_x, ν_y and ν_z denote the components of vector $\boldsymbol{\nu}$ along the axes of the base frame, one has the result

$$\frac{\nu_y}{\nu_x} = -\frac{1}{\tan\vartheta_1} \qquad \nu_z = 0,$$

[13] Notice that, anyhow, in case of kinematic singularities, it is necessary to resort to an inverse kinematics scheme that does not require inversion of the Jacobian.

[14] Notice that the negative definite term is a quadratic function of the error, while the other term is a linear function of the error. Therefore, for an error of very small norm, the linear term prevails over the quadratic term, and the norm of $\boldsymbol{K}$ should be increased to reduce the norm of $\boldsymbol{e}$ as much as possible.

implying that the direction of $\mathcal{N}(\boldsymbol{J}_P^T)$ coincides with the direction orthogonal to the plane of the structure (Fig. 3.13). The Jacobian transpose algorithm gets stuck if, with $\boldsymbol{K}$ diagonal and having all equal elements, the desired position is along the line normal to the plane of the structure at the intersection with the wrist point. On the other hand, the end-effector cannot physically move from the singular configuration along such a line. Instead, if the prescribed path has a non-null component in the plane of the structure at the singularity, algorithm convergence is ensured, since in that case $\boldsymbol{K}\boldsymbol{e} \notin \mathcal{N}(\boldsymbol{J}_P^T)$.

In summary, the algorithm based on the computation of the Jacobian transpose provides a computationally efficient inverse kinematics method that can be utilized also for paths crossing kinematic singularities.

3.7.3 Orientation Error

The inverse kinematics algorithms presented in the above sections utilize the analytical Jacobian since they operate on error variables (position and orientation) that are defined in the operational space.

For what concerns the position error, it is obvious that its expression is given by

$$\boldsymbol{e}_P = \boldsymbol{p}_d - \boldsymbol{p}_e(\boldsymbol{q}) \tag{3.78}$$

where $\boldsymbol{p}_d$ and $\boldsymbol{p}_e$ denote respectively the desired and computed end-effector positions. Further, its time derivative is

$$\dot{\boldsymbol{e}}_P = \dot{\boldsymbol{p}}_d - \dot{\boldsymbol{p}}_e. \tag{3.79}$$

On the other hand, for what concerns the *orientation error*, its expression depends on the particular representation of end-effector orientation, namely, Euler angles, angle and axis, and unit quaternion.

Euler angles

The orientation error is chosen according to an expression formally analogous to (3.78), i.e.,

$$\boldsymbol{e}_O = \boldsymbol{\phi}_d - \boldsymbol{\phi}_e(\boldsymbol{q}) \tag{3.80}$$

where $\boldsymbol{\phi}_d$ and $\boldsymbol{\phi}_e$ denote respectively the desired and computed set of Euler angles. Further, its time derivative is

$$\dot{\boldsymbol{e}}_O = \dot{\boldsymbol{\phi}}_d - \dot{\boldsymbol{\phi}}_e. \tag{3.81}$$

Therefore, assuming that neither kinematic nor representation singularities occur, the Jacobian inverse solution for a nonredundant manipulator is derived from (3.70), i.e.,

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_A^{-1}(\boldsymbol{q}) \begin{bmatrix} \dot{\boldsymbol{p}}_d + \boldsymbol{K}_P \boldsymbol{e}_P \\ \dot{\boldsymbol{\phi}}_d + \boldsymbol{K}_O \boldsymbol{e}_O \end{bmatrix} \tag{3.82}$$

where $\boldsymbol{K}_P$ and $\boldsymbol{K}_O$ are positive definite matrices.

As already pointed out in Sect. 2.10 for computation of the direct kinematics function in the form (2.82), the determination of the orientation variables from the joint variables is not easy except for simple cases (see Example 2.5). To this end, it is worth recalling that computation of the angles $\boldsymbol{\phi}_e$, in a minimal representation of orientation, requires computation of the rotation matrix $\boldsymbol{R}_e = [\,\boldsymbol{n}_e \quad \boldsymbol{s}_e \quad \boldsymbol{a}_e\,]$; in fact, only the dependence of $\boldsymbol{R}_e$ on $\boldsymbol{q}$ is known in closed form, but not that of $\boldsymbol{\phi}_e$ on $\boldsymbol{q}$. Further, the use of inverse functions (Atan2) in (2.19), (2.22) involves a non-negligible complexity in the computation of the analytical Jacobian, and the occurrence of representation singularities constitutes another drawback for the orientation error based on Euler angles.

Different kinds of remarks are to be made about the way to assign a time profile for the reference variables $\boldsymbol{\phi}_d$ chosen to represent end-effector orientation. The most intuitive way to specify end-effector orientation is to refer to the orientation of the end-effector frame ($\boldsymbol{n}_d$, $\boldsymbol{s}_d$, $\boldsymbol{a}_d$) with respect to the base frame. Given the limitations pointed out in Sect. 2.10 about guaranteeing orthonormality of the unit vectors along time, it is necessary first to compute the Euler angles corresponding to the initial and final orientation of the end-effector frame via (2.19), (2.22); only then a time evolution can be generated. Such solutions will be presented in Chap. 4.

A radical simplification of the problem at issue can be obtained for manipulators having a spherical wrist. Section 2.12.2 pointed out the possibility to solve the inverse kinematics problem for the position part separately from that for the orientation part. This result also has an impact at algorithmic level. In fact, the implementation of an inverse kinematics algorithm for determining the joint variables influencing the wrist position allows the computation of the time evolution of the wrist frame $\boldsymbol{R}_W(t)$. Hence, once the desired time evolution of the end-effector frame $\boldsymbol{R}_d(t)$ is given, it is sufficient to compute the Euler angles ZYZ from the matrix $\boldsymbol{R}_W^T \boldsymbol{R}_d$ by applying (2.19). As shown in Sect. 2.12.5, these angles are directly the joint variables of the spherical wrist. See also Problem 3.14.

The above considerations show that the inverse kinematics algorithms based on the analytical Jacobian are effective for kinematic structures having a spherical wrist which are of significant interest. For manipulator structures which cannot be reduced to that class, it may be appropriate to reformulate the inverse kinematics problem on the basis of a different definition of the orientation error.

Angle and axis

If $\boldsymbol{R}_d = [\,\boldsymbol{n}_d \quad \boldsymbol{s}_d \quad \boldsymbol{a}_d\,]$ denotes the desired rotation matrix of the end-effector frame and $\boldsymbol{R}_e = [\,\boldsymbol{n}_e \quad \boldsymbol{s}_e \quad \boldsymbol{a}_e\,]$ the rotation matrix that can be computed from the joint variables, the orientation error between the two frames can be expressed as

$$\boldsymbol{e}_O = \boldsymbol{r} \sin\vartheta \tag{3.83}$$

where ϑ and $\boldsymbol{r}$ identify the *angle and axis* of the equivalent rotation that can be deduced from the matrix

$$\boldsymbol{R}(\vartheta, \boldsymbol{r}) = \boldsymbol{R}_d \boldsymbol{R}_e^T(\boldsymbol{q}), \tag{3.84}$$

describing the rotation needed to align $\boldsymbol{R}$ with $\boldsymbol{R}_d$. Notice that (3.83) gives a unique relationship for $-\pi/2 < \vartheta < \pi/2$. The angle ϑ represents the magnitude of an orientation error, and thus the above limitation is not restrictive since the tracking error is typically small for an inverse kinematics algorithm.

By comparing the off-diagonal terms of the expression of $\boldsymbol{R}(\vartheta, \boldsymbol{r})$ in (2.25) with the corresponding terms resulting on the right-hand side of (3.84), it can be found that a functional expression of the orientation error in (3.83) is (see Problem 3.16)

$$\boldsymbol{e}_O = \frac{1}{2}(\boldsymbol{n}_e(\boldsymbol{q}) \times \boldsymbol{n}_d + \boldsymbol{s}_e(\boldsymbol{q}) \times \boldsymbol{s}_d + \boldsymbol{a}_e(\boldsymbol{q}) \times \boldsymbol{a}_d); \tag{3.85}$$

the limitation on ϑ is transformed in the condition $\boldsymbol{n}_e^T \boldsymbol{n}_d \geq 0$, $\boldsymbol{s}_e^T \boldsymbol{s}_d \geq 0$, $\boldsymbol{a}_e^T \boldsymbol{a}_d \geq 0$.

Differentiating (3.85) with respect to time and accounting for the expression of the columns of the derivative of a rotation matrix in (3.8) gives (see Problem 3.19)

$$\dot{\boldsymbol{e}}_O = \boldsymbol{L}^T \boldsymbol{\omega}_d - \boldsymbol{L} \boldsymbol{\omega}_e \tag{3.86}$$

where

$$\boldsymbol{L} = -\frac{1}{2}\big(\boldsymbol{S}(\boldsymbol{n}_d)\boldsymbol{S}(\boldsymbol{n}_e) + \boldsymbol{S}(\boldsymbol{s}_d)\boldsymbol{S}(\boldsymbol{s}_e) + \boldsymbol{S}(\boldsymbol{a}_d)\boldsymbol{S}(\boldsymbol{a}_e)\big). \tag{3.87}$$

At this point, by exploiting the relations (3.2), (3.3) of the geometric Jacobian expressing $\dot{\boldsymbol{p}}_e$ and $\boldsymbol{\omega}_e$ as a function of $\dot{\boldsymbol{q}}$, (3.79), (3.86) become

$$\dot{\boldsymbol{e}} = \begin{bmatrix} \dot{\boldsymbol{e}}_P \\ \dot{\boldsymbol{e}}_O \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{p}}_d - \boldsymbol{J}_P(\boldsymbol{q})\dot{\boldsymbol{q}} \\ \boldsymbol{L}^T \boldsymbol{\omega}_d - \boldsymbol{L}\boldsymbol{J}_O(\boldsymbol{q})\dot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{p}}_d \\ \boldsymbol{L}^T \boldsymbol{\omega}_d \end{bmatrix} - \begin{bmatrix} \boldsymbol{I} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{L} \end{bmatrix} \boldsymbol{J}\dot{\boldsymbol{q}}. \tag{3.88}$$

The expression in (3.88) suggests the possibility of devising inverse kinematics algorithms analogous to the ones derived above, but using the geometric Jacobian in place of the analytical Jacobian. For instance, the Jacobian inverse solution for a nonredundant nonsingular manipulator is

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q}) \begin{bmatrix} \dot{\boldsymbol{p}}_d + \boldsymbol{K}_P \boldsymbol{e}_P \\ \boldsymbol{L}^{-1}\left(\boldsymbol{L}^T \boldsymbol{\omega}_d + \boldsymbol{K}_O \boldsymbol{e}_O\right) \end{bmatrix}. \tag{3.89}$$

It is worth remarking that the inverse kinematics solution based on (3.89) is expected to perform better than the solution based on (3.82) since it uses the geometric Jacobian in lieu of the analytical Jacobian, thus avoiding the occurrence of representation singularities.

Unit quaternion

In order to devise an inverse kinematics algorithm based on the *unit quaternion*, a suitable orientation error should be defined. Let $\mathcal{Q}_d = \{\eta_d, \boldsymbol{\epsilon}_d\}$ and $\mathcal{Q}_e = \{\eta_e, \boldsymbol{\epsilon}_e\}$ represent the quaternions associated with $\boldsymbol{R}_d$ and $\boldsymbol{R}_e$, respectively. The orientation error can be described by the rotation matrix $\boldsymbol{R}_d\boldsymbol{R}_e^T$ and, in view of (2.37), can be expressed in terms of the quaternion $\Delta\mathcal{Q} = \{\Delta\eta, \Delta\boldsymbol{\epsilon}\}$ where

$$\Delta\mathcal{Q} = \mathcal{Q}_d * \mathcal{Q}_e^{-1}. \tag{3.90}$$

It can be recognized that $\Delta\mathcal{Q} = \{1, \boldsymbol{0}\}$ if and only if $\boldsymbol{R}_e$ and $\boldsymbol{R}_d$ are aligned. Hence, it is sufficient to define the orientation error as

$$\boldsymbol{e}_O = \Delta\boldsymbol{\epsilon} = \eta_e(\boldsymbol{q})\boldsymbol{\epsilon}_d - \eta_d\boldsymbol{\epsilon}_e(\boldsymbol{q}) - \boldsymbol{S}(\boldsymbol{\epsilon}_d)\boldsymbol{\epsilon}_e(\boldsymbol{q}), \tag{3.91}$$

where the skew-symmetric operator $\boldsymbol{S}(\cdot)$ has been used. Notice, however, that the explicit computation of η_e and $\boldsymbol{\epsilon}_e$ from the joint variables is not possible but it requires the intermediate computation of the rotation matrix $\boldsymbol{R}_e$ that is available from the manipulator direct kinematics; then, the quaternion can be extracted using (2.34).

At this point, a Jacobian inverse solution can be computed as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q}) \begin{bmatrix} \dot{\boldsymbol{p}}_d + \boldsymbol{K}_P\boldsymbol{e}_P \\ \boldsymbol{\omega}_d + \boldsymbol{K}_O\boldsymbol{e}_O \end{bmatrix} \tag{3.92}$$

where noticeably the geometric Jacobian has been used. Substituting (3.92) into (3.4) gives (3.79) and

$$\boldsymbol{\omega}_d - \boldsymbol{\omega}_e + \boldsymbol{K}_O\boldsymbol{e}_O = \boldsymbol{0}. \tag{3.93}$$

It should be observed that now the orientation error equation is nonlinear in $\boldsymbol{e}_O$ since it contains the end-effector angular velocity error instead of the time derivative of the orientation error. To this end, it is worth considering the relationship between the time derivative of the quaternion $\mathcal{Q}_e$ and the angular velocity $\boldsymbol{\omega}_e$. This can be found to be (see Problem 3.19)

$$\dot{\eta}_e = -\frac{1}{2}\boldsymbol{\epsilon}_e^T\boldsymbol{\omega}_e \tag{3.94}$$

$$\dot{\boldsymbol{\epsilon}}_e = \frac{1}{2}\left(\eta_e\boldsymbol{I}_3 - \boldsymbol{S}(\boldsymbol{\epsilon}_e)\right)\boldsymbol{\omega}_e \tag{3.95}$$

which is the so-called *quaternion propagation*. A similar relationship holds between the time derivative of $\mathcal{Q}_d$ and $\boldsymbol{\omega}_d$.

To study stability of system (3.93), consider the positive definite Lyapunov function candidate

$$V = (\eta_d - \eta_e)^2 + (\boldsymbol{\epsilon}_d - \boldsymbol{\epsilon}_e)^T(\boldsymbol{\epsilon}_d - \boldsymbol{\epsilon}_e). \tag{3.96}$$

In view of (3.94), (3.95), differentiating (3.96) with respect to time and accounting for (3.93) yields (see Problem 3.20)

$$\dot{V} = -\boldsymbol{e}_O^T\boldsymbol{K}_O\boldsymbol{e}_O \tag{3.97}$$

which is negative definite, implying that $\boldsymbol{e}_O$ converges to zero.

In summary, the inverse kinematics solution based on (3.92) uses the geometric Jacobian as the solution based on (3.89) but is computationally lighter.

3.7.4 Second-order Algorithms

The above inverse kinematics algorithms can be defined as *first-order* algorithms, in that they allow the inversion of a motion trajectory, specified at the end-effector in terms of of position and orientation, into the equivalent joint positions and velocities.

Nevertheless, as will be seen in Chap. 8, for control purposes it may be necessary to invert a motion trajectory specified in terms of position, velocity and acceleration. On the other hand, the manipulator is inherently a *second-order* mechanical system, as will be revealed by the dynamic model to be derived in Chap. 7.

The time differentiation of the differential kinematics equation (3.62) leads to

$$\ddot{\boldsymbol{x}}_e = \boldsymbol{J}_A(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_A(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \tag{3.98}$$

which gives the relationship between the joint space accelerations and the operational space accelerations.

Under the assumption of a square and non-singular matrix $\boldsymbol{J}_A$, the second-order differential kinematics (3.98) can be inverted in terms of the joint accelerations

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_A^{-1}(\boldsymbol{q})\left(\ddot{\boldsymbol{x}}_e - \dot{\boldsymbol{J}}_A(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}\right). \tag{3.99}$$

The numerical integration of (3.99) to reconstruct the joint velocities and positions would unavoidably lead to a drift of the solution; therefore, similarly to the inverse kinematics algorithm with the Jacobian inverse, it is worth considering the error defined in (3.68) along with its derivative

$$\ddot{\boldsymbol{e}} = \ddot{\boldsymbol{x}}_d - \ddot{\boldsymbol{x}}_e \tag{3.100}$$

which, in view of (3.98), yields

$$\ddot{\boldsymbol{e}} = \ddot{\boldsymbol{x}}_d - \boldsymbol{J}_A(\boldsymbol{q})\ddot{\boldsymbol{q}} - \dot{\boldsymbol{J}}_A(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}. \tag{3.101}$$

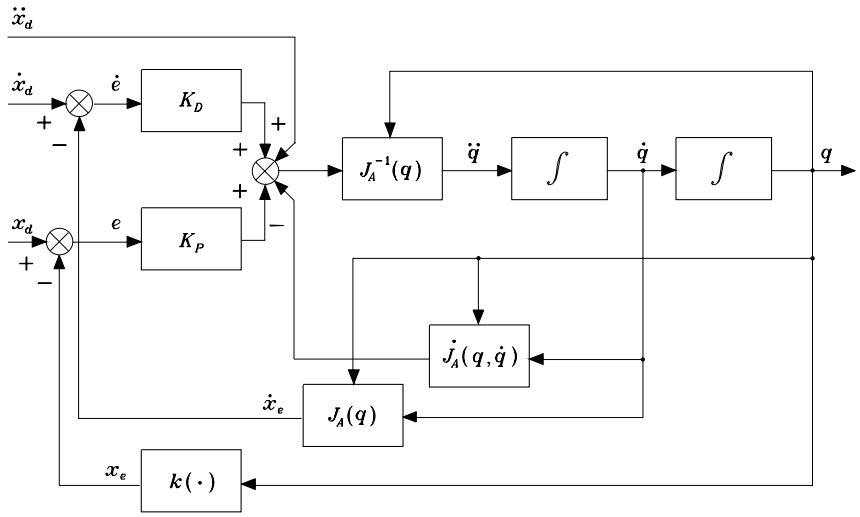

Fig. 3.14. Block scheme of the second-order inverse kinematics algorithm with Jacobian inverse

At this point, it is advisable to choose the joint acceleration vector as

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_A^{-1}(\boldsymbol{q}) \left(\ddot{\boldsymbol{x}}_d + \boldsymbol{K}_D \dot{\boldsymbol{e}} + \boldsymbol{K}_P \boldsymbol{e} - \dot{\boldsymbol{J}}_A(\boldsymbol{q}, \dot{\boldsymbol{q}}) \dot{\boldsymbol{q}} \right) \tag{3.102}$$

where $\boldsymbol{K}_D$ and $\boldsymbol{K}_P$ are positive definite (typically diagonal) matrices. Substituting (3.102) into (3.101) leads to the equivalent linear error system

$$\ddot{\boldsymbol{e}} + \boldsymbol{K}_D \dot{\boldsymbol{e}} + \boldsymbol{K}_P \boldsymbol{e} = \boldsymbol{0} \tag{3.103}$$

which is *asymptotically stable*: the error tends to zero along the trajectory with a convergence speed depending on the choice of the matrices $\boldsymbol{K}_P$ e $\boldsymbol{K}_D$. The second-order inverse kinematics algorithm is illustrated in the block scheme of Fig. 3.14.

In the case of a *redundant manipulator*, the generalization of (3.102) leads to an algorithmic solution based on the Jacobian pseudo-inverse of the kind

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}_A^{\dagger} \left(\ddot{\boldsymbol{x}}_d + \boldsymbol{K}_D \dot{\boldsymbol{e}} + \boldsymbol{K}_P \boldsymbol{e} - \dot{\boldsymbol{J}}_A(\boldsymbol{q}, \dot{\boldsymbol{q}}) \dot{\boldsymbol{q}} \right) + (\boldsymbol{I}_n - \boldsymbol{J}_A^{\dagger} \boldsymbol{J}_A) \ddot{\boldsymbol{q}}_0 \tag{3.104}$$

where the vector $\ddot{\boldsymbol{q}}_0$ represents arbitrary joint accelerations which can be chosen so as to (locally) optimize an objective function like those considered in Sect. 3.5.1.

As for the first-order inverse kinematics algorithms, it is possible to consider other expressions for the orientation error which, unlike the Euler angles, refer to an angle and axis description, else to the unit quaternion.

3.7.5 Comparison Among Inverse Kinematics Algorithms

In order to make a comparison of performance among the inverse kinematics algorithms presented above, consider the 3-link planar arm in Fig. 2.20 whose link lengths are $a_1 = a_2 = a_3 = 0.5\,\text{m}$. The direct kinematics for this arm is given by (2.83), while its Jacobian can be found from (3.35) by considering the 3 non-null rows of interest for the operational space.

Let the arm be at the initial posture $\boldsymbol{q} = [\,\pi \quad -\pi/2 \quad -\pi/2\,]^T\,\text{rad}$, corresponding to the end-effector pose: $\boldsymbol{p} = [\,0 \quad 0.5\,]^T\,\text{m}$, $\phi = 0\,\text{rad}$. A circular path of radius 0.25 m and centre at $(0.25, 0.5)\,\text{m}$ is assigned to the end-effector. Let the motion trajectory be

$$\boldsymbol{p}_d(t) = \begin{bmatrix} 0.25(1 - \cos \pi t) \\ 0.25(2 + \sin \pi t) \end{bmatrix} \qquad 0 \le t \le 4;$$

i.e., the end-effector has to make two complete circles in a time of 2 s per circle. As regards end-effector orientation, initially it is required to follow the trajectory

$$\phi_d(t) = \sin \frac{\pi}{24} t \qquad 0 \le t \le 4;$$

i.e., the end-effector has to attain a different orientation ($\phi_d = 0.5\,\text{rad}$) at the end of the two circles.

The inverse kinematics algorithms were implemented on a computer by adopting the Euler numerical integration scheme (3.48) with an integration time $\Delta t = 1\,\text{ms}$.

At first, the inverse kinematics along the given trajectory has been performed by using (3.47). The results obtained in Fig. 3.15 show that the norm of the position error along the whole trajectory is bounded; at steady state, after $t = 4$, the error sets to a constant value in view of the typical *drift* of *open-loop* schemes. A similar drift can be observed for the orientation error.

Next, the inverse kinematics algorithm based on (3.70) using the Jacobian *inverse* has been used, with the matrix gain $\boldsymbol{K} = \text{diag}\{500, 500, 100\}$. The resulting joint positions and velocities as well as the tracking errors are shown in Fig. 3.16. The norm of the position error is radically decreased and converges to zero at steady state, thanks to the *closed-loop* feature of the scheme; the orientation error, too, is decreased and tends to zero at steady state.

On the other hand, if the end-effector orientation is not constrained, the operational space becomes two-dimensional and is characterized by the first two rows of the direct kinematics in (2.83) as well as by the Jacobian in (3.36); a *redundant* DOF is then available. Hence, the inverse kinematics algorithm based on (3.72) using the Jacobian *pseudo-inverse* has been used with $\boldsymbol{K} = \text{diag}\{500, 500\}$. If redundancy is not exploited ($\dot{\boldsymbol{q}}_0 = \boldsymbol{0}$), the results in Fig. 3.17 reveal that position tracking remains satisfactory and, of course, the end-effector orientation freely varies along the given trajectory.

With reference to the previous situation, the use of the Jacobian *transpose* algorithm based on (3.76) with $\boldsymbol{K} = \text{diag}\{500, 500\}$ gives rise to a tracking

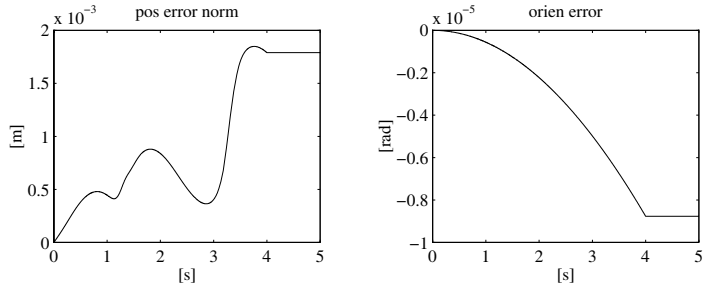

Fig. 3.15. Time history of the norm of end-effector position error and orientation error with the open-loop inverse Jacobian algorithm

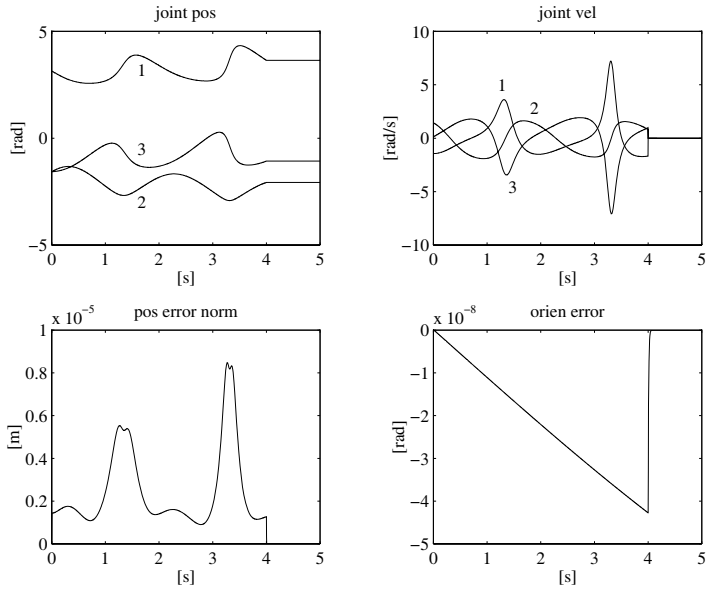

Fig. 3.16. Time history of the joint positions and velocities, and of the norm of end-effector position error and orientation error with the closed-loop inverse Jacobian algorithm

error (Fig. 3.18) which is anyhow bounded and rapidly tends to zero at steady state.

In order to show the capability of handling the degree of redundancy, the algorithm based on (3.72) with $\dot{\boldsymbol{q}}_0 \neq \mathbf{0}$ has been used; two types of constraints

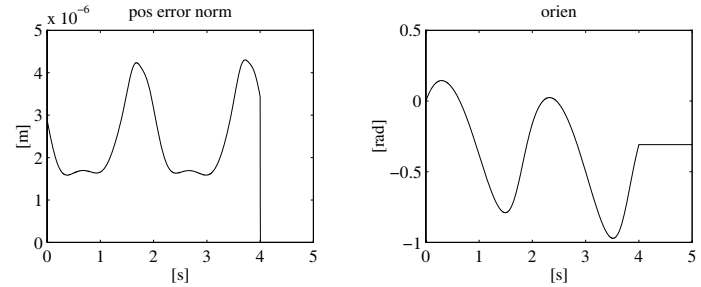

Fig. 3.17. Time history of the norm of end-effector position error and orientation with the Jacobian pseudo-inverse algorithm

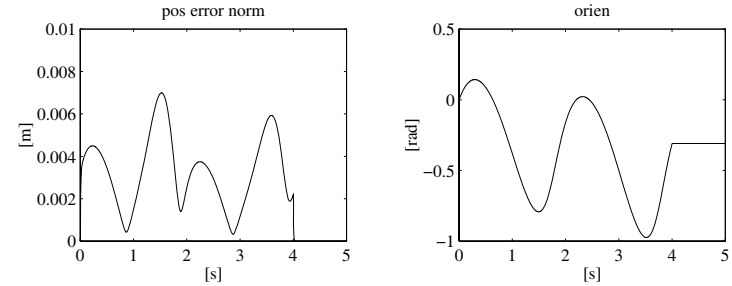

Fig. 3.18. Time history of the norm of end-effector position error and orientation with the Jacobian transpose algorithm

have been considered concerning an objective function to locally maximize according to the choice (3.55). The first function is

$$w(\vartheta_2, \vartheta_3) = \frac{1}{2}(s_2^2 + s_3^2)$$

that provides a *manipulability measure*. Notice that such a function is computationally simpler than the function in (3.56), but it still describes a distance from kinematic singularities in an effective way. The gain in (3.55)) has been set to $k_0 = 50$. In Fig. 3.19, the joint trajectories are reported for the two cases with and without ($k_0 = 0$) constraint. The addition of the constraint leads to having coincident trajectories for Joints 2 and 3. The manipulability measure in the constrained case (*continuous line*) attains larger values along the trajectory compared to the unconstrained case (*dashed line*). It is worth underlining that the tracking position error is practically the same in the two cases (Fig. 3.17), since the additional joint velocity contribution is projected in the null space of the Jacobian so as not to alter the performance of the end-effector position task.

Finally, it is worth noticing that in the constrained case the resulting joint trajectories are *cyclic*, i.e., they take on the same values after a period of

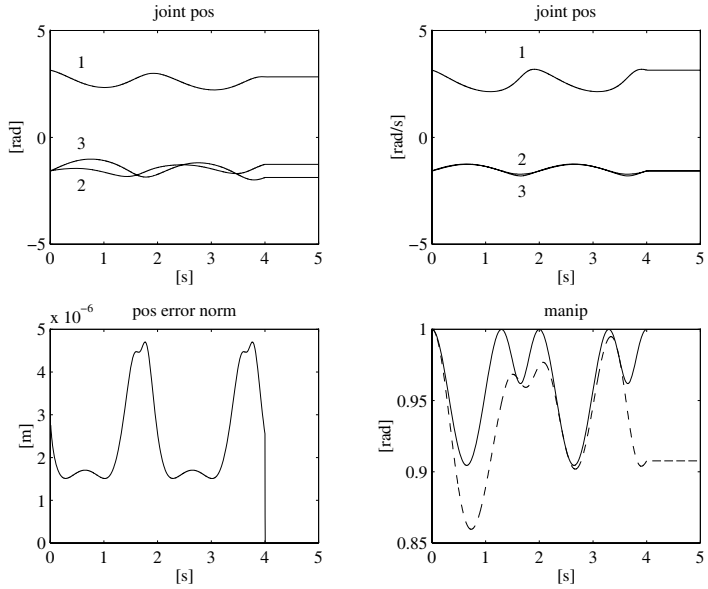

Fig. 3.19. Time history of the joint positions, the norm of end-effector position error, and the manipulability measure with the Jacobian pseudo-inverse algorithm and manipulability constraint; *upper left*: with the unconstrained solution, *upper right*: with the constrained solution

the circular path. This does not happen for the unconstrained case, since the internal motion of the structure causes the arm to be in a different posture after one circle.

The second objective function considered is the *distance from mechanical joint limits* in (3.57). Specifically, it is assumed what follows: the first joint does not have limits ($q_{1m} = -2\pi$, $q_{1M} = 2\pi$), the second joint has limits $q_{2m} = -\pi/2$, $q_{2M} = \pi/2$, and the third joint has limits $q_{3m} = -3\pi/2$, $q_{3M} = -\pi/2$. It is not difficult to verify that, in the unconstrained case, the trajectories of Joints 2 and 3 in Fig. 3.19 violate the respective limits. The gain in (3.55) has been set to $k_0 = 250$. The results in Fig. 3.20 show the effectiveness of the technique with utilization of redundancy, since both Joints 2 and 3 tend to invert their motion — with respect to the unconstrained trajectories in Fig. 3.19 — and keep far from the minimum limit for Joint 2 and the maximum limit for Joint 3, respectively. Such an effort does not appreciably affect the position tracking error, whose norm is bounded anyhow within acceptable values.

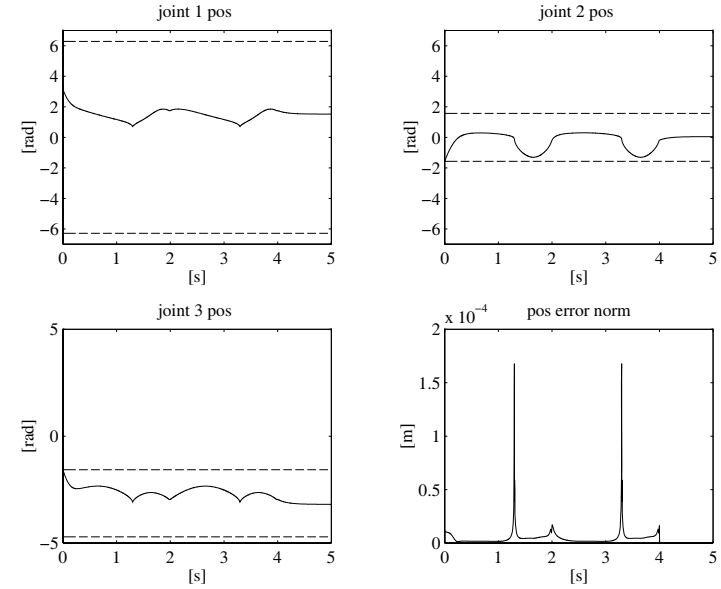

Fig. 3.20. Time history of the joint positions and the norm of end-effector position error with the Jacobian pseudo-inverse algorithm and joint limit constraint (joint limits are denoted by *dashed lines*)

3.8 Statics

The goal of *statics* is to determine the relationship between the generalized forces applied to the end-effector and the generalized forces applied to the joints — forces for prismatic joints, torques for revolute joints — with the manipulator at an equilibrium configuration.

Let $\boldsymbol{\tau}$ denote the $(n \times 1)$ vector of joint torques and $\boldsymbol{\gamma}$ the $(r \times 1)$ vector of end-effector forces[15] where r is the dimension of the operational space of interest.

The application of the *principle of virtual work* allows the determination of the required relationship. The mechanical manipulators considered are systems with time-invariant, holonomic constraints, and thus their configurations depend only on the joint variables $\boldsymbol{q}$ and not explicitly on time. This implies that virtual displacements coincide with elementary displacements.

Consider the elementary works performed by the two force systems. As for the joint torques, the elementary work associated with them is

$$dW_\tau = \boldsymbol{\tau}^T d\boldsymbol{q}. \tag{3.105}$$

[15] Hereafter, generalized forces at the joints are often called *torques*, while generalized forces at the end-effector are often called *forces*.

As for the end-effector forces $\boldsymbol{\gamma}$, if the force contributions $\boldsymbol{f}_e$ are separated by the moment contributions $\boldsymbol{\mu}_e$, the elementary work associated with them is

$$dW_\gamma = \boldsymbol{f}_e^T d\boldsymbol{p}_e + \boldsymbol{\mu}_e^T \boldsymbol{\omega}_e dt, \tag{3.106}$$

where $d\boldsymbol{p}_e$ is the linear displacement and $\boldsymbol{\omega}_e dt$ is the angular displacement[16]

By accounting for the differential kinematics relationship in (3.4), (3.5), the relation (3.106) can be rewritten as

$$\begin{aligned} dW_\gamma &= \boldsymbol{f}_e^T \boldsymbol{J}_P(\boldsymbol{q}) d\boldsymbol{q} + \boldsymbol{\mu}_e^T \boldsymbol{J}_O(\boldsymbol{q}) d\boldsymbol{q} \\ &= \boldsymbol{\gamma}_e^T \boldsymbol{J}(\boldsymbol{q}) d\boldsymbol{q} \end{aligned} \tag{3.107}$$

where $\boldsymbol{\gamma}_e = [\, \boldsymbol{f}_e^T \quad \boldsymbol{\mu}_e^T \,]^T$. Since virtual and elementary displacements coincide, the virtual works associated with the two force systems are

$$\delta W_\tau = \boldsymbol{\tau}^T \delta \boldsymbol{q} \tag{3.108}$$

$$\delta W_\gamma = \boldsymbol{\gamma}_e^T \boldsymbol{J}(\boldsymbol{q}) \delta \boldsymbol{q}, \tag{3.109}$$

where δ is the usual symbol to indicate virtual quantities.

According to the principle of virtual work, the manipulator is at *static equilibrium* if and only if

$$\delta W_\tau = \delta W_\gamma \qquad \forall \delta \boldsymbol{q}, \tag{3.110}$$

i.e., the difference between the virtual work of the joint torques and the virtual work of the end-effector forces must be null for all joint displacements.

From (3.109), notice that the virtual work of the end-effector forces is null for any displacement in the null space of $\boldsymbol{J}$. This implies that the joint torques associated with such displacements must be null at static equilibrium. Substituting (3.108), (3.109) into (3.110) leads to the notable result

$$\boldsymbol{\tau} = \boldsymbol{J}^T(\boldsymbol{q}) \boldsymbol{\gamma}_e \tag{3.111}$$

stating that the relationship between the end-effector forces and the joint torques is established by the transpose of the manipulator geometric Jacobian.

3.8.1 Kineto-Statics Duality

The statics relationship in (3.111), combined with the differential kinematics equation in (3.45), points out a property of *kineto-statics duality*. In fact, by adopting a representation similar to that of Fig. 3.7 for differential kinematics, one has that (Fig. 3.21):

- The range space of $\boldsymbol{J}^T$ is the subspace $\mathcal{R}(\boldsymbol{J}^T)$ in $\mathbb{R}^n$ of the joint torques that can balance the end-effector forces, in the given manipulator posture.

[16] The angular displacement has been indicated by $\boldsymbol{\omega}_e dt$ in view of the problems of integrability of $\boldsymbol{\omega}_e$ discussed in Sect. 3.6.

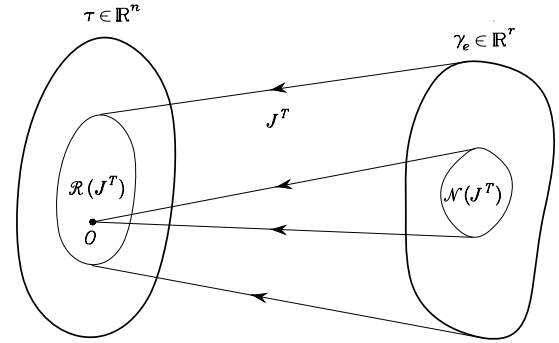

Fig. 3.21. Mapping between the end-effector force space and the joint torque space

- The null space of $\boldsymbol{J}^T$ is the subspace $\mathcal{N}(\boldsymbol{J}^T)$ in $\mathbb{R}^r$ of the end-effector forces that do not require any balancing joint torques, in the given manipulator posture.

It is worth remarking that the end-effector forces $\boldsymbol{\gamma}_e \in \mathcal{N}(\boldsymbol{J}^T)$ are entirely absorbed by the structure in that the mechanical constraint reaction forces can balance them exactly. Hence, a manipulator at a singular configuration remains in the given posture whatever end-effector force $\boldsymbol{\gamma}_e$ is applied so that $\boldsymbol{\gamma}_e \in \mathcal{N}(\boldsymbol{J}^T)$.

The relations between the two subspaces are established by

$$\mathcal{N}(\boldsymbol{J}) \equiv \mathcal{R}^\perp(\boldsymbol{J}^T) \qquad\qquad \mathcal{R}(\boldsymbol{J}) \equiv \mathcal{N}^\perp(\boldsymbol{J}^T)$$

and then, once the manipulator Jacobian is known, it is possible to characterize completely differential kinematics and statics in terms of the range and null spaces of the Jacobian and its transpose.

On the basis of the above duality, the inverse kinematics scheme with the Jacobian transpose in Fig. 3.12 admits an interesting physical interpretation. Consider a manipulator with ideal dynamics $\boldsymbol{\tau} = \dot{\boldsymbol{q}}$ (null masses and unit viscous friction coefficients); the algorithm update law $\dot{\boldsymbol{q}} = \boldsymbol{J}^T \boldsymbol{K} \boldsymbol{e}$ plays the role of a generalized spring of stiffness constant $\boldsymbol{K}$ generating a force $\boldsymbol{K}\boldsymbol{e}$ that pulls the end-effector towards the desired posture in the operational space. If this manipulator is allowed to move, e.g., in the case $\boldsymbol{K}\boldsymbol{e} \notin \mathcal{N}(\boldsymbol{J}^T)$, the end-effector attains the desired posture and the corresponding joint variables are determined.

3.8.2 Velocity and Force Transformation

The kineto-statics duality concept presented above can be useful to characterize the transformation of velocities and forces between two coordinate frames.

Consider a reference coordinate frame O_0–$x_0 y_0 z_0$ and a rigid body moving with respect to such a frame. Then let O_1–$x_1 y_1 z_1$ and O_2–$x_2 y_2 z_2$ be two

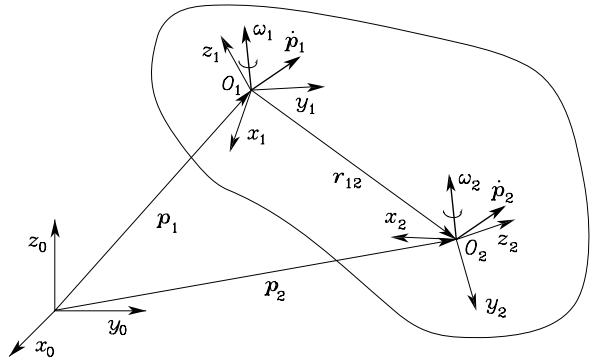

Fig. 3.22. Representation of linear and angular velocities in different coordinate frames on the same rigid body

coordinate frames attached to the body (Fig. 3.22). The relationships between translational and rotational velocities of the two frames with respect to the reference frame are given by

$$\boldsymbol{\omega}_2 = \boldsymbol{\omega}_1$$
$$\dot{\boldsymbol{p}}_2 = \dot{\boldsymbol{p}}_1 + \boldsymbol{\omega}_1 \times \boldsymbol{r}_{12}.$$

By exploiting the skew-symmetric operator $\boldsymbol{S}(\cdot)$ in (3.9), the above relations can be compactly written as

$$\begin{bmatrix} \dot{\boldsymbol{p}}_2 \\ \boldsymbol{\omega}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{S}(\boldsymbol{r}_{12}) \\ \boldsymbol{O} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{p}}_1 \\ \boldsymbol{\omega}_1 \end{bmatrix}. \tag{3.112}$$

All vectors in (3.112) are meant to be referred to the reference frame O_0–$x_0y_0z_0$. On the other hand, if vectors are referred to their own frames, it is

$$\boldsymbol{r}_{12} = \boldsymbol{R}_1 \boldsymbol{r}_{12}^1$$

and also

$$\dot{\boldsymbol{p}}_1 = \boldsymbol{R}_1 \dot{\boldsymbol{p}}_1^1 \qquad \dot{\boldsymbol{p}}_2 = \boldsymbol{R}_2 \dot{\boldsymbol{p}}_2^2 = \boldsymbol{R}_1 \boldsymbol{R}_2^1 \dot{\boldsymbol{p}}_2^2$$
$$\boldsymbol{\omega}_1 = \boldsymbol{R}_1 \boldsymbol{\omega}_1^1 \qquad \boldsymbol{\omega}_2 = \boldsymbol{R}_2 \boldsymbol{\omega}_2^2 = \boldsymbol{R}_1 \boldsymbol{R}_2^1 \boldsymbol{\omega}_2^2.$$

Accounting for (3.112) and (3.11) gives

$$\boldsymbol{R}_1 \boldsymbol{R}_2^1 \dot{\boldsymbol{p}}_2^2 = \boldsymbol{R}_1 \dot{\boldsymbol{p}}_1^1 - \boldsymbol{R}_1 \boldsymbol{S}(\boldsymbol{r}_{12}^1) \boldsymbol{R}_1^T \boldsymbol{R}_1 \boldsymbol{\omega}_1^1$$
$$\boldsymbol{R}_1 \boldsymbol{R}_2^1 \boldsymbol{\omega}_2^2 = \boldsymbol{R}_1 \boldsymbol{\omega}_1^1.$$

Eliminating the dependence on $\boldsymbol{R}_1$, which is premultiplied to each term on both sides of the previous relations, yields[17]

$$\begin{bmatrix} \dot{\boldsymbol{p}}_2^2 \\ \boldsymbol{\omega}_2^2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_1^2 & -\boldsymbol{R}_1^2 \boldsymbol{S}(\boldsymbol{r}_{12}^1) \\ \boldsymbol{O} & \boldsymbol{R}_1^2 \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{p}}_1^1 \\ \boldsymbol{\omega}_1^1 \end{bmatrix} \tag{3.113}$$

[17] Recall that $\boldsymbol{R}^T \boldsymbol{R} = \boldsymbol{I}$, as in (2.4).

giving the sought general relationship of *velocity transformation* between two frames.

It may be observed that the transformation matrix in (3.113) plays the role of a true Jacobian, since it characterizes a velocity transformation, and thus (3.113) may be shortly written as

$$\boldsymbol{v}_2^2 = \boldsymbol{J}_1^2 \boldsymbol{v}_1^1. \tag{3.114}$$

At this point, by virtue of the kineto-statics duality, the *force transformation* between two frames can be directly derived in the form

$$\boldsymbol{\gamma}_1^1 = \boldsymbol{J}_1^{2T} \boldsymbol{\gamma}_2^2 \tag{3.115}$$

which can be detailed into[18]

$$\begin{bmatrix} \boldsymbol{f}_1^1 \\ \boldsymbol{\mu}_1^1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_2^1 & \boldsymbol{O} \\ \boldsymbol{S}(\boldsymbol{r}_{12}^1) \boldsymbol{R}_2^1 & \boldsymbol{R}_2^1 \end{bmatrix} \begin{bmatrix} \boldsymbol{f}_2^2 \\ \boldsymbol{\mu}_2^2 \end{bmatrix}. \tag{3.116}$$

Finally, notice that the above analysis is instantaneous in that, if a coordinate frame varies with respect to the other, it is necessary to recompute the Jacobian of the transformation through the computation of the related rotation matrix of one frame with respect to the other.

3.8.3 Closed Chain

As discussed in Sect. 2.8.3, whenever the manipulator contains a closed chain, there is a functional relationship between the joint variables. In particular, the closed chain structure is transformed into a tree-structured open chain by virtually cutting the loop at a joint. It is worth choosing such a cut joint as one of the unactuated joints. Then, the constraints (2.59) or (2.60) should be solved for a reduced number of joint variables, corresponding to the DOFs of the chain. Therefore, it is reasonable to assume that at least such independent joints are actuated, while the others may or may not be actuated. Let $\boldsymbol{q}_o = [\, \boldsymbol{q}_a^T \quad \boldsymbol{q}_u^T \,]^T$ denote the vector of joint variables of the tree-structured open chain, where $\boldsymbol{q}_a$ and $\boldsymbol{q}_u$ are the vectors of *actuated* and *unactuated* joint variables, respectively. Assume that from the above constraints it is possible to determine a functional expression

$$\boldsymbol{q}_u = \boldsymbol{q}_u(\boldsymbol{q}_a). \tag{3.117}$$

Time differentiation of (3.117) gives the relationship between joint velocities in the form

$$\dot{\boldsymbol{q}}_o = \boldsymbol{\Upsilon} \dot{\boldsymbol{q}}_a \tag{3.118}$$

where

$$\boldsymbol{\Upsilon} = \begin{bmatrix} \boldsymbol{I} \\ \dfrac{\partial \boldsymbol{q}_u}{\partial \boldsymbol{q}_a} \end{bmatrix} \tag{3.119}$$

[18] The skew-symmetry property $\boldsymbol{S} + \boldsymbol{S}^T = \boldsymbol{O}$ is utilized.

is the transformation matrix between the two vectors of joint velocities, which in turn plays the role of a Jacobian.

At this point, according to an intuitive kineto-statics duality concept, it is possible to describe the transformation between the corresponding vectors of joint torques in the form

$$\boldsymbol{\tau}_a = \boldsymbol{\Upsilon}^T \boldsymbol{\tau}_o \tag{3.120}$$

where $\boldsymbol{\tau}_o = [\,\boldsymbol{\tau}_a^T \quad \boldsymbol{\tau}_u^T\,]^T$, with obvious meaning of the quantities.

Example 3.5

Consider the parallelogram arm of Sect. 2.9.2. On the assumption to actuate the two Joints $1'$ and $1''$ at the base, it is $\boldsymbol{q}_a = [\,\vartheta_{1'} \quad \vartheta_{1''}\,]^T$ and $\boldsymbol{q}_u = [\,\vartheta_{2'} \quad \vartheta_{3'}\,]^T$. Then, using (2.64), the transformation matrix in (3.119) is

$$\boldsymbol{\Upsilon} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Hence, in view of (3.120), the torque vector of the actuated joints is

$$\boldsymbol{\tau}_a = \begin{bmatrix} \tau_{1'} - \tau_{2'} + \tau_{3'} \\ \tau_{1''} + \tau_{2'} - \tau_{3'} \end{bmatrix} \tag{3.121}$$

while obviously $\boldsymbol{\tau}_u = [\,0 \quad 0\,]^T$ in agreement with the fact that both Joints $2'$ and $3'$ are unactuated.

3.9 Manipulability Ellipsoids

The differential kinematics equation in (3.45) and the statics equation in (3.111), together with the duality property, allow the definition of indices for the evaluation of manipulator performance. Such indices can be helpful both for mechanical manipulator design and for determining suitable manipulator postures to execute a given task in the current configuration.

First, it is desired to represent the attitude of a manipulator to arbitrarily change end-effector position and orientation. This capability is described in an effective manner by the *velocity manipulability ellipsoid*.

Consider the set of joint velocities of constant (unit) norm

$$\dot{\boldsymbol{q}}^T \dot{\boldsymbol{q}} = 1; \tag{3.122}$$

this equation describes the points on the surface of a sphere in the joint velocity space. It is desired to describe the operational space velocities that can

be generated by the given set of joint velocities, with the manipulator in a given posture. To this end, one can utilize the differential kinematics equation in (3.45) solved for the joint velocities; in the general case of a redundant manipulator ($r < n$) at a nonsingular configuration, the minimum-norm solution $\dot{\boldsymbol{q}} = \boldsymbol{J}^\dagger(\boldsymbol{q})\boldsymbol{v}_e$ can be considered which, substituted into (3.122), yields

$$\boldsymbol{v}_e^T \big(\boldsymbol{J}^{\dagger T}(\boldsymbol{q})\boldsymbol{J}^\dagger(\boldsymbol{q})\big)\boldsymbol{v}_e = 1.$$

Accounting for the expression of the pseudo-inverse of $\boldsymbol{J}$ in (3.52) gives

$$\boldsymbol{v}_e^T \big(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})\big)^{-1}\boldsymbol{v}_e = 1, \tag{3.123}$$

which is the equation of the points on the surface of an ellipsoid in the end-effector velocity space.

The choice of the minimum-norm solution rules out the presence of internal motions for the redundant structure. If the general solution (3.54) is used for $\dot{\boldsymbol{q}}$, the points satisfying (3.122) are mapped into points inside the ellipsoid whose surface is described by (3.123).

For a nonredundant manipulator, the differential kinematics solution (3.47) is used to derive (3.123); in this case the points on the surface of the sphere in the joint velocity space are mapped into points on the surface of the ellipsoid in the end-effector velocity space.

Along the direction of the major axis of the ellipsoid, the end-effector can move at large velocity, while along the direction of the minor axis small end-effector velocities are obtained. Further, the closer the ellipsoid is to a sphere — unit eccentricity — the better the end-effector can move isotropically along all directions of the operational space. Hence, it can be understood why this ellipsoid is an index characterizing manipulation ability of the structure in terms of velocities.

As can be recognized from (3.123), the shape and orientation of the ellipsoid are determined by the core of its quadratic form and then by the matrix $\boldsymbol{J}\boldsymbol{J}^T$ which is in general a function of the manipulator configuration. The directions of the principal axes of the ellipsoid are determined by the eigenvectors $\boldsymbol{u}_i$, for $i = 1, \ldots, r$, of the matrix $\boldsymbol{J}\boldsymbol{J}^T$, while the dimensions of the axes are given by the singular values of $\boldsymbol{J}$, $\sigma_i = \sqrt{\lambda_i(\boldsymbol{J}\boldsymbol{J}^T)}$, for $i = 1, \ldots, r$, where $\lambda_i(\boldsymbol{J}\boldsymbol{J}^T)$ denotes the generic eigenvalue of $\boldsymbol{J}\boldsymbol{J}^T$.

A global representative measure of manipulation ability can be obtained by considering the volume of the ellipsoid. This volume is proportional to the quantity

$$w(\boldsymbol{q}) = \sqrt{\det\big(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})\big)}$$

which is the *manipulability measure* already introduced in (3.56). In the case of a nonredundant manipulator ($r = n$), w reduces to

$$w(\boldsymbol{q}) = \big|\det\big(\boldsymbol{J}(\boldsymbol{q})\big)\big|. \tag{3.124}$$

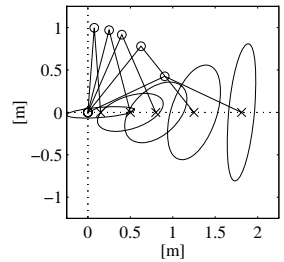

Fig. 3.23. Velocity manipulability ellipses for a two-link planar arm in different postures

It is easy to recognize that it is always $w > 0$, except for a manipulator at a singular configuration when $w = 0$. For this reason, this measure is usually adopted as a distance of the manipulator from singular configurations.

Example 3.6

Consider the two-link planar arm. From the expression in (3.41), the manipulability measure is in this case

$$w = |\det(\boldsymbol{J})| = a_1 a_2 |s_2|.$$

Therefore, as a function of the arm postures, the manipulability is maximum for $\vartheta_2 = \pm\pi/2$. On the other hand, for a given constant reach $a_1 + a_2$, the structure offering the maximum manipulability, independently of ϑ_1 and ϑ_2, is the one with $a_1 = a_2$.

These results have a biomimetic interpretation in the human arm, if that is regarded as a two-link arm (arm + forearm). The condition $a_1 = a_2$ is satisfied with good approximation. Further, the elbow angle ϑ_2 is usually in the neighbourhood of $\pi/2$ in the execution of several tasks, such as that of writing. Hence, the human being tends to dispose the arm in the most dexterous configuration from a manipulability viewpoint.

Figure 3.23 illustrates the velocity manipulability ellipses for a certain number of postures with the tip along the horizontal axis and $a_1 = a_2 = 1$. It can be seen that when the arm is outstretched the ellipsoid is very thin along the vertical direction. Hence, one recovers the result anticipated in the study of singularities that the arm in this posture can generate tip velocities preferably along the vertical direction. In Fig. 3.24, moreover, the behaviour of the minimum and maximum singular values of the matrix $\boldsymbol{J}$ is illustrated as a function of tip position along axis x; it can be verified that the minimum singular value is null when the manipulator is at a singularity (retracted or outstretched).

Therefore, with reference to the postures, manipulability has a maximum for $\vartheta_2 = \pm\pi/2$. On the other hand, for a given total extension $a_1 + a_2$, the structure which, independently of ϑ_1 and ϑ_2, offers the largest manipulability is that with $a_1 = a_2$.

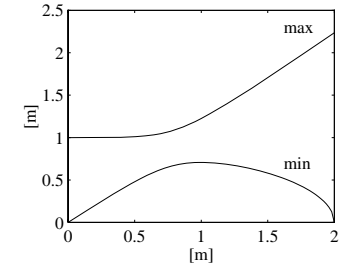

Fig. 3.24. Minimum and maximum singular values of $\boldsymbol{J}$ for a two-link planar arm as a function of the arm posture

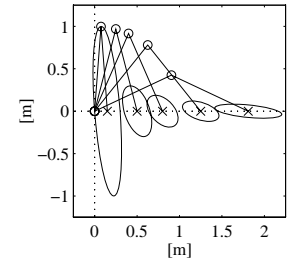

Fig. 3.25. Force manipulability ellipses for a two-link planar arm in different postures

The manipulability measure w has the advantage of being easy to compute, through the determinant of matrix $\boldsymbol{J}\boldsymbol{J}^T$. However, its numerical value does not constitute an absolute measure of the actual closeness of the manipulator to a singularity. It is enough to consider the above example and take two arms of identical structure, one with links of 1 m and the other with links of 1 cm. Two different values of manipulability are obtained which differ by four orders of magnitude. Hence, in that case it is convenient to consider only $|s_2|$ — eventually $|\vartheta_2|$ — as the manipulability measure. In more general cases when it is not easy to find a simple, meaningful index, one can consider the ratio between the minimum and maximum singular values of the Jacobian σ_r/σ_1 which is equivalent to the inverse of the condition number of matrix $\boldsymbol{J}$. This ratio gives not only a measure of the distance from a singularity ($\sigma_r = 0$), but also a direct measure of eccentricity of the ellipsoid. The disadvantage in utilizing this index is its computational complexity; it is practically impossible to compute it in symbolic form, i.e., as a function of the joint configuration, except for matrices of reduced dimension.

On the basis of the existing duality between differential kinematics and statics, it is possible to describe the manipulability of a structure not only

with reference to velocities, but also with reference to forces. To be specific, one can consider the sphere in the space of joint torques

$$\boldsymbol{\tau}^T\boldsymbol{\tau} = 1 \tag{3.125}$$

which, accounting for (3.111), is mapped into the ellipsoid in the space of end-effector forces

$$\boldsymbol{\gamma}_e^T\big(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})\big)\boldsymbol{\gamma}_e = 1 \tag{3.126}$$

which is defined as the *force manipulability ellipsoid*. This ellipsoid characterizes the end-effector forces that can be generated with the given set of joint torques, with the manipulator in a given posture.

As can be easily recognized from (3.126), the core of the quadratic form is constituted by the inverse of the matrix core of the velocity ellipsoid in (3.123). This feature leads to the notable result that the principal axes of the force manipulability ellipsoid coincide with the principal axes of the velocity manipulability ellipsoid, while the dimensions of the respective axes are in inverse proportion. Therefore, according to the concept of force/velocity duality, a direction along which good velocity manipulability is obtained is a direction along which poor force manipulability is obtained, and vice versa.

In Fig. 3.25, the manipulability ellipses for the same postures as those of the example in Fig. 3.23 are illustrated. A comparison of the shape and orientation of the ellipses confirms the force/velocity duality effect on the manipulability along different directions.

It is worth pointing out that these manipulability ellipsoids can be represented geometrically in all cases of an operational space of dimension at most 3. Therefore, if it is desired to analyze manipulability in a space of greater dimension, it is worth separating the components of linear velocity (force) from those of angular velocity (moment), also avoiding problems due to non-homogeneous dimensions of the relevant quantities (e.g., m/s vs rad/s). For instance, for a manipulator with a spherical wrist, the manipulability analysis is naturally prone to a decoupling between arm and wrist.

An effective interpretation of the above results can be achieved by regarding the manipulator as a *mechanical transformer* of velocities and forces from the joint space to the operational space. Conservation of energy dictates that an amplification in the velocity transformation is necessarily accompanied by a reduction in the force transformation, and vice versa. The transformation ratio along a given direction is determined by the intersection of the vector along that direction with the surface of the ellipsoid. Once a unit vector $\boldsymbol{u}$ along a direction has been assigned, it is possible to compute the transformation ratio for the force manipulability ellipsoid as

$$\alpha(\boldsymbol{q}) = \left(\boldsymbol{u}^T\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{u}\right)^{-1/2} \tag{3.127}$$

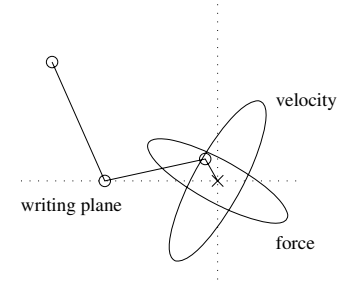

Fig. 3.26. Velocity and force manipulability ellipses for a 3-link planar arm in a typical configuration for a task of controlling force and velocity

and for the velocity manipulability ellipsoid as

$$\beta(\boldsymbol{q}) = \left(\boldsymbol{u}^T\big(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})\big)^{-1}\boldsymbol{u}\right)^{-1/2}. \tag{3.128}$$

The manipulability ellipsoids can be conveniently utilized not only for analyzing manipulability of the structure along different directions of the operational space, but also for determining compatibility of the structure to execute a task assigned along a direction. To this end, it is useful to distinguish between actuation tasks and control tasks of velocity and force. In terms of the relative ellipsoid, the task of actuating a velocity (force) requires preferably a large transformation ratio along the task direction, since for a given set of joint velocities (forces) at the joints it is possible to generate a large velocity (force) at the end-effector. On the other hand, for a control task it is important to have a small transformation ratio so as to gain good sensitivity to errors that may occur along the given direction.

Revisiting once again the duality between velocity manipulability ellipsoid and force manipulability ellipsoid, it can be found that an optimal direction to actuate a velocity is also an optimal direction to control a force. Analogously, a good direction to actuate a force is also a good direction to control a velocity.

To have a tangible example of the above concept, consider the typical task of writing on a horizontal surface for the human arm; this time, the arm is regarded as a 3-link planar arm: arm + forearm + hand. Restricting the analysis to a two-dimensional task space (the direction vertical to the surface and the direction of the line of writing), one has to achieve fine control of the vertical force (the pressure of the pen on the paper) and of the horizontal velocity (to write in good calligraphy). As a consequence, the force manipulability ellipse tends to be oriented horizontally for correct task execution. Correspondingly, the velocity manipulability ellipse tends to be oriented vertically in perfect agreement with the task requirement. In this case, from Fig. 3.26 the typical configuration of the human arm when writing can be recognized.

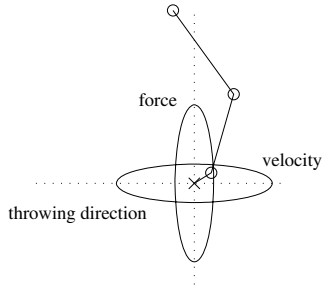

Fig. 3.27. Velocity and force manipulability ellipses for a 3-link planar arm in a typical configuration for a task of actuating force and velocity

An opposite example to the previous one is that of the human arm when throwing a weight in the horizontal direction. In fact, now it is necessary to actuate a large vertical force (to sustain the weight) and a large horizontal velocity (to throw the load for a considerable distance). Unlike the above, the force (velocity) manipulability ellipse tends to be oriented vertically (horizontally) to successfully execute the task. The relative configuration in Fig. 3.27 is representative of the typical attitude of the human arm when, for instance, releasing the ball in a bowling game.

In the above two examples, it is worth pointing out that the presence of a two-dimensional operational space is certainly advantageous to try reconfiguring the structure in the best configuration compatible with the given task. In fact, the transformation ratios defined in (3.127) and (3.128) are scalar functions of the manipulator configurations that can be optimized locally according to the technique for exploiting redundant DOFs previously illustrated.

Bibliography

The concept of geometric Jacobian was originally introduced in [240] and the problem of its computationally efficient determination is considered in [173]. The concept of analytical Jacobian is presented in [114] with reference to operational space control.

Inverse differential kinematics dates back to [240] under the name of resolved rate control. The use of the Jacobian pseudo-inverse is due to [118]. The adoption of the damped least-squares inverse has been independently proposed by [161] and [238]; a tutorial on the topic is [42]. The inverse kinematics algorithm based on the Jacobian transpose has been originally proposed in [198, 16]. Further details about the orientation error are found in [142, 250, 132, 41].

The utilization of the joint velocities in the null space of the Jacobian for redundancy resolution is proposed in [129] and further refined in [147] regarding the choice of the objective functions. The approach based on task priority is presented in [163]; other approaches based on the concept of augmented task space are presented in [14, 69, 199, 203, 194, 37]. For global redundancy resolutions see [162]. A complete treatment of redundant manipulators can be found in [160] while a tutorial is [206].

The extension of inverse kinematics to the second order has been proposed in [207], while the symbolic differentiation of the solutions in terms of joint velocities to obtain stable acceleration solutions can be found in [208]. Further details about redundancy resolution are in [59].

The concepts of kineto-statics duality are discussed in [191]. The manipulability ellipsoids are proposed in [245, 248] and employed in [44] for posture dexterity analysis with regard to manipulation tasks.

Problems

3.1. Prove (3.11).

3.2. Compute the Jacobian of the cylindrical arm in Fig. 2.35.

3.3. Compute the Jacobian of the SCARA manipulator in Fig. 2.36.

3.4. Find the singularities of the 3-link planar arm in Fig. 2.20.

3.5. Find the singularities of the spherical arm in Fig. 2.22.

3.6. Find the singularities of the cylindrical arm in Fig. 2.35.

3.7. Find the singularities of the SCARA manipulator in Fig. 2.36.

3.8. Show that the manipulability measure defined in (3.56) is given by the product of the singular values of the Jacobian matrix.

3.9. For the 3-link planar arm in Fig. 2.20, find an expression of the distance of the arm from a circular obstacle of given radius and coordinates.

3.10. Find the solution to the differential kinematics equation with the damped least-square inverse in (3.59).

3.11. Prove (3.64) in an alternative way, i.e., by computing $\boldsymbol{S}(\boldsymbol{\omega}_e)$ as in (3.6) starting from $\boldsymbol{R}(\boldsymbol{\phi})$ in (2.18).

3.12. With reference to (3.64), find the transformation matrix $\boldsymbol{T}(\boldsymbol{\phi}_e)$ in the case of RPY angles.

3.13. With reference to (3.64), find the triplet of Euler angles for which $\boldsymbol{T}(\boldsymbol{0}) = \boldsymbol{I}$.

3.14. Show how the inverse kinematics scheme of Fig. 3.11 can be simplified in the case of a manipulator having a spherical wrist.

3.15. Find an expression of the upper bound on the norm of $\boldsymbol{e}$ for the solution (3.76) in the case $\dot{\boldsymbol{x}}_d \neq \boldsymbol{0}$.

3.16. Prove (3.81).

3.17. Prove (3.86), (3.87).

3.18. Prove that the equation relating the angular velocity to the time derivative of the quaternion is given by

$$\boldsymbol{\omega} = 2\boldsymbol{S}(\boldsymbol{\epsilon})\dot{\boldsymbol{\epsilon}} + 2\eta\dot{\boldsymbol{\epsilon}} - 2\dot{\eta}\boldsymbol{\epsilon}.$$

[*Hint*: Start by showing that (2.33) can be rewritten as $\boldsymbol{R}(\eta, \boldsymbol{\epsilon}) = (2\eta^2 - 1)\boldsymbol{I} + 2\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T + 2\eta\boldsymbol{S}(\boldsymbol{\epsilon})$].

3.19. Prove (3.94), (3.95).

3.20. Prove that the time derivative of the Lyapunov function in (3.96) is given by (3.97).

3.21. Consider the 3-link planar arm in Fig. 2.20, whose link lengths are respectively 0.5 m, 0.3 m, 0.3 m. Perform a computer implementation of the inverse kinematics algorithm using the Jacobian pseudo-inverse along the operational space path given by a straight line connecting the points of coordinates $(0.8, 0.2)$ m and $(0.8, -0.2)$ m. Add a constraint aimed at avoiding link collision with a circular object located at $\emptyset = [\,0.3 \quad 0\,]^T$ m of radius 0.1 m. The initial arm configuration is chosen so that $\boldsymbol{p}_e(0) = \boldsymbol{p}_d(0)$. The final time is 2 s. Use sinusoidal motion timing laws. Adopt the Euler numerical integration scheme (3.48) with an integration time $\Delta t = 1$ ms.

3.22. Consider the SCARA manipulator in Fig. 2.36, whose links both have a length of 0.5 m and are located at a height of 1 m from the supporting plane. Perform a computer implementation of the inverse kinematics algorithms with both Jacobian inverse and Jacobian transpose along the operational space path whose position is given by a straight line connecting the points of coordinates $(0.7, 0, 0)$ m and $(0, 0.8, 0.5)$ m, and whose orientation is given by a rotation from 0 rad to $\pi/2$ rad. The initial arm configuration is chosen so that $\boldsymbol{x}_e(0) = \boldsymbol{x}_d(0)$. The final time is 2 s. Use sinusoidal motion timing laws. Adopt the Euler numerical integration scheme (3.48) with an integration time $\Delta t = 1$ ms.

3.23. Prove that the directions of the principal axes of the force and velocity manipulability ellipsoids coincide while their dimensions are in inverse proportion.

4

Trajectory Planning

For the execution of a specific robot task, it is worth considering the main features of motion planning algorithms. The goal of *trajectory planning* is to generate the reference inputs to the motion control system which ensures that the manipulator executes the planned trajectories. The user typically specifies a number of parameters to describe the desired trajectory. Planning consists of generating a time sequence of the values attained by an interpolating function (typically a polynomial) of the desired trajectory. This chapter presents some techniques for trajectory generation, both in the case when the initial and final point of the path are assigned (*point-to-point motion*), and in the case when a finite sequence of points are assigned along the path (*motion through a sequence of points*). First, the problem of trajectory planning in the *joint space* is considered, and then the basic concepts of trajectory planning in the *operational space* are illustrated. The treatment of the motion planning problem for mobile robots is deferred to Chap. 12.

4.1 Path and Trajectory

The minimal requirement for a manipulator is the capability to move from an initial posture to a final assigned posture. The transition should be characterized by motion laws requiring the actuators to exert joint generalized forces which do not violate the saturation limits and do not excite the typically modelled resonant modes of the structure. It is then necessary to devise planning algorithms that generate suitably smooth trajectories.

In order to avoid confusion between terms often used as synonyms, the difference between a path and a trajectory is to be explained. A *path* denotes the locus of points in the joint space, or in the operational space, which the manipulator has to follow in the execution of the assigned motion; a path is then a pure geometric description of motion. On the other hand, a *trajectory* is a path on which a timing law is specified, for instance in terms of velocities and/or accelerations at each point.

In principle, it can be conceived that the inputs to a *trajectory planning* algorithm are the path description, the path constraints, and the constraints imposed by manipulator dynamics, whereas the outputs are the end-effector trajectories in terms of a time sequence of the values attained by position, velocity and acceleration.

A geometric path cannot be fully specified by the user for obvious complexity reasons. Typically, a reduced number of parameters is specified such as extremal points, possible intermediate points, and geometric primitives interpolating the points. Also, the motion timing law is not typically specified at each point of the geometric path, but rather it regards the total trajectory time, the constraints on the maximum velocities and accelerations, and eventually the assignment of velocity and acceleration at points of particular interest. On the basis of the above information, the trajectory planning algorithm generates a time sequence of variables that describe end-effector position and orientation over time in respect of the imposed constraints. Since the control action on the manipulator is carried out in the joint space, a suitable inverse kinematics algorithm is to be used to reconstruct the time sequence of joint variables corresponding to the above sequence in the operational space.

Trajectory planning in the operational space naturally allows the presence of path constraints to be accounted; these are due to regions of workspace which are forbidden to the manipulator, e.g., due to the presence of obstacles. In fact, such constraints are typically better described in the operational space, since their corresponding points in the joint space are difficult to compute.

With regard to motion in the neighbourhood of singular configurations and presence of redundant DOFs, trajectory planning in the operational space may involve problems difficult to solve. In such cases, it may be advisable to specify the path in the joint space, still in terms of a reduced number of parameters. Hence, a time sequence of joint variables has to be generated which satisfy the constraints imposed on the trajectory.

For the sake of clarity, in the following, the case of joint space trajectory planning is treated first. The results will then be extended to the case of trajectories in the operational space.

4.2 Joint Space Trajectories

A manipulator motion is typically assigned in the operational space in terms of trajectory parameters such as the initial and final end-effector pose, possible intermediate poses, and travelling time along particular geometric paths. If it is desired to plan a trajectory in the *joint space*, the values of the joint variables have to be determined first from the end-effector position and orientation specified by the user. It is then necessary to resort to an inverse kinematics algorithm, if planning is done off-line, or to directly measure the above variables, if planning is done by the teaching-by-showing technique (see Chap. 6).

The planning algorithm generates a function $\boldsymbol{q}(t)$ interpolating the given vectors of joint variables at each point, in respect of the imposed constraints.

In general, a joint space trajectory planning algorithm is required to have the following features:

- the generated trajectories should be not very demanding from a computational viewpoint,
- the joint positions and velocities should be continuous functions of time (continuity of accelerations may be imposed, too),
- undesirable effects should be minimized, e.g., nonsmooth trajectories interpolating a sequence of points on a path.

At first, the case is examined when only the initial and final points on the path and the traveling time are specified (*point-to-point*); the results are then generalized to the case when also intermediate points along the path are specified (*motion through a sequence of points*). Without loss of generality, the single joint variable $q(t)$ is considered.

4.2.1 Point-to-Point Motion

In *point-to-point motion*, the manipulator has to move from an initial to a final joint configuration in a given time t_f. In this case, the actual end-effector path is of no concern. The algorithm should generate a trajectory which, in respect to the above general requirements, is also capable of optimizing some performance index when the joint is moved from one position to another.

A suggestion for choosing the motion primitive may stem from the analysis of an incremental motion problem. Let I be the moment of inertia of a rigid body about its rotation axis. It is required to take the angle q from an initial value q_i to a final value q_f in a time t_f. It is obvious that infinite solutions exist to this problem. Assumed that rotation is executed through a torque τ supplied by a motor, a solution can be found which minimizes the energy dissipated in the motor. This optimization problem can be formalized as follows. Having set $\dot{q} = \omega$, determine the solution to the differential equation

$$I\dot{\omega} = \tau$$

subject to the condition

$$\int_o^{t_f} \omega(t)dt = q_f - q_i$$

so as to minimize the performance index

$$\int_0^{t_f} \tau^2(t)dt.$$

It can be shown that the resulting solution is of the type

$$\omega(t) = at^2 + bt + c.$$

Even though the joint dynamics cannot be described in the above simple manner,[1] the choice of a third-order polynomial function to generate a joint trajectory represents a valid solution for the problem at issue.

Therefore, to determine a joint motion, the *cubic polynomial*

$$q(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0 \tag{4.1}$$

can be chosen, resulting into a parabolic velocity profile

$$\dot{q}(t) = 3a_3 t^2 + 2a_2 t + a_1$$

and a linear acceleration profile

$$\ddot{q}(t) = 6a_3 t + 2a_2.$$

Since four coefficients are available, it is possible to impose, besides the initial and final joint position values q_i and q_f, also the initial and final joint velocity values $\dot{q}_i$ and $\dot{q}_f$ which are usually set to zero. Determination of a specific trajectory is given by the solution to the following system of equations:

$$\begin{aligned} a_0 &= q_i \\ a_1 &= \dot{q}_i \\ a_3 t_f^3 + a_2 t_f^2 + a_1 t_f + a_0 &= q_f \\ 3a_3 t_f^2 + 2a_2 t_f + a_1 &= \dot{q}_f, \end{aligned}$$

that allows the computation of the coefficients of the polynomial in (4.1).[2] Figure 4.1 illustrates the timing law obtained with the following data: $q_i = 0$, $q_f = \pi$, $t_f = 1$, and $\dot{q}_i = \dot{q}_f = 0$. As anticipated, velocity has a parabolic profile, while acceleration has a linear profile with initial and final discontinuity.

If it is desired to assign also the initial and final values of acceleration, six constraints have to be satisfied and then a polynomial of at least *fifth* order is needed. The motion timing law for the generic joint is then given by

$$q(t) = a_5 t^5 + a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0, \tag{4.2}$$

whose coefficients can be computed, as for the previous case, by imposing the conditions for $t = 0$ and $t = t_f$ on the joint variable $q(t)$ and on its first two derivatives. With the choice (4.2), one obviously gives up minimizing the above performance index.

An alternative approach with timing laws of blended polynomial type is frequently adopted in industrial practice, which allows a direct verification

[1] In fact, recall that the moment of inertia about the joint axis is a function of manipulator configuration.

[2] Notice that it is possible to normalize the computation of the coefficients, so as to be independent both on the final time t_f and on the path length $|q_f - q_i|$.

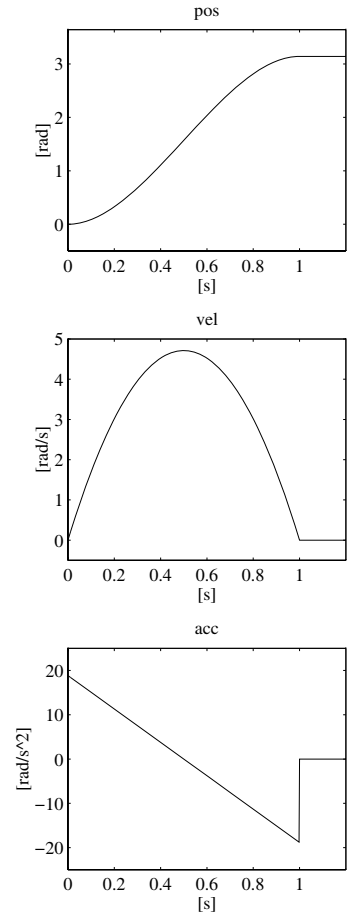

Fig. 4.1. Time history of position, velocity and acceleration with a cubic polynomial timing law

of whether the resulting velocities and accelerations can be supported by the physical mechanical manipulator.

In this case, a *trapezoidal velocity profile* is assigned, which imposes a constant acceleration in the start phase, a cruise velocity, and a constant deceleration in the arrival phase. The resulting trajectory is formed by a linear segment connected by two parabolic segments to the initial and final positions.

In the following, the problem is formulated by assuming that the final time of trajectory duration has been assigned. However, in industrial practice, the user is offered the option to specify the velocity percentage with respect to the maximum allowable velocity; this choice is aimed at avoiding occurrences when

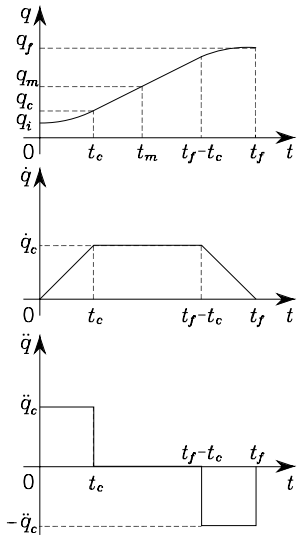

Fig. 4.2. Characterization of a timing law with trapezoidal velocity profile in terms of position, velocity and acceleration

the specification of a much too short motion duration would involve much too large values of velocities and/or accelerations, beyond those achievable by the manipulator.

As can be seen from the velocity profiles in Fig. 4.2, it is assumed that both initial and final velocities are null and the segments with constant accelerations have the same time duration; this implies an equal magnitude $\ddot{q}_c$ in the two segments. Notice also that the above choice leads to a symmetric trajectory with respect to the average point $q_m = (q_f + q_i)/2$ at $t_m = t_f/2$.

The trajectory has to satisfy some constraints to ensure the transition from q_i to q_f in a time t_f. The velocity at the end of the parabolic segment must be equal to the (constant) velocity of the linear segment, i.e.,

$$\ddot{q}_c t_c = \frac{q_m - q_c}{t_m - t_c} \tag{4.3}$$

where q_c is the value attained by the joint variable at the end of the parabolic segment at time t_c with constant acceleration $\ddot{q}_c$ (recall that $\dot{q}(0) = 0$). It is then

$$q_c = q_i + \frac{1}{2}\ddot{q}_c t_c^2. \tag{4.4}$$

Combining (4.3), (4.4) gives

$$\ddot{q}_c t_c^2 - \ddot{q}_c t_f t_c + q_f - q_i = 0. \tag{4.5}$$

Usually, $\ddot{q}_c$ is specified with the constraint that $\operatorname{sgn} \ddot{q}_c = \operatorname{sgn}(q_f - q_i)$; hence, for given t_f, q_i and q_f, the solution for t_c is computed from (4.5) as ($t_c \leq t_f/2$)

$$t_c = \frac{t_f}{2} - \frac{1}{2}\sqrt{\frac{t_f^2 \ddot{q}_c - 4(q_f - q_i)}{\ddot{q}_c}}. \tag{4.6}$$

Acceleration is then subject to the constraint

$$|\ddot{q}_c| \geq \frac{4|q_f - q_i|}{t_f^2}. \tag{4.7}$$

When the acceleration $\ddot{q}_c$ is chosen so as to satisfy (4.7) with the equality sign, the resulting trajectory does not feature the constant velocity segment any more and has only the acceleration and deceleration segments (*triangular* profile).

Given q_i, q_f and t_f, and thus also an average transition velocity, the constraint in (4.7) allows the imposition of a value of acceleration consistent with the trajectory. Then, t_c is computed from (4.6), and the following sequence of polynomials is generated:

$$q(t) = \begin{cases} q_i + \frac{1}{2}\ddot{q}_c t^2 & 0 \leq t \leq t_c \\ q_i + \ddot{q}_c t_c (t - t_c/2) & t_c < t \leq t_f - t_c \\ q_f - \frac{1}{2}\ddot{q}_c (t_f - t)^2 & t_f - t_c < t \leq t_f. \end{cases} \tag{4.8}$$

Figure 4.3 illustrates a representation of the motion timing law obtained by imposing the data: $q_i = 0$, $q_f = \pi$, $t_f = 1$, and $|\ddot{q}_c| = 6\pi$.

Specifying acceleration in the parabolic segment is not the only way to determine trajectories with trapezoidal velocity profile. Besides q_i, q_f and t_f, one can specify also the cruise velocity $\dot{q}_c$ which is subject to the constraint

$$\frac{|q_f - q_i|}{t_f} < |\dot{q}_c| \leq \frac{2|q_f - q_i|}{t_f}. \tag{4.9}$$

By recognizing that $\dot{q}_c = \ddot{q}_c t_c$, (4.5) allows the computation of t_c as

$$t_c = \frac{q_i - q_f + \dot{q}_c t_f}{\dot{q}_c}, \tag{4.10}$$

and thus the resulting acceleration is

$$\ddot{q}_c = \frac{\dot{q}_c^2}{q_i - q_f + \dot{q}_c t_f}. \tag{4.11}$$

The computed values of t_c and $\ddot{q}_c$ as in (4.10), (4.11) allow the generation of the sequence of polynomials expressed by (4.8).

The adoption of a trapezoidal velocity profile results in a worse performance index compared to the cubic polynomial. The decrease is, however, limited; the term $\int_0^{t_f} \tau^2 dt$ increases by 12.5% with respect to the optimal case.

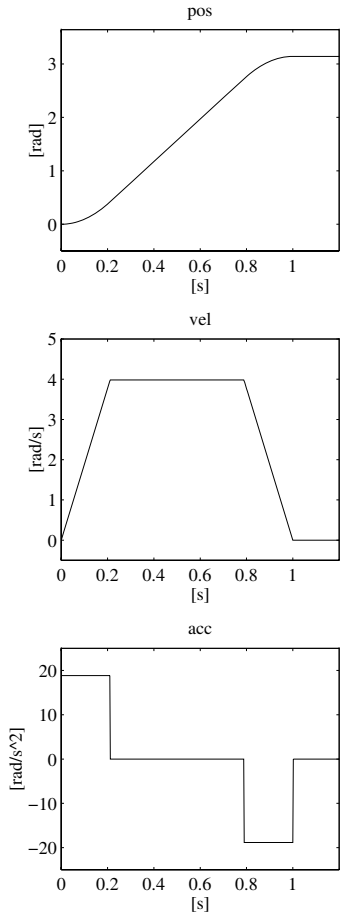

Fig. 4.3. Time history of position, velocity and acceleration with a trapezoidal velocity profile timing law

4.2.2 Motion Through a Sequence of Points

In several applications, the path is described in terms of a number of points greater than two. For instance, even for the simple point-to-point motion of a pick-and-place task, it may be worth assigning two intermediate points between the initial point and the final point; suitable positions can be set for lifting off and setting down the object, so that reduced velocities are obtained with respect to direct transfer of the object. For more complex applications, it may be convenient to assign a *sequence of points* so as to guarantee better monitoring on the executed trajectories; the points are to be specified more densely in those segments of the path where obstacles have to be avoided

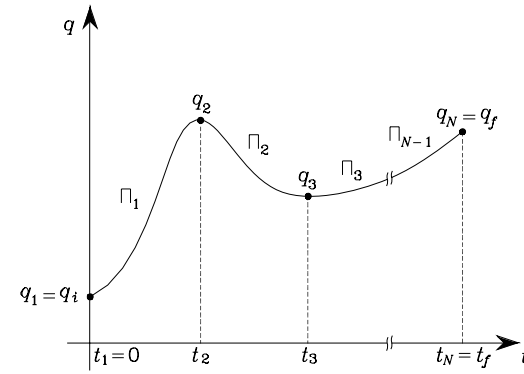

Fig. 4.4. Characterization of a trajectory on a given path obtained through interpolating polynomials

or a high path curvature is expected. It should not be forgotten that the corresponding joint variables have to be computed from the operational space poses.

Therefore, the problem is to generate a trajectory when N points, termed *path points*, are specified and have to be reached by the manipulator at certain instants of time. For each joint variable there are N constraints, and then one might want to use an $(N-1)$-order polynomial. This choice, however, has the following disadvantages:

- It is not possible to assign the initial and final velocities.
- As the order of a polynomial increases, its oscillatory behaviour increases, and this may lead to trajectories which are not natural for the manipulator.
- Numerical accuracy for computation of polynomial coefficients decreases as order increases.
- The resulting system of constraint equations is heavy to solve.
- Polynomial coefficients depend on all the assigned points; thus, if it is desired to change a point, all of them have to be recomputed.

These drawbacks can be overcome if a suitable number of low-order *interpolating polynomials*, continuous at the path points, are considered in place of a single high-order polynomial.

According to the previous section, the interpolating polynomial of lowest order is the *cubic polynomial*, since it allows the imposition of continuity of velocities at the path points. With reference to the single joint variable, a function $q(t)$ is sought, formed by a sequence of $N-1$ cubic polynomials $\Pi_k(t)$, for $k = 1, \ldots, N-1$, continuous with continuous first derivatives. The function $q(t)$ attains the values q_k for $t = t_k$ $(k = 1, \ldots, N)$, and $q_1 = q_i$, $t_1 = 0$, $q_N = q_f$, $t_N = t_f$; the q_k's represent the path points describing

the desired trajectory at $t = t_k$ (Fig. 4.4). The following situations can be considered:

- Arbitrary values of $\dot{q}(t)$ are imposed at the path points.
- The values of $\dot{q}(t)$ at the path points are assigned according to a certain criterion.
- The acceleration $\ddot{q}(t)$ has to be continuous at the path points.

To simplify the problem, it is also possible to find interpolating polynomials of order less than three which determine trajectories passing nearby the path points at the given instants of time.

Interpolating polynomials with imposed velocities at path points

This solution requires the user to be able to specify the desired velocity at each path point; the solution does not possess any novelty with respect to the above concepts.

The system of equations allowing computation of the coefficients of the $N-1$ cubic polynomials interpolating the N path points is obtained by imposing the following conditions on the generic polynomial $\Pi_k(t)$ interpolating q_k and q_{k+1}, for $k = 1, \ldots, N-1$:

$$\begin{aligned} \Pi_k(t_k) &= q_k \\ \Pi_k(t_{k+1}) &= q_{k+1} \\ \dot{\Pi}_k(t_k) &= \dot{q}_k \\ \dot{\Pi}_k(t_{k+1}) &= \dot{q}_{k+1}. \end{aligned}$$

The result is $N-1$ systems of four equations in the four unknown coefficients of the generic polynomial; these can be solved one independently of the other. The initial and final velocities of the trajectory are typically set to zero ($\dot{q}_1 = \dot{q}_N = 0$) and continuity of velocity at the path points is ensured by setting

$$\dot{\Pi}_k(t_{k+1}) = \dot{\Pi}_{k+1}(t_{k+1})$$

for $k = 1, \ldots, N-2$.

Figure 4.5 illustrates the time history of position, velocity and acceleration obtained with the data: $q_1 = 0$, $q_2 = 2\pi$, $q_3 = \pi/2$, $q_4 = \pi$, $t_1 = 0$, $t_2 = 2$, $t_3 = 3$, $t_4 = 5$, $\dot{q}_1 = 0$, $\dot{q}_2 = \pi$, $\dot{q}_3 = -\pi$, $\dot{q}_4 = 0$. Notice the resulting discontinuity on the acceleration, since only continuity of velocity is guaranteed.

Interpolating polynomials with computed velocities at path points

In this case, the joint velocity at a path point has to be computed according to a certain criterion. By interpolating the path points with linear segments, the relative velocities can be computed according to the following rules:

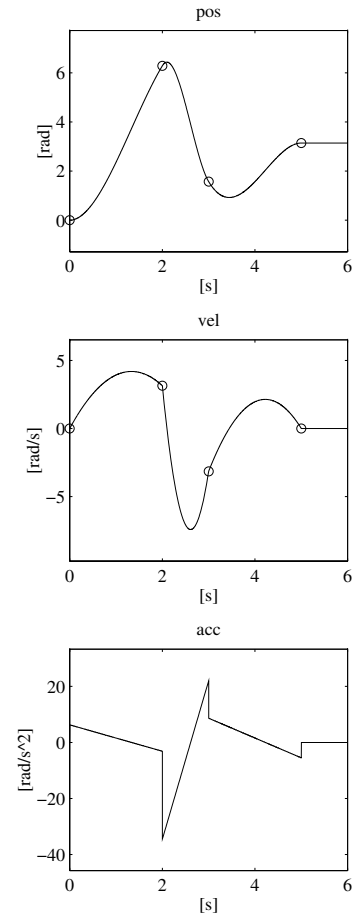

Fig. 4.5. Time history of position, velocity and acceleration with a timing law of interpolating polynomials with velocity constraints at path points

$$\begin{aligned} \dot{q}_1 &= 0 \\ \dot{q}_k &= \begin{cases} 0 & \operatorname{sgn}(v_k) \neq \operatorname{sgn}(v_{k+1}) \\ \frac{1}{2}(v_k + v_{k+1}) & \operatorname{sgn}(v_k) = \operatorname{sgn}(v_{k+1}) \end{cases} \\ \dot{q}_N &= 0, \end{aligned} \tag{4.12}$$

where $v_k = (q_k - q_{k-1})/(t_k - t_{k-1})$ gives the slope of the segment in the time interval $[t_{k-1}, t_k]$. With the above settings, the determination of the interpolating polynomials is reduced to the previous case.

Figure 4.6 illustrates the time history of position, velocity and acceleration obtained with the following data: $q_1 = 0$, $q_2 = 2\pi$, $q_3 = \pi/2$, $q_4 = \pi$, $t_1 = 0$, $t_2 = 2$, $t_3 = 3$, $t_4 = 5$, $\dot{q}_1 = 0$, $\dot{q}_4 = 0$. It is easy to recognize that the imposed

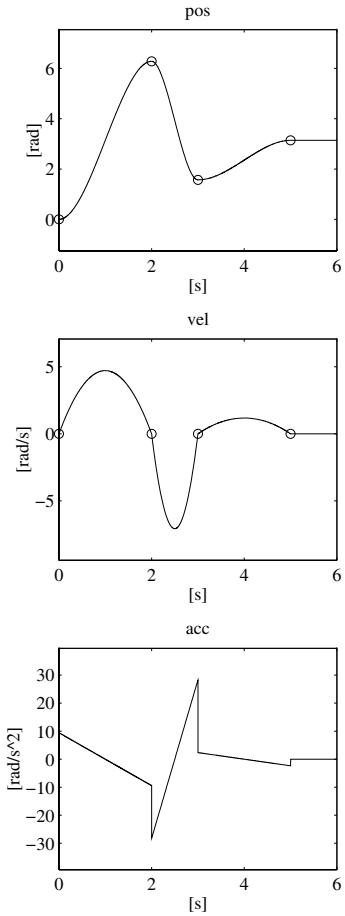

Fig. 4.6. Time history of position, velocity and acceleration with a timing law of interpolating polynomials with computed velocities at path points

sequence of path points leads to having zero velocity at the intermediate points.

Interpolating polynomials with continuous accelerations at path points (splines)

Both the above two solutions do not ensure continuity of accelerations at the path points. Given a sequence of N path points, the acceleration is also continuous at each t_k if four constraints are imposed, namely, two position constraints for each of the adjacent cubics and two constraints guaranteeing

continuity of velocity and acceleration. The following equations have then to be satisfied:

$$\begin{aligned}
&\Pi_{k-1}(t_k) = q_k \\
&\Pi_{k-1}(t_k) = \Pi_k(t_k) \\
&\dot{\Pi}_{k-1}(t_k) = \dot{\Pi}_k(t_k) \\
&\ddot{\Pi}_{k-1}(t_k) = \ddot{\Pi}_k(t_k).
\end{aligned}$$

The resulting system for the N path points, including the initial and final points, cannot be solved. In fact, it is formed by $4(N-2)$ equations for the intermediate points and 6 equations for the extremal points; the position constraints for the polynomials $\Pi_0(t_1) = q_i$ and $\Pi_N(t_f) = q_f$ have to be excluded since they are not defined. Also, $\dot{\Pi}_0(t_1)$, $\ddot{\Pi}_0(t_1)$, $\dot{\Pi}_N(t_f)$, $\ddot{\Pi}_N(t_f)$ do not have to be counted as polynomials since they are just the imposed values of initial and final velocities and accelerations. In summary, one has $4N-2$ equations in $4(N-1)$ unknowns.

The system can be solved only if one eliminates the two equations which allow the arbitrary assignment of the initial and final acceleration values. Fourth-order polynomials should be used to include this possibility for the first and last segment.

On the other hand, if only third-order polynomials are to be used, the following deception can be operated. Two *virtual points* are introduced for which continuity constraints on position, velocity and acceleration can be imposed, without specifying the actual positions, though. It is worth remarking that the effective location of these points is irrelevant, since their position constraints regard continuity only. Hence, the introduction of two virtual points implies the determination of $N+1$ cubic polynomials.

Consider $N+2$ time instants t_k, where t_2 and t_{N+1} conventionally refer to the virtual points. The system of equations for determining the $N+1$ cubic polynomials can be found by taking the $4(N-2)$ equations:

$$\Pi_{k-1}(t_k) = q_k \tag{4.13}$$
$$\Pi_{k-1}(t_k) = \Pi_k(t_k) \tag{4.14}$$
$$\dot{\Pi}_{k-1}(t_k) = \dot{\Pi}_k(t_k) \tag{4.15}$$
$$\ddot{\Pi}_{k-1}(t_k) = \ddot{\Pi}_k(t_k) \tag{4.16}$$

for $k = 3, \ldots, N$, written for the $N-2$ intermediate path points, the 6 equations:

$$\Pi_1(t_1) = q_i \tag{4.17}$$
$$\dot{\Pi}_1(t_1) = \dot{q}_i \tag{4.18}$$
$$\ddot{\Pi}_1(t_1) = \ddot{q}_i, \tag{4.19}$$

$$\Pi_{N+1}(t_{N+2}) = q_f \tag{4.20}$$

$$\dot{\Pi}_{N+1}(t_{N+2}) = \dot{q}_f \tag{4.21}$$
$$\ddot{\Pi}_{N+1}(t_{N+2}) = \ddot{q}_f \tag{4.22}$$

written for the initial and final points, and the 6 equations:

$$\Pi_{k-1}(t_k) = \Pi_k(t_k) \tag{4.23}$$
$$\dot{\Pi}_{k-1}(t_k) = \dot{\Pi}_k(t_k) \tag{4.24}$$
$$\ddot{\Pi}_{k-1}(t_k) = \ddot{\Pi}_k(t_k) \tag{4.25}$$

for $k = 2, N+1$, written for the two virtual points. The resulting system has $4(N+1)$ equations in $4(N+1)$ unknowns, that are the coefficients of the $N+1$ cubic polynomials.

The solution to the system is computationally demanding, even for low values of N. Nonetheless, the problem can be cast in a suitable form so as to solve the resulting system of equations with a computationally efficient algorithm. Since the generic polynomial $\Pi_k(t)$ is a cubic, its second derivative must be a linear function of time which then can be written as

$$\ddot{\Pi}_k(t) = \frac{\ddot{\Pi}_k(t_k)}{\Delta t_k}(t_{k+1} - t) + \frac{\ddot{\Pi}_k(t_{k+1})}{\Delta t_k}(t - t_k) \qquad k = 1, \ldots, N+1, \tag{4.26}$$

where $\Delta t_k = t_{k+1} - t_k$ indicates the time interval to reach q_{k+1} from q_k. By integrating (4.26) twice over time, the generic polynomial can be written as

$$\begin{aligned}\Pi_k(t) = {} & \frac{\ddot{\Pi}_k(t_k)}{6\Delta t_k}(t_{k+1} - t)^3 + \frac{\ddot{\Pi}_k(t_{k+1})}{6\Delta t_k}(t - t_k)^3 \\ & + \left(\frac{\Pi_k(t_{k+1})}{\Delta t_k} - \frac{\Delta t_k \ddot{\Pi}_k(t_{k+1})}{6}\right)(t - t_k) \\ & + \left(\frac{\Pi_k(t_k)}{\Delta t_k} - \frac{\Delta t_k \ddot{\Pi}_k(t_k)}{6}\right)(t_{k+1} - t) \qquad k = 1, \ldots, N+1,\end{aligned} \tag{4.27}$$

which depends on the 4 unknowns: $\Pi_k(t_k)$, $\Pi_k(t_{k+1})$, $\ddot{\Pi}_k(t_k)$, $\ddot{\Pi}_k(t_{k+1})$.

Notice that the N variables q_k for $k \neq 2, N+1$ are given via (4.13), while continuity is imposed for q_2 and q_{N+1} via (4.23). By using (4.14), (4.17), (4.20), the unknowns in the $N+1$ equations in (4.27) reduce to $2(N+2)$. By observing that the equations in (4.18), (4.21) depend on q_2 and q_{N+1}, and that $\dot{q}_i$ and $\dot{q}_f$ are given, q_2 and q_{N+1} can be computed as a function of $\ddot{\Pi}_1(t_1)$ and $\ddot{\Pi}_{N+1}(t_{N+2})$, respectively. Thus, a number of $2(N+1)$ unknowns are left.

By accounting for (4.16), (4.25), and noticing that in ((4.19), (4.22) $\ddot{q}_i$ and $\ddot{q}_f$ are given, the unknowns reduce to N.

At this point, (4.15), (4.24) can be utilized to write the system of N equations in N unknowns:

$$\dot{\Pi}_1(t_2) = \dot{\Pi}_2(t_2)$$

$$\vdots$$
$$\dot{\Pi}_N(t_{N+1}) = \dot{\Pi}_{N+1}(t_{N+1}).$$

Time-differentiation of (4.27) gives both $\dot{\Pi}_k(t_{k+1})$ and $\dot{\Pi}_{k+1}(t_{k+1})$ for $k = 1, \ldots, N$, and thus it is possible to write a system of linear equations of the kind

$$\boldsymbol{A}\,[\,\ddot{\Pi}_2(t_2) \quad \ldots \quad \ddot{\Pi}_{N+1}(t_{N+1})\,]^T = \boldsymbol{b} \tag{4.28}$$

which presents a vector $\boldsymbol{b}$ of known terms and a nonsingular coefficient matrix $\boldsymbol{A}$; the solution to this system always exists and is unique. It can be shown that the matrix $\boldsymbol{A}$ has a tridiagonal band structure of the type

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & 0 & 0 \\ a_{21} & a_{22} & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & a_{N-1,N-1} & a_{N-1,N} \\ 0 & 0 & \ldots & a_{N,N-1} & a_{NN} \end{bmatrix},$$

which simplifies the solution to the system (see Problem 4.4). This matrix is the same for all joints, since it depends only on the time intervals Δt_k specified.

An efficient solution algorithm exists for the above system which is given by a *forward* computation followed by a *backward* computation. From the first equation, $\ddot{\Pi}_2(t_2)$ can be computed as a function of $\ddot{\Pi}_3(t_3)$ and then substituted in the second equation, which then becomes an equation in the unknowns $\ddot{\Pi}_3(t_3)$ and $\ddot{\Pi}_4(t_4)$. This is carried out forward by transforming all the equations in equations with two unknowns, except the last one which will have $\ddot{\Pi}_{N+1}(t_{N+1})$ only as unknown. At this point, all the unknowns can be determined step by step through a backward computation.

The above sequence of cubic polynomials is termed *spline* to indicate smooth functions that interpolate a sequence of given points ensuring continuity of the function and its derivatives.

Figure 4.7 illustrates the time history of position, velocity and acceleration obtained with the data: $q_1 = 0$, $q_3 = 2\pi$, $q_4 = \pi/2$, $q_6 = \pi$, $t_1 = 0$, $t_3 = 2$, $t_4 = 3$, $t_6 = 5$, $\dot{q}_1 = 0$, $\dot{q}_6 = 0$. Two different pairs of virtual points were considered at the time instants: $t_2 = 0.5$, $t_5 = 4.5$ (solid line in the figure), and $t_2 = 1.5$, $t_5 = 3.5$ (dashed line in the figure), respectively. Notice the parabolic velocity profile and the linear acceleration profile. Further, for the second pair, larger values of acceleration are obtained, since the relative time instants are closer to those of the two intermediate points.

Interpolating linear polynomials with parabolic blends

A simplification in trajectory planning can be achieved as follows. Consider the case when it is desired to interpolate N path points $q_1, \ldots, q_N$ at time

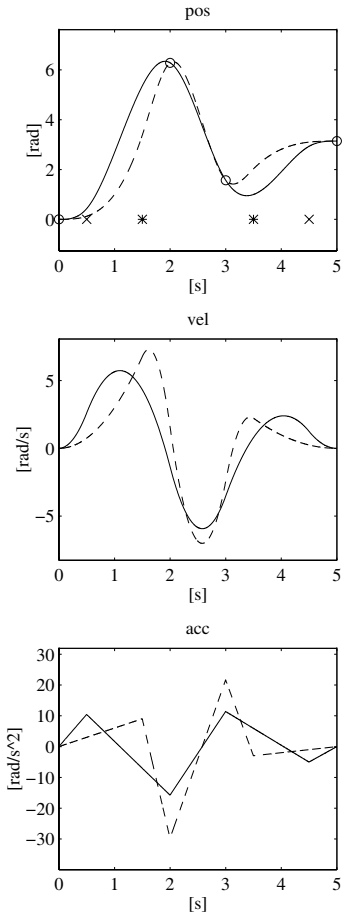

Fig. 4.7. Time history of position, velocity and acceleration with a timing law of cubic splines for two different pairs of virtual points

instants $t_1, \ldots, t_N$ with linear segments. To avoid discontinuity problems on the first derivative at the time instants t_k, the function $q(t)$ must have a parabolic profile (*blend*) around t_k; as a consequence, the entire trajectory is composed of a sequence of linear and quadratic polynomials, which in turn implies that a discontinuity on $\ddot{q}(t)$ is tolerated.

Then let $\Delta t_k = t_{k+1} - t_k$ be the time distance between q_k and q_{k+1}, and $\Delta t_{k,k+1}$ be the time interval during which the trajectory interpolating q_k and q_{k+1} is a linear function of time. Also let $\dot{q}_{k,k+1}$ be the constant velocity and $\ddot{q}_k$ be the acceleration in the parabolic blend whose duration is $\Delta t'_k$. The resulting trajectory is illustrated in Fig. 4.8. The values of q_k, Δt_k, and $\Delta t'_k$

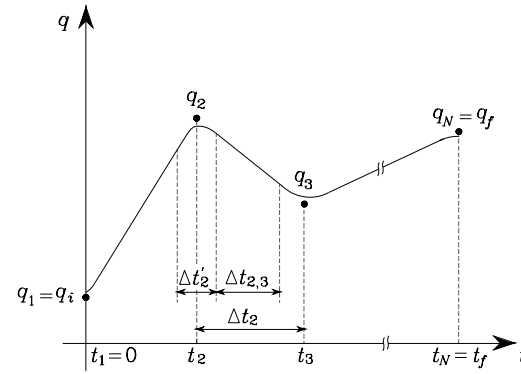

Fig. 4.8. Characterization of a trajectory with interpolating linear polynomials with parabolic blends

are assumed to be given. Velocity and acceleration for the intermediate points are computed as

$$\dot{q}_{k-1,k} = \frac{q_k - q_{k-1}}{\Delta t_{k-1}} \tag{4.29}$$

$$\ddot{q}_k = \frac{\dot{q}_{k,k+1} - \dot{q}_{k-1,k}}{\Delta t'_k}; \tag{4.30}$$

these equations are straightforward.

The first and last segments deserve special care. In fact, if it is desired to maintain the coincidence of the trajectory with the first and last segments, at least for a portion of time, the resulting trajectory has a longer duration given by $t_N - t_1 + (\Delta t'_1 + \Delta t'_N)/2$, where $\dot{q}_{0,1} = \dot{q}_{N,N+1} = 0$ has been imposed for computing initial and final accelerations.

Notice that $q(t)$ reaches none of the path points q_k but passes nearby (Fig. 4.8). In this situation, the path points are more appropriately termed *via points*; the larger the blending acceleration, the closer the passage to a via point.

On the basis of the given q_k, Δt_k and $\Delta t'_k$, the values of $\dot{q}_{k-1,k}$ and $\ddot{q}_k$ are computed via (4.29), (4.30) and a sequence of linear polynomials with parabolic blends is generated. Their expressions as a function of time are not derived here to avoid further loading of the analytic presentation.

Figure 4.9 illustrates the time history of position, velocity and acceleration obtained with the data: $q_1 = 0$, $q_2 = 2\pi$, $q_3 = \pi/2$, $q_4 = \pi$, $t_1 = 0$, $t_2 = 2$, $t_3 = 3$, $t_4 = 5$, $\dot{q}_1 = 0$, $\dot{q}_4 = 0$. Two different values for the blend times have been considered: $\Delta t'_k = 0.2$ (solid line in the figure) and $\Delta t'_k = 0.6$ (dashed line in the figure), for $k = 1, \ldots, 4$, respectively. Notice that in the first case the passage of $q(t)$ is closer to the via points, though at the expense of higher acceleration values.

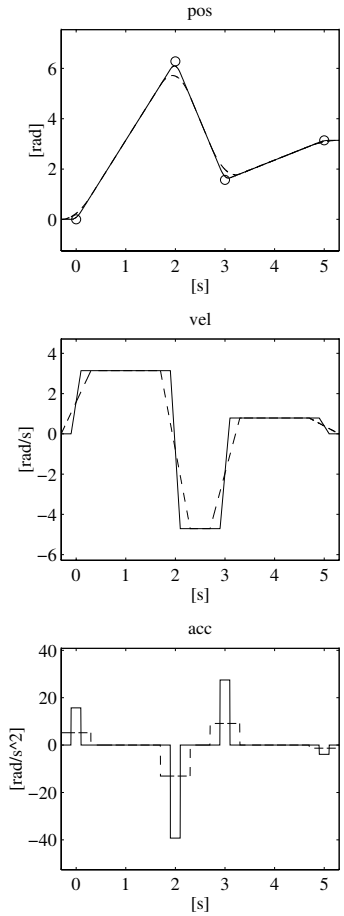

Fig. 4.9. Time history of position, velocity and acceleration with a timing law of interpolating linear polynomials with parabolic blends

The technique presented above turns out to be an application of the trapezoidal velocity profile law to the interpolation problem. If one gives up a trajectory passing near a via point at a prescribed instant of time, the use of trapezoidal velocity profiles allows the development of a trajectory planning algorithm which is attractive for its simplicity.

In particular, consider the case of one intermediate point only, and suppose that trapezoidal velocity profiles are considered as motion primitives with the possibility to specify the initial and final point and the duration of the motion only; it is assumed that $\dot{q}_i = \dot{q}_f = 0$. If two segments with trapezoidal velocity profiles were generated, the manipulator joint would certainly reach

the intermediate point, but it would be forced to stop there, before continuing the motion towards the final point. A keen alternative is to start generating the second segment ahead of time with respect to the end of the first segment, using the sum of velocities (or positions) as a reference. In this way, the joint is guaranteed to reach the final position; crossing of the intermediate point at the specified instant of time is not guaranteed, though.

Figure 4.10 illustrates the time history of position, velocity and acceleration obtained with the data: $q_i = 0$, $q_f = 3\pi/2$, $t_i = 0$, $t_f = 2$. The intermediate point is located at $q = \pi$ with $t = 1$, the maximum acceleration values in the two segments are respectively $|\ddot{q}_c| = 6\pi$ and $|\ddot{q}_c| = 3\pi$, and the time anticipation is 0.18. As predicted, with time anticipation, the assigned intermediate position becomes a via point with the advantage of an overall shorter time duration. Notice, also, that velocity does not vanish at the intermediate point.

4.3 Operational Space Trajectories

A joint space trajectory planning algorithm generates a time sequence of values for the joint variables $\boldsymbol{q}(t)$ so that the manipulator is taken from the initial to the final configuration, eventually by moving through a sequence of intermediate configurations. The resulting end-effector motion is not easily predictable, in view of the nonlinear effects introduced by direct kinematics. Whenever it is desired that the end-effector motion follows a geometrically specified path in the *operational space*, it is necessary to plan trajectory execution directly in the same space. Planning can be done either by interpolating a sequence of prescribed path points or by generating the analytical motion primitive and the relative trajectory in a punctual way.

In both cases, the time sequence of the values attained by the operational space variables is utilized in real time to obtain the corresponding sequence of values of the joint space variables, via an inverse kinematics algorithm. In this regard, the computational complexity induced by trajectory generation in the operational space and related kinematic inversion sets an upper limit on the maximum sampling rate to generate the above sequences. Since these sequences constitute the reference inputs to the motion control system, a linear *microinterpolation* is typically carried out. In this way, the frequency at which reference inputs are updated is increased so as to enhance dynamic performance of the system.

Whenever the path is not to be followed exactly, its characterization can be performed through the assignment of N points specifying the values of the variables $\boldsymbol{x}_e$ chosen to describe the end-effector pose in the operational space at given time instants t_k, for $k = 1, \ldots, N$. Similar to what was presented in the above sections, the trajectory is generated by determining a smooth interpolating vector function between the various path points. Such a function

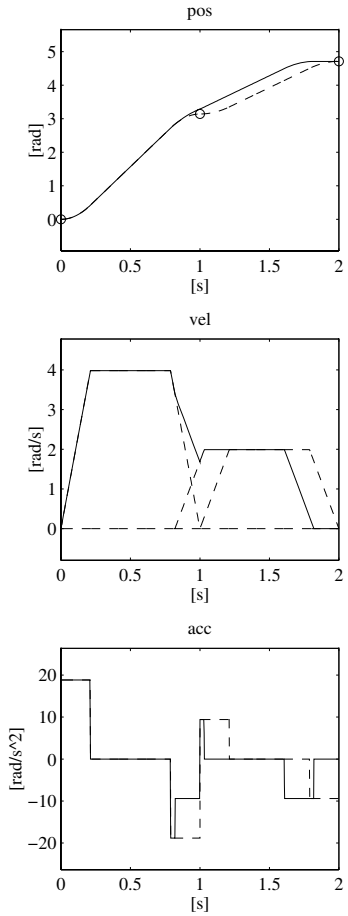

Fig. 4.10. Time history of position, velocity and acceleration with a timing law of interpolating linear polynomials with parabolic blends obtained by anticipating the generation of the second segment of trajectory

can be computed by applying to each component of $\boldsymbol{x}_e$ any of the interpolation techniques illustrated in Sect. 4.2.2 for the single joint variable.

Therefore, for given path (or via) points $\boldsymbol{x}_e(t_k)$, the corresponding components $x_{ei}(t_k)$, for $i = 1, \dots r$ (where r is the dimension of the operational space of interest) can be interpolated with a sequence of cubic polynomials, a sequence of linear polynomials with parabolic blends, and so on.

On the other hand, if the end-effector motion has to follow a prescribed trajectory of motion, this must be expressed analytically. It is then necessary

to refer to motion primitives defining the geometric features of the path and time primitives defining the timing law on the path itself.

4.3.1 Path Primitives

For the definition of *path primitives* it is convenient to refer to the parametric description of paths in space. Then let $\boldsymbol{p}$ be a (3×1) vector and $\boldsymbol{f}(\sigma)$ a continuous vector function defined in the interval $[\sigma_i, \sigma_f]$. Consider the equation

$$\boldsymbol{p} = \boldsymbol{f}(\sigma); \tag{4.31}$$

with reference to its geometric description, the sequence of values of $\boldsymbol{p}$ with σ varying in $[\sigma_i, \sigma_f]$ is termed *path* in space. The equation in (4.31) defines the *parametric representation* of the path Γ and the scalar σ is called parameter. As σ increases, the point $\boldsymbol{p}$ moves on the path in a given direction. This direction is said to be the direction induced on Γ by the parametric representation (4.31). A path is closed when $\boldsymbol{p}(\sigma_f) = \boldsymbol{p}(\sigma_i)$; otherwise it is open.

Let $\boldsymbol{p}_i$ be a point on the open path Γ on which a direction has been fixed. The *arc length* s of the generic point $\boldsymbol{p}$ is the length of the arc of Γ with extremes $\boldsymbol{p}$ and $\boldsymbol{p}_i$ if $\boldsymbol{p}$ follows $\boldsymbol{p}_i$, the opposite of this length if $\boldsymbol{p}$ precedes $\boldsymbol{p}_i$. The point $\boldsymbol{p}_i$ is said to be the origin of the arc length ($s = 0$).

From the above presentation it follows that to each value of s a well-determined path point corresponds, and then the arc length can be used as a parameter in a different parametric representation of the path Γ:

$$\boldsymbol{p} = \boldsymbol{f}(s); \tag{4.32}$$

the range of variation of the parameter s will be the sequence of arc lengths associated with the points of Γ.

Consider a path Γ represented by (4.32). Let $\boldsymbol{p}$ be a point corresponding to the arc length s. Except for special cases, $\boldsymbol{p}$ allows the definition of three unit vectors characterizing the path. The orientation of such vectors depends exclusively on the path geometry, while their direction depends also on the direction induced by (4.32) on the path.

The first of such unit vectors is the *tangent unit vector* denoted by $\boldsymbol{t}$. This vector is oriented along the direction induced on the path by s.

The second unit vector is the *normal unit vector* denoted by $\boldsymbol{n}$. This vector is oriented along the line intersecting $\boldsymbol{p}$ at a right angle with $\boldsymbol{t}$ and lies in the so-called *osculating plane* $\mathcal{O}$ (Fig. 4.11); such plane is the limit position of the plane containing the unit vector $\boldsymbol{t}$ and a point $\boldsymbol{p}' \in \Gamma$ when $\boldsymbol{p}'$ tends to $\boldsymbol{p}$ along the path. The direction of $\boldsymbol{n}$ is so that the path Γ, in the neighbourhood of $\boldsymbol{p}$ with respect to the plane containing $\boldsymbol{t}$ and normal to $\boldsymbol{n}$, lies on the same side of $\boldsymbol{n}$.

The third unit vector is the *binormal unit vector* denoted by $\boldsymbol{b}$. This vector is so that the frame $(\boldsymbol{t}, \boldsymbol{n}, \boldsymbol{b})$ is right-handed (Fig. 4.11). Notice that it is not always possible to define uniquely such a frame.

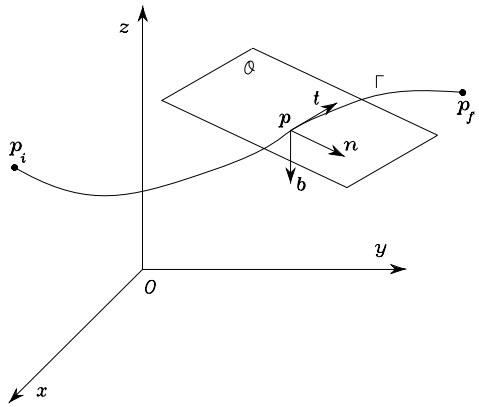

Fig. 4.11. Parametric representation of a path in space

It can be shown that the above three unit vectors are related by simple relations to the path representation Γ as a function of the arc length. In particular, it is

$$\begin{aligned} \boldsymbol{t} &= \frac{d\boldsymbol{p}}{ds} \\ \boldsymbol{n} &= \frac{1}{\left\| \frac{d^2\boldsymbol{p}}{ds^2} \right\|} \frac{d^2\boldsymbol{p}}{ds^2} \\ \boldsymbol{b} &= \boldsymbol{t} \times \boldsymbol{n}. \end{aligned} \tag{4.33}$$

Typical path parametric representations are reported below which are useful for trajectory generation in the operational space.

Rectilinear path

Consider the linear segment connecting point $\boldsymbol{p}_i$ to point $\boldsymbol{p}_f$. The parametric representation of this path is

$$\boldsymbol{p}(s) = \boldsymbol{p}_i + \frac{s}{\|\boldsymbol{p}_f - \boldsymbol{p}_i\|}(\boldsymbol{p}_f - \boldsymbol{p}_i). \tag{4.34}$$

Notice that $\boldsymbol{p}(0) = \boldsymbol{p}_i$ and $\boldsymbol{p}(\|\boldsymbol{p}_f - \boldsymbol{p}_i\|) = \boldsymbol{p}_f$. Hence, the direction induced on Γ by the parametric representation (4.34) is that going from $\boldsymbol{p}_i$ to $\boldsymbol{p}_f$. Differentiating (4.34) with respect to s gives

$$\frac{d\boldsymbol{p}}{ds} = \frac{1}{\|\boldsymbol{p}_f - \boldsymbol{p}_i\|}(\boldsymbol{p}_f - \boldsymbol{p}_i) \tag{4.35}$$

$$\frac{d^2\boldsymbol{p}}{ds^2} = \boldsymbol{0}. \tag{4.36}$$

In this case it is not possible to define the frame $(\boldsymbol{t}, \boldsymbol{n}, \boldsymbol{b})$ uniquely.

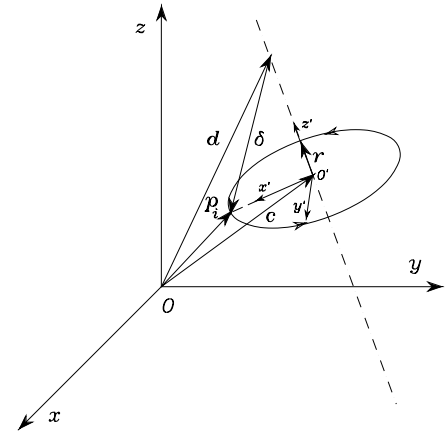

Fig. 4.12. Parametric representation of a circle in space

Circular path

Consider a circle Γ in space. Before deriving its parametric representation, it is necessary to introduce its significant parameters. Suppose that the circle is specified by assigning (Fig. 4.12):

- the unit vector of the circle axis $\boldsymbol{r}$,
- the position vector $\boldsymbol{d}$ of a point along the circle axis,
- the position vector $\boldsymbol{p}_i$ of a point on the circle.

With these parameters, the position vector $\boldsymbol{c}$ of the centre of the circle can be found. Let $\boldsymbol{\delta} = \boldsymbol{p}_i - \boldsymbol{d}$; for $\boldsymbol{p}_i$ not to be on the axis, i.e., for the circle not to degenerate into a point, it must be

$$|\boldsymbol{\delta}^T \boldsymbol{r}| < \|\boldsymbol{\delta}\|;$$

in this case it is

$$\boldsymbol{c} = \boldsymbol{d} + (\boldsymbol{\delta}^T \boldsymbol{r})\boldsymbol{r}. \tag{4.37}$$

It is now desired to find a parametric representation of the circle as a function of the arc length. Notice that this representation is very simple for a suitable choice of the reference frame. To see this, consider the frame O'–$x'y'z'$, where O' coincides with the centre of the circle, axis x' is oriented along the direction of the vector $\boldsymbol{p}_i - \boldsymbol{c}$, axis z' is oriented along $\boldsymbol{r}$ and axis y' is chosen so as to complete a right-handed frame. When expressed in this reference frame, the parametric representation of the circle is

$$\boldsymbol{p}'(s) = \begin{bmatrix} \rho\cos(s/\rho) \\ \rho\sin(s/\rho) \\ 0 \end{bmatrix}, \tag{4.38}$$

where $\rho = \|\boldsymbol{p}_i - \boldsymbol{c}\|$ is the radius of the circle and the point $\boldsymbol{p}_i$ has been assumed as the origin of the arc length. For a different reference frame, the path representation becomes

$$\boldsymbol{p}(s) = \boldsymbol{c} + \boldsymbol{R}\boldsymbol{p}'(s), \tag{4.39}$$

where $\boldsymbol{c}$ is expressed in the frame O–xyz and $\boldsymbol{R}$ is the rotation matrix of frame O'– $x'y'z'$ with respect to frame O–xyz which, in view of (2.3), can be written as

$$\boldsymbol{R} = [\,\boldsymbol{x}' \quad \boldsymbol{y}' \quad \boldsymbol{z}'\,];$$

$\boldsymbol{x}'$, $\boldsymbol{y}'$, $\boldsymbol{z}'$ indicate the unit vectors of the frame expressed in the frame O–xyz. Differentiating (4.39) with respect to s gives

$$\frac{d\boldsymbol{p}}{ds} = \boldsymbol{R}\begin{bmatrix} -\sin(s/\rho) \\ \cos(s/\rho) \\ 0 \end{bmatrix} \tag{4.40}$$

$$\frac{d^2\boldsymbol{p}}{ds^2} = \boldsymbol{R}\begin{bmatrix} -\cos(s/\rho)/\rho \\ -\sin(s/\rho)/\rho \\ 0 \end{bmatrix}. \tag{4.41}$$

4.3.2 Position

Let $\boldsymbol{x}_e$ be the vector of operational space variables expressing the *pose* of the manipulator's end-effector as in (2.80). Generating a trajectory in the operational space means to determine a function $\boldsymbol{x}_e(t)$ taking the end-effector frame from the initial to the final pose in a time t_f along a given path with a specific motion timing law. First, consider end-effector position. Orientation will follow.

Let $\boldsymbol{p}_e = \boldsymbol{f}(s)$ be the (3×1) vector of the parametric representation of the path Γ as a function of the arc length s; the origin of the end-effector frame moves from $\boldsymbol{p}_i$ to $\boldsymbol{p}_f$ in a time t_f. For simplicity, suppose that the origin of the arc length is at $\boldsymbol{p}_i$ and the direction induced on Γ is that going from $\boldsymbol{p}_i$ to $\boldsymbol{p}_f$. The arc length then goes from the value $s = 0$ at $t = 0$ to the value $s = s_f$ (path length) at $t = t_f$. The timing law along the path is described by the function $s(t)$.

In order to find an analytic expression for $s(t)$, any of the above techniques for joint trajectory generation can be employed. In particular, either a cubic polynomial or a sequence of linear segments with parabolic blends can be chosen for $s(t)$.

It is worth making some remarks on the time evolution of $\boldsymbol{p}_e$ on Γ, for a given timing law $s(t)$. The velocity of point $\boldsymbol{p}_e$ is given by the time derivative of $\boldsymbol{p}_e$

$$\dot{\boldsymbol{p}}_e = \dot{s}\frac{d\boldsymbol{p}_e}{ds} = \dot{s}\boldsymbol{t},$$

where $\boldsymbol{t}$ is the tangent vector to the path at point $\boldsymbol{p}$ in (4.33). Then, $\dot{s}$ represents the magnitude of the velocity vector relative to point $\boldsymbol{p}$, taken with the positive or negative sign depending on the direction of $\dot{\boldsymbol{p}}$ along $\boldsymbol{t}$. The magnitude of $\dot{\boldsymbol{p}}$ starts from zero at $t = 0$, then it varies with a parabolic or trapezoidal profile as per either of the above choices for $s(t)$, and finally it returns to zero at $t = t_f$.

As a first example, consider the segment connecting point $\boldsymbol{p}_i$ with point $\boldsymbol{p}_f$. The parametric representation of this path is given by (4.34). Velocity and acceleration of $\boldsymbol{p}_e$ can be easily computed by recalling the rule of differentiation of compound functions, i.e.,

$$\dot{\boldsymbol{p}}_e = \frac{\dot{s}}{\|\boldsymbol{p}_f - \boldsymbol{p}_i\|}(\boldsymbol{p}_f - \boldsymbol{p}_i) = \dot{s}\boldsymbol{t} \tag{4.42}$$

$$\ddot{\boldsymbol{p}}_e = \frac{\ddot{s}}{\|\boldsymbol{p}_f - \boldsymbol{p}_i\|}(\boldsymbol{p}_f - \boldsymbol{p}_i) = \ddot{s}\boldsymbol{t}. \tag{4.43}$$

As a further example, consider a circle Γ in space. From the parametric representation derived above, in view of (4.40), (4.41), velocity and acceleration of point $\boldsymbol{p}_e$ on the circle are

$$\dot{\boldsymbol{p}}_e = \boldsymbol{R}\begin{bmatrix} -\dot{s}\sin(s/\rho) \\ \dot{s}\cos(s/\rho) \\ 0 \end{bmatrix} \tag{4.44}$$

$$\ddot{\boldsymbol{p}}_e = \boldsymbol{R}\begin{bmatrix} -\dot{s}^2\cos(s/\rho)/\rho - \ddot{s}\sin(s/\rho) \\ -\dot{s}^2\sin(s/\rho)/\rho + \ddot{s}\cos(s/\rho) \\ 0 \end{bmatrix}. \tag{4.45}$$

Notice that the velocity vector is aligned with $\boldsymbol{t}$, and the acceleration vector is given by two contributions: the first is aligned with $\boldsymbol{n}$ and represents the centripetal acceleration, while the second is aligned with $\boldsymbol{t}$ and represents the tangential acceleration.

Finally, consider the path consisting of a sequence of $N+1$ points, $\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_N$, connected by N segments. A feasible parametric representation of the overall path is the following:

$$\boldsymbol{p}_e = \boldsymbol{p}_0 + \sum_{j=1}^{N} \frac{s_j}{\|\boldsymbol{p}_j - \boldsymbol{p}_{j-1}\|}(\boldsymbol{p}_j - \boldsymbol{p}_{j-1}), \tag{4.46}$$

with $j = 1, \ldots, N$. In (4.46) s_j is the arc length associated with the j-th segment of the path, connecting point $\boldsymbol{p}_{j-1}$ to point $\boldsymbol{p}_j$, defined as

$$s_j(t) = \begin{cases} 0 & 0 \le t \le t_{j-1} \\ s'_j(t) & t_{j-1} < t < t_j \\ \|\boldsymbol{p}_j - \boldsymbol{p}_{j-1}\| & t_j \le t \le t_f, \end{cases} \tag{4.47}$$

where $t_0 = 0$ and $t_N = t_f$ are respectively the initial and final time instants of the trajectory, t_j is the time instant corresponding to point $\boldsymbol{p}_j$ and $s'_j(t)$ can be an analytical function of cubic polynomial type, linear type with parabolic blends, and so forth, which varies continuously from the value $s'_j = 0$ at $t = t_{j-1}$ to the value $s'_j = \|\boldsymbol{p}_j - \boldsymbol{p}_{j-1}\|$ at $t = t_j$.

The velocity and acceleration of $\boldsymbol{p}_e$ can be easily found by differentiating (4.46):

$$\dot{\boldsymbol{p}}_e = \sum_{j=1}^{N} \frac{\dot{s}_j}{\|\boldsymbol{p}_j - \boldsymbol{p}_{j-1}\|}(\boldsymbol{p}_j - \boldsymbol{p}_{j-1}) = \sum_{j=1}^{N} \dot{s}_j \boldsymbol{t}_j \tag{4.48}$$

$$\ddot{\boldsymbol{p}}_e = \sum_{j=1}^{N} \frac{\ddot{s}_j}{\|\boldsymbol{p}_j - \boldsymbol{p}_{j-1}\|}(\boldsymbol{p}_j - \boldsymbol{p}_{j-1}) = \sum_{j=1}^{N} \ddot{s}_j \boldsymbol{t}_j, \tag{4.49}$$

where $\boldsymbol{t}_j$ is the tangent unit vector of the j-th segment.

Because of the discontinuity of the first derivative at the path points between two non-aligned segments, the manipulator will have to stop and then go along the direction of the following segment. Assumed a relaxation of the constraint to pass through the path points, it is possible to avoid a manipulator stop by connecting the segments near the above points, which will then be named *operational space via points* so as to guarantee, at least, continuity of the first derivative.

As already illustrated for planning of interpolating linear polynomials with parabolic blends passing by the via points in the joint space, the use of trapezoidal velocity profiles for the arc lengths allows the development of a rather simple planning algorithm

In detail, it will be sufficient to properly anticipate the generation of the single segments, before the preceding segment has been completed. This leads to modifying (4.47) as follows:

$$s_j(t) = \begin{cases} 0 & 0 \le t \le t_{j-1} - \Delta t_j \\ s'_j(t + \Delta t_j) & t_{j-1} - \Delta t_j < t < t_j - \Delta t_j \\ \|\boldsymbol{p}_j - \boldsymbol{p}_{j-1}\| & t_j - \Delta t_j \le t \le t_f - \Delta t_N, \end{cases} \tag{4.50}$$

where Δt_j is the time advance at which the j-th segment is generated, which can be recursively evaluated as

$$\Delta t_j = \Delta t_{j-1} + \delta t_j,$$

with $j = 1, \ldots, N$ e $\Delta t_0 = 0$. Notice that this time advance is given by the sum of two contributions: the former, Δt_{j-1}, accounts for the sum of the time advances at which the preceding segments have been generated, while the latter, δt_j, is the time advance at which the generation of the current segment starts.

4.3.3 Orientation

Consider now end-effector orientation. Typically, this is specified in terms of the rotation matrix of the (time-varying) end-effector frame with respect to the base frame. As is well known, the three columns of the rotation matrix represent the three unit vectors of the end-effector frame with respect to the base frame. To generate a trajectory, however, a linear interpolation on the unit vectors $\boldsymbol{n}_e$, $\boldsymbol{s}_e$, $\boldsymbol{a}_e$ describing the initial and final orientation does not guarantee orthonormality of the above vectors at each instant of time.

Euler angles

In view of the above difficulty, for trajectory generation purposes, orientation is often described in terms of the Euler angles triplet $\boldsymbol{\phi}_e = (\varphi, \vartheta, \psi)$ for which a timing law can be specified. Usually, $\boldsymbol{\phi}_e$ moves along the segment connecting its initial value $\boldsymbol{\phi}_i$ to its final value $\boldsymbol{\phi}_f$. Also in this case, it is convenient to choose a cubic polynomial or a linear segment with parabolic blends timing law. In this way, in fact, the angular velocity $\boldsymbol{\omega}_e$ of the time-varying frame, which is related to $\dot{\boldsymbol{\phi}}_e$ by the linear relationship (3.64), will have continuous magnitude.

Therefore, for given $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_f$ and timing law, the position, velocity and acceleration profiles are

$$\begin{aligned} \boldsymbol{\phi}_e &= \boldsymbol{\phi}_i + \frac{s}{\|\boldsymbol{\phi}_f - \boldsymbol{\phi}_i\|}(\boldsymbol{\phi}_f - \boldsymbol{\phi}_i) \\ \dot{\boldsymbol{\phi}}_e &= \frac{\dot{s}}{\|\boldsymbol{\phi}_f - \boldsymbol{\phi}_i\|}(\boldsymbol{\phi}_f - \boldsymbol{\phi}_i) \\ \ddot{\boldsymbol{\phi}}_e &= \frac{\ddot{s}}{\|\boldsymbol{\phi}_f - \boldsymbol{\phi}_i\|}(\boldsymbol{\phi}_f - \boldsymbol{\phi}_i); \end{aligned} \tag{4.51}$$

where the timing law for $s(t)$ has to be specified. The three unit vectors of the end-effector frame can be computed — with reference to Euler angles ZYZ — as in (2.18), the end-effector frame angular velocity as in (3.64), and the angular acceleration by differentiating (3.64) itself.

Angle and axis

An alternative way to generate a trajectory for orientation of clearer interpretation in the Cartesian space can be derived by resorting to the the angle and axis description presented in Sect. 2.5. Given two coordinate frames in the Cartesian space with the same origin and different orientation, it is always possible to determine a unit vector so that the second frame can be obtained from the first frame by a rotation of a proper angle about the axis of such unit vector.

Let $\boldsymbol{R}_i$ and $\boldsymbol{R}_f$ denote respectively the rotation matrices of the initial frame O_i–$x_i y_i z_i$ and the final frame O_f–$x_f y_f z_f$, both with respect to the base frame. The rotation matrix between the two frames can be computed by recalling that $\boldsymbol{R}_f = \boldsymbol{R}_i \boldsymbol{R}_f^i$; the expression in (2.5) leads to

$$\boldsymbol{R}_f^i = \boldsymbol{R}_i^T \boldsymbol{R}_f = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}.$$

If the matrix $\boldsymbol{R}^i(t)$ is defined to describe the transition from $\boldsymbol{R}_i$ to $\boldsymbol{R}_f$, it must be $\boldsymbol{R}^i(0) = \boldsymbol{I}$ and $\boldsymbol{R}^i(t_f) = \boldsymbol{R}_f^i$. Hence, the matrix $\boldsymbol{R}_f^i$ can be expressed as the rotation matrix about a fixed axis in space; the unit vector $\boldsymbol{r}^i$ of the axis and the angle of rotation ϑ_f can be computed by using (2.27):

$$\vartheta_f = \cos^{-1}\left(\frac{r_{11} + r_{22} + r_{33} - 1}{2}\right) \tag{4.52}$$

$$\boldsymbol{r} = \frac{1}{2\sin\vartheta_f}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \tag{4.53}$$

for $\sin\vartheta_f \neq 0$.

The matrix $\boldsymbol{R}^i(t)$ can be interpreted as a matrix $\boldsymbol{R}^i(\vartheta(t), \boldsymbol{r}^i)$ and computed via (2.25); it is then sufficient to assign a timing law to ϑ, of the type of those presented for the single joint with $\vartheta(0) = 0$ and $\vartheta(t_f) = \vartheta_f$, and compute the components of $\boldsymbol{r}^i$ from (4.52). Since $\boldsymbol{r}^i$ is constant, the resulting velocity and acceleration are respectively

$$\boldsymbol{\omega}^i = \dot{\vartheta}\,\boldsymbol{r}^i \tag{4.54}$$

$$\dot{\boldsymbol{\omega}}^i = \ddot{\vartheta}\,\boldsymbol{r}^i. \tag{4.55}$$

Finally, in order to characterize the end-effector orientation trajectory with respect to the base frame, the following transformations are needed:

$$\begin{aligned} \boldsymbol{R}_e(t) &= \boldsymbol{R}_i \boldsymbol{R}^i(t) \\ \boldsymbol{\omega}_e(t) &= \boldsymbol{R}_i \boldsymbol{\omega}^i(t) \\ \dot{\boldsymbol{\omega}}_e(t) &= \boldsymbol{R}_i \dot{\boldsymbol{\omega}}^i(t). \end{aligned}$$

Once a path and a trajectory have been specified in the operational space in terms of $\boldsymbol{p}_e(t)$ and $\boldsymbol{\phi}_e(t)$ or $\boldsymbol{R}_e(t)$, inverse kinematics techniques can be used to find the corresponding trajectories in the joint space $\boldsymbol{q}(t)$.

Bibliography

Trajectory planning for robot manipulators has been addressed since the first works in the field of robotics [178]. The formulation of the interpolation problem of the path points by means of different classes of functions has been suggested in [26].

The generation of motion trajectories through sequences of points in the joint space using splines is due to [131]. Alternative formulations for this problem are found in [56]. For a complete treatment of splines, including geometric properties and computational aspects, see [54]. In [155] a survey on the functions employed for trajectory planning of a single motion axis is given, which accounts for performance indices and effects of unmodelled flexible dynamics.

Cartesian space trajectory planning and the associated motion control problem have been originally treated in [179]. The systematic management of the motion by the via points using interpolating linear polynomials with parabolic blends has been proposed in [229]. A detailed presentation of the general aspects of the geometric primitives that can be utilized in robotics to define Cartesian space paths can be found in the computer graphics text [73].

Problems

4.1. Compute the joint trajectory from $q(0) = 1$ to $q(2) = 4$ with null initial and final velocities and accelerations.

4.2. Compute the timing law $q(t)$ for a joint trajectory with velocity profile of the type $\dot{q}(t) = k(1 - \cos(at))$ from $q(0) = 0$ to $q(2) = 3$.

4.3. Given the values for the joint variable: $q(0) = 0$, $q(2) = 2$, and $q(4) = 3$, compute the two fifth-order interpolating polynomials with continuous velocities and accelerations.

4.4. Show that the matrix $\boldsymbol{A}$ in (4.28) has a tridiagonal band structure.

4.5. Given the values for the joint variable: $q(0) = 0$, $q(2) = 2$, and $q(4) = 3$, compute the cubic interpolating spline with null initial and final velocities and accelerations.

4.6. Given the values for the joint variable: $q(0) = 0$, $q(2) = 2$, and $q(4) = 3$, find the interpolating polynomial with linear segments and parabolic blends with null initial and final velocities.

4.7. Find the timing law $\boldsymbol{p}(t)$ for a Cartesian space rectilinear path with trapezoidal velocity profile from $\boldsymbol{p}(0) = [\,0 \quad 0.5 \quad 0\,]^T$ to $\boldsymbol{p}(2) = [\,1 \quad -0.5 \quad 0\,]^T$.

4.8. Find the timing law $\boldsymbol{p}(t)$ for a Cartesian space circular path with trapezoidal velocity profile from $\boldsymbol{p}(0) = [\,0 \quad 0.5 \quad 1\,]^T$ to $\boldsymbol{p}(2) = [\,0 \quad -0.5 \quad 1\,]^T$; the circle is located in the plane $x = 0$ with centre at $\boldsymbol{c} = [\,0 \quad 0 \quad 1\,]^T$ and radius $\rho = 0.5$, and is executed clockwise for an observer aligned with x.

7

Dynamics

Derivation of the *dynamic model* of a manipulator plays an important role for simulation of motion, analysis of manipulator structures, and design of control algorithms. Simulating manipulator motion allows control strategies and motion planning techniques to be tested without the need to use a physically available system. The analysis of the dynamic model can be helpful for mechanical design of prototype arms. Computation of the forces and torques required for the execution of typical motions provides useful information for designing joints, transmissions and actuators. The goal of this chapter is to present two methods for derivation of the equations of motion of a manipulator in the *joint space*. The first method is based on the *Lagrange formulation* and is conceptually simple and systematic. The second method is based on the *Newton–Euler formulation* and yields the model in a recursive form; it is computationally more efficient since it exploits the typically open structure of the manipulator kinematic chain. Then, a technique for *dynamic parameter identification* is presented. Further, the problems of *direct dynamics* and *inverse dynamics* are formalized, and a technique for trajectory *dynamic scaling* is introduced, which adapts trajectory planning to the dynamic characteristics of the manipulator. The chapter ends with the derivation of the *dynamic model* of a manipulator in the *operational space* and the definition of the *dynamic manipulability ellipsoid*.

7.1 Lagrange Formulation

The dynamic model of a manipulator provides a description of the relationship between the joint actuator torques and the motion of the structure.

With *Lagrange formulation*, the equations of motion can be derived in a systematic way independently of the reference coordinate frame. Once a set of variables q_i, $i = 1, \dots, n$, termed *generalized coordinates*, are chosen which effectively describe the link positions of an n-DOF manipulator, the

Lagrangian of the mechanical system can be defined as a function of the generalized coordinates:

$$\mathcal{L} = \mathcal{T} - \mathcal{U} \tag{7.1}$$

where T and U respectively denote the total *kinetic energy* and *potential energy* of the system.

The Lagrange equations are expressed by

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = \xi_i \qquad i = 1, \dots, n \tag{7.2}$$

where ξ_i is the *generalized force* associated with the generalized coordinate q_i. Equations (7.2) can be written in compact form as

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{q}}}\right)^T - \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{q}}\right)^T = \boldsymbol{\xi} \tag{7.3}$$

where, for a manipulator with an open kinematic chain, the generalized coordinates are gathered in the vector of *joint variables* $\boldsymbol{q}$. The contributions to the generalized forces are given by the nonconservative forces, i.e., the joint actuator torques, the joint friction torques, as well as the joint torques induced by end-effector forces at the contact with the environment.[1]

The equations in (7.2) establish the relations existing between the generalized forces applied to the manipulator and the joint positions, velocities and accelerations. Hence, they allow the derivation of the dynamic model of the manipulator starting from the determination of kinetic energy and potential energy of the mechanical system.

Example 7.1

In order to understand the Lagrange formulation technique for deriving the dynamic model, consider again the simple case of the pendulum in Example 5.1. With reference to Fig. 5.8, let ϑ denote the angle with respect to the reference position of the body hanging down ($\vartheta = 0$). By choosing ϑ as the generalized coordinate, the kinetic energy of the system is given by

$$\mathcal{T} = \frac{1}{2} I \dot{\vartheta}^2 + \frac{1}{2} I_m k_r^2 \dot{\vartheta}^2.$$

The system potential energy, defined at less than a constant, is expressed by

$$\mathcal{U} = mg\ell(1 - \cos\vartheta).$$

Therefore, the Lagrangian of the system is

$$\mathcal{L} = \frac{1}{2} I \dot{\vartheta}^2 + \frac{1}{2} I_m k_r^2 \dot{\vartheta}^2 - mg\ell(1 - \cos\vartheta).$$

[1] The term *torque* is used as a synonym of joint *generalized force*.

Substituting this expression in the Lagrange equation in (7.2) yields

$$(I + I_m k_r^2)\ddot{\vartheta} + mg\ell \sin\vartheta = \xi.$$

The generalized force ξ is given by the contributions of the actuation torque τ at the joint and of the viscous friction torques $-F\dot{\vartheta}$ and $-F_m k_r^2 \dot{\vartheta}$, where the latter has been reported to the joint side. Hence, it is

$$\xi = \tau - F\dot{\vartheta} - F_m k_r^2 \dot{\vartheta}$$

leading to the complete dynamic model of the system as the second-order differential equation

$$(I + I_m k_r^2)\ddot{\vartheta} + (F + F_m k_r^2)\dot{\vartheta} + mg\ell \sin\vartheta = \tau.$$

It is easy to verify how this equation is equivalent to (5.25) when reported to the joint side.

7.1.1 Computation of Kinetic Energy

Consider a manipulator with n *rigid links*. The total kinetic energy is given by the sum of the contributions relative to the motion of each link and the contributions relative to the motion of each joint actuator:[2]

$$\mathcal{T} = \sum_{i=1}^{n} (\mathcal{T}_{\ell_i} + \mathcal{T}_{m_i}), \tag{7.4}$$

where $\mathcal{T}_{\ell_i}$ is the kinetic energy of Link i and $\mathcal{T}_{m_i}$ is the kinetic energy of the motor actuating Joint i.

The kinetic energy contribution of Link i is given by

$$\mathcal{T}_{\ell_i} = \frac{1}{2} \int_{V_{\ell_i}} \dot{\boldsymbol{p}}_i^{*T} \dot{\boldsymbol{p}}_i^{*} \rho dV, \tag{7.5}$$

where $\dot{\boldsymbol{p}}_i^*$ denotes the linear velocity vector and ρ is the density of the elementary particle of volume dV; V_{ℓ_i} is the volume of Link i.

Consider the position vector $\boldsymbol{p}_i^*$ of the elementary particle and the position vector $\boldsymbol{p}_{C_i}$ of the link centre of mass, both expressed in the *base frame*. One has

$$\boldsymbol{r}_i = [\, r_{ix} \quad r_{iy} \quad r_{iz} \,]^T = \boldsymbol{p}_i^* - \boldsymbol{p}_{\ell_i} \tag{7.6}$$

with

$$\boldsymbol{p}_{\ell_i} = \frac{1}{m_{\ell_i}} \int_{V_{\ell_i}} \boldsymbol{p}_i^* \rho dV \tag{7.7}$$

[2] Link 0 is fixed and thus gives no contribution.

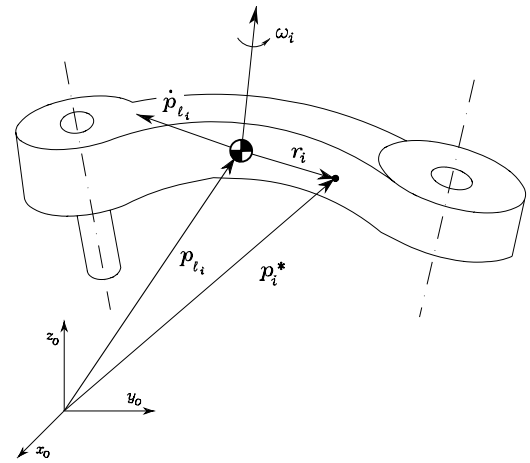

Fig. 7.1. Kinematic description of Link i for Lagrange formulation

where m_{ℓ_i} is the link mass. As a consequence, the link point velocity can be expressed as

$$\begin{aligned} \dot{\boldsymbol{p}}_i^* &= \dot{\boldsymbol{p}}_{\ell_i} + \boldsymbol{\omega}_i \times \boldsymbol{r}_i \\ &= \dot{\boldsymbol{p}}_{\ell_i} + \boldsymbol{S}(\boldsymbol{\omega}_i)\boldsymbol{r}_i, \end{aligned} \tag{7.8}$$

where $\dot{\boldsymbol{p}}_{\ell_i}$ is the linear velocity of the centre of mass and $\boldsymbol{\omega}_i$ is the angular velocity of the link (Fig. 7.1).

By substituting the velocity expression (7.8) into (7.5), it can be recognized that the kinetic energy of each link is formed by the following contributions.

Translational

The contribution is

$$\frac{1}{2} \int_{V_{\ell_i}} \dot{\boldsymbol{p}}_{\ell_i}^T \dot{\boldsymbol{p}}_{\ell_i} \rho dV = \frac{1}{2} m_{\ell_i} \dot{\boldsymbol{p}}_{\ell_i}^T \dot{\boldsymbol{p}}_{\ell_i}. \tag{7.9}$$

Mutual

The contribution is

$$2\left(\frac{1}{2} \int_{V_{\ell_i}} \dot{\boldsymbol{p}}_{\ell_i}^T \boldsymbol{S}(\boldsymbol{\omega}_i)\boldsymbol{r}_i \rho dV\right) = 2\left(\frac{1}{2} \dot{\boldsymbol{p}}_{\ell_i}^T \boldsymbol{S}(\boldsymbol{\omega}_i) \int_{V_{\ell_i}} (\boldsymbol{p}_i^* - \boldsymbol{p}_{\ell_i}) \rho dV\right) = 0$$

since, by virtue of (7.7), it is

$$\int_{V_{\ell_i}} \boldsymbol{p}_i^* \rho dV = \boldsymbol{p}_{\ell_i} \int_{V_{\ell_i}} \rho dV.$$

Rotational

The contribution is

$$\frac{1}{2}\int_{V_{\ell_i}} \boldsymbol{r}_i^T \boldsymbol{S}^T(\boldsymbol{\omega}_i)\boldsymbol{S}(\boldsymbol{\omega}_i)\boldsymbol{r}_i \rho dV = \frac{1}{2}\boldsymbol{\omega}_i^T \left(\int_{V_{\ell_i}} \boldsymbol{S}^T(\boldsymbol{r}_i)\boldsymbol{S}(\boldsymbol{r}_i)\rho dV\right)\boldsymbol{\omega}_i$$

where the property $\boldsymbol{S}(\boldsymbol{\omega}_i)\boldsymbol{r}_i = -\boldsymbol{S}(\boldsymbol{r}_i)\boldsymbol{\omega}_i$ has been exploited. In view of the expression of the matrix operator $\boldsymbol{S}(\cdot)$

$$\boldsymbol{S}(\boldsymbol{r}_i) = \begin{bmatrix} 0 & -r_{iz} & r_{iy} \\ r_{iz} & 0 & -r_{ix} \\ -r_{iy} & r_{ix} & 0 \end{bmatrix},$$

it is

$$\frac{1}{2}\int_{V_{\ell_i}} \boldsymbol{r}_i^T \boldsymbol{S}^T(\boldsymbol{\omega}_i)\boldsymbol{S}(\boldsymbol{\omega}_i)\boldsymbol{r}_i \rho dV = \frac{1}{2}\boldsymbol{\omega}_i^T \boldsymbol{I}_{\ell_i}\boldsymbol{\omega}_i. \tag{7.10}$$

The matrix

$$\boldsymbol{I}_{\ell_i} = \begin{bmatrix} \int(r_{iy}^2 + r_{iz}^2)\rho dV & -\int r_{ix}r_{iy}\rho dV & -\int r_{ix}r_{iz}\rho dV \\ * & \int(r_{ix}^2 + r_{iz}^2)\rho dV & -\int r_{iy}r_{iz}\rho dV \\ * & * & \int(r_{ix}^2 + r_{iy}^2)\rho dV \end{bmatrix} \tag{7.11}$$

$$= \begin{bmatrix} I_{\ell_i xx} & -I_{\ell_i xy} & -I_{\ell_i xz} \\ * & I_{\ell_i yy} & -I_{\ell_i yz} \\ * & * & I_{\ell_i zz} \end{bmatrix}.$$

is symmetric[3] and represents the *inertia tensor* relative to the centre of mass of Link i when expressed in the base frame. Notice that the position of Link i depends on the manipulator configuration and thus the inertia tensor, when expressed in the base frame, is configuration-dependent. If the angular velocity of Link i is expressed with reference to a frame attached to the link (as in the Denavit–Hartenberg convention), it is

$$\boldsymbol{\omega}_i^i = \boldsymbol{R}_i^T \boldsymbol{\omega}_i$$

where $\boldsymbol{R}_i$ is the rotation matrix from Link i frame to the base frame. When referred to the link frame, the inertia tensor is constant. Let $\boldsymbol{I}_{\ell_i}^i$ denote such tensor; then it is easy to verify the following relation:

$$\boldsymbol{I}_{\ell_i} = \boldsymbol{R}_i \boldsymbol{I}_{\ell_i}^i \boldsymbol{R}_i^T. \tag{7.12}$$

If the axes of Link i frame coincide with the central axes of inertia, then the inertia products are null and the inertia tensor relative to the centre of mass is a diagonal matrix.

[3] The symbol '$*$' has been used to avoid rewriting the symmetric elements.

By summing the translational and rotational contributions (7.9) and (7.10), the kinetic energy of Link i is

$$\mathcal{T}_{\ell_i} = \frac{1}{2}m_{\ell_i}\dot{\boldsymbol{p}}_{\ell_i}^T\dot{\boldsymbol{p}}_{\ell_i} + \frac{1}{2}\boldsymbol{\omega}_i^T \boldsymbol{R}_i \boldsymbol{I}_{\ell_i}^i \boldsymbol{R}_i^T \boldsymbol{\omega}_i. \tag{7.13}$$

At this point, it is necessary to express the kinetic energy as a function of the generalized coordinates of the system, that are the joint variables. To this end, the geometric method for Jacobian computation can be applied to the intermediate link other than the end-effector, yielding

$$\dot{\boldsymbol{p}}_{\ell_i} = \boldsymbol{\jmath}_{P1}^{(\ell_i)}\dot{q}_1 + \ldots + \boldsymbol{\jmath}_{Pi}^{(\ell_i)}\dot{q}_i = \boldsymbol{J}_P^{(\ell_i)}\dot{\boldsymbol{q}} \tag{7.14}$$

$$\boldsymbol{\omega}_i = \boldsymbol{\jmath}_{O1}^{(\ell_i)}\dot{q}_1 + \ldots + \boldsymbol{\jmath}_{Oi}^{(\ell_i)}\dot{q}_i = \boldsymbol{J}_O^{(\ell_i)}\dot{\boldsymbol{q}}, \tag{7.15}$$

where the contributions of the Jacobian columns relative to the joint velocities have been taken into account up to current Link i. The Jacobians to consider are then:

$$\boldsymbol{J}_P^{(\ell_i)} = \begin{bmatrix} \boldsymbol{\jmath}_{P1}^{(\ell_i)} & \ldots & \boldsymbol{\jmath}_{Pi}^{(\ell_i)} & \boldsymbol{0} & \ldots & \boldsymbol{0} \end{bmatrix} \tag{7.16}$$

$$\boldsymbol{J}_O^{(\ell_i)} = \begin{bmatrix} \boldsymbol{\jmath}_{O1}^{(\ell_i)} & \ldots & \boldsymbol{\jmath}_{Oi}^{(\ell_i)} & \boldsymbol{0} & \ldots & \boldsymbol{0} \end{bmatrix}; \tag{7.17}$$

the columns of the matrices in (7.16) and (7.17) can be computed according to (3.30), giving

$$\boldsymbol{\jmath}_{Pj}^{(\ell_i)} = \begin{cases} \boldsymbol{z}_{j-1} & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{z}_{j-1} \times (\boldsymbol{p}_{\ell_i} - \boldsymbol{p}_{j-1}) & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{7.18}$$

$$\boldsymbol{\jmath}_{Oj}^{(\ell_i)} = \begin{cases} \boldsymbol{0} & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{z}_{j-1} & \text{for a } \textit{revolute} \text{ joint.} \end{cases} \tag{7.19}$$

where $\boldsymbol{p}_{j-1}$ is the position vector of the origin of Frame $j-1$ and $\boldsymbol{z}_{j-1}$ is the unit vector of axis z of Frame $j-1$. It follows that the kinetic energy of Link i in (7.13) can be written as

$$\mathcal{T}_{\ell_i} = \frac{1}{2}m_{\ell_i}\dot{\boldsymbol{q}}^T \boldsymbol{J}_P^{(\ell_i)T}\boldsymbol{J}_P^{(\ell_i)}\dot{\boldsymbol{q}} + \frac{1}{2}\dot{\boldsymbol{q}}^T \boldsymbol{J}_O^{(\ell_i)T}\boldsymbol{R}_i \boldsymbol{I}_{\ell_i}^i \boldsymbol{R}_i^T \boldsymbol{J}_O^{(\ell_i)}\dot{\boldsymbol{q}}. \tag{7.20}$$

The kinetic energy contribution of the motor of Joint i can be computed in a formally analogous way to that of the link. Consider the typical case of rotary electric motors (that can actuate both revolute and prismatic joints by means of suitable transmissions). It can be assumed that the contribution of the fixed part (stator) is included in that of the link on which such motor is located, and thus the sole contribution of the rotor is to be computed.

With reference to Fig. 7.2, the motor of Joint i is assumed to be located on Link $i-1$. In practice, in the design of the mechanical structure of an open kinematic chain manipulator one attempts to locate the motors as close as possible to the base of the manipulator so as to lighten the dynamic load of

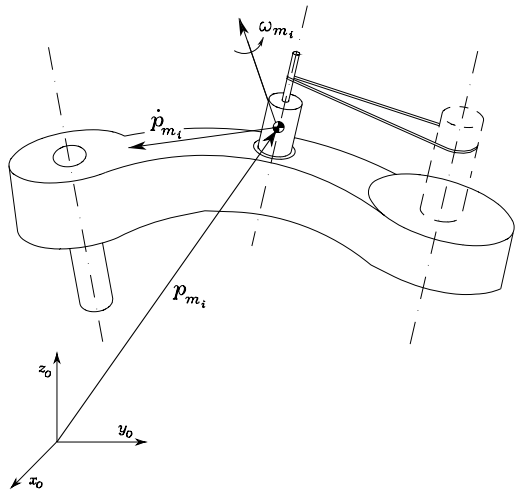

Fig. 7.2. Kinematic description of Motor i

the first joints of the chain. The joint actuator torques are delivered by the motors by means of mechanical transmissions (gears).[4] The contribution of the gears to the kinetic energy can be suitably included in that of the motor. It is assumed that no induced motion occurs, i.e., the motion of Joint i does not actuate the motion of other joints.

The kinetic energy of Rotor i can be written as

$$\mathcal{T}_{m_i} = \frac{1}{2} m_{m_i} \dot{\boldsymbol{p}}_{m_i}^T \dot{\boldsymbol{p}}_{m_i} + \frac{1}{2} \boldsymbol{\omega}_{m_i}^T \boldsymbol{I}_{m_i} \boldsymbol{\omega}_{m_i}, \tag{7.21}$$

where m_{m_i} is the mass of the rotor, $\dot{\boldsymbol{p}}_{m_i}$ denotes the linear velocity of the centre of mass of the rotor, $\boldsymbol{I}_{m_i}$ is the inertia tensor of the rotor relative to its centre of mass, and $\boldsymbol{\omega}_{m_i}$ denotes the angular velocity of the rotor.

Let ϑ_{m_i} denote the angular position of the rotor. On the assumption of a *rigid transmission*, one has

$$k_{ri} \dot{q}_i = \dot{\vartheta}_{m_i} \tag{7.22}$$

where k_{ri} is the gear reduction ratio. Notice that, in the case of actuation of a prismatic joint, the gear reduction ratio is a dimensional quantity.

According to the angular velocity composition rule (3.18) and the relation (7.22), the total angular velocity of the rotor is

$$\boldsymbol{\omega}_{m_i} = \boldsymbol{\omega}_{i-1} + k_{ri} \dot{q}_i \boldsymbol{z}_{m_i} \tag{7.23}$$

where $\boldsymbol{\omega}_{i-1}$ is the angular velocity of Link $i-1$ on which the motor is located, and $\boldsymbol{z}_{m_i}$ denotes the unit vector along the rotor axis.

[4] Alternatively, the joints may be actuated by torque motors directly coupled to the rotation axis without gears.

To express the rotor kinetic energy as a function of the joint variables, it is worth expressing the linear velocity of the rotor centre of mass — similarly to (7.14) — as

$$\dot{\boldsymbol{p}}_{m_i} = \boldsymbol{J}_P^{(m_i)} \dot{\boldsymbol{q}}. \tag{7.24}$$

The Jacobian to compute is then

$$\boldsymbol{J}_P^{(m_i)} = \begin{bmatrix} \boldsymbol{\jmath}_{P1}^{(m_i)} & \dots & \boldsymbol{\jmath}_{P,i-1}^{(m_i)} & \boldsymbol{0} & \dots & \boldsymbol{0} \end{bmatrix} \tag{7.25}$$

whose columns are given by

$$\boldsymbol{\jmath}_{Pj}^{(m_i)} = \begin{cases} \boldsymbol{z}_{j-1} & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{z}_{j-1} \times (\boldsymbol{p}_{m_i} - \boldsymbol{p}_{j-1}) & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{7.26}$$

where $\boldsymbol{p}_{j-1}$ is the position vector of the origin of Frame $j-1$. Notice that $\boldsymbol{\jmath}_{Pi}^{(m_i)} = \boldsymbol{0}$ in (7.25), since the centre of mass of the rotor has been taken along its axis of rotation.

The angular velocity in (7.23) can be expressed as a function of the joint variables, i.e.,

$$\boldsymbol{\omega}_{m_i} = \boldsymbol{J}_O^{(m_i)} \dot{\boldsymbol{q}}. \tag{7.27}$$

The Jacobian to compute is then

$$\boldsymbol{J}_O^{(m_i)} = \begin{bmatrix} \boldsymbol{\jmath}_{O1}^{(m_i)} & \dots & \boldsymbol{\jmath}_{O,i-1}^{(m_i)} & \boldsymbol{\jmath}_{Oi}^{(m_i)} & \boldsymbol{0} & \dots & \boldsymbol{0} \end{bmatrix} \tag{7.28}$$

whose columns, in view of (7.23), (7.15), are respectively given by

$$\boldsymbol{\jmath}_{Oj}^{(m_i)} = \begin{cases} \boldsymbol{\jmath}_{Oj}^{(\ell_i)} & j = 1, \dots, i-1 \\ k_{ri} \boldsymbol{z}_{m_i} & j = i. \end{cases} \tag{7.29}$$

To compute the second relation in (7.29), it is sufficient to know the components of the unit vector of the rotor rotation axis $\boldsymbol{z}_{m_i}$ with respect to the base frame. Hence, the kinetic energy of Rotor i can be written as

$$\mathcal{T}_{m_i} = \frac{1}{2} m_{m_i} \dot{\boldsymbol{q}}^T \boldsymbol{J}_P^{(m_i)T} \boldsymbol{J}_P^{(m_i)} \dot{\boldsymbol{q}} + \frac{1}{2} \dot{\boldsymbol{q}}^T \boldsymbol{J}_O^{(m_i)T} \boldsymbol{R}_{m_i} \boldsymbol{I}_{m_i}^{m_i} \boldsymbol{R}_{m_i}^T \boldsymbol{J}_O^{(m_i)} \dot{\boldsymbol{q}}. \tag{7.30}$$

Finally, by summing the various contributions relative to the single links (7.20) and single rotors (7.30) as in (7.4), the total kinetic energy of the manipulator with actuators is given by the quadratic form

$$\mathcal{T} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij}(\boldsymbol{q}) \dot{q}_i \dot{q}_j = \frac{1}{2} \dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q}) \dot{\boldsymbol{q}} \tag{7.31}$$

where

$$\boldsymbol{B}(\boldsymbol{q}) = \sum_{i=1}^{n} \Big(m_{\ell_i} \boldsymbol{J}_P^{(\ell_i)T} \boldsymbol{J}_P^{(\ell_i)} + \boldsymbol{J}_O^{(\ell_i)T} \boldsymbol{R}_i \boldsymbol{I}_{\ell_i}^i \boldsymbol{R}_i^T \boldsymbol{J}_O^{(\ell_i)} + m_{m_i} \boldsymbol{J}_P^{(m_i)T} \boldsymbol{J}_P^{(m_i)} + \boldsymbol{J}_O^{(m_i)T} \boldsymbol{R}_{m_i} \boldsymbol{I}_{m_i}^{m_i} \boldsymbol{R}_{m_i}^T \boldsymbol{J}_O^{(m_i)} \Big) \tag{7.32}$$

is the $(n \times n)$ *inertia matrix* which is:

- *symmetric*,
- *positive definite*,
- (in general) *configuration-dependent*.

7.1.2 Computation of Potential Energy

As done for kinetic energy, the potential energy stored in the manipulator is given by the sum of the contributions relative to each link as well as to each rotor:

$$\mathcal{U} = \sum_{i=1}^{n} (\mathcal{U}_{\ell_i} + \mathcal{U}_{m_i}). \tag{7.33}$$

On the assumption of *rigid links*, the contribution due only to gravitational forces[5] is expressed by

$$\mathcal{U}_{\ell_i} = -\int_{V_{\ell_i}} \boldsymbol{g}_0^T \boldsymbol{p}_i^* \rho dV = -m_{\ell_i} \boldsymbol{g}_0^T \boldsymbol{p}_{\ell_i} \tag{7.34}$$

where $\boldsymbol{g}_0$ is the gravity acceleration vector in the base frame (e.g., $\boldsymbol{g}_0 = [0 \quad 0 \quad -g]^T$ if z is the vertical axis), and (7.7) has been utilized for the coordinates of the centre of mass of Link i. As regards the contribution of Rotor i, similarly to (7.34), one has

$$\mathcal{U}_{m_i} = -m_{m_i} \boldsymbol{g}_0^T \boldsymbol{p}_{m_i}. \tag{7.35}$$

By substituting (7.34), (7.35) into (7.33), the *potential energy* is given by

$$\mathcal{U} = -\sum_{i=1}^{n} (m_{\ell_i} \boldsymbol{g}_0^T \boldsymbol{p}_{\ell_i} + m_{m_i} \boldsymbol{g}_0^T \boldsymbol{p}_{m_i}) \tag{7.36}$$

which reveals that potential energy, through the vectors $\boldsymbol{p}_{\ell_i}$ and $\boldsymbol{p}_{m_i}$ is a function only of the joint variables $\boldsymbol{q}$, and *not* of the joint velocities $\dot{\boldsymbol{q}}$.

7.1.3 Equations of Motion

Having computed the total kinetic and potential energy of the system as in (7.31), (7.36), the Lagrangian (7.1) for the manipulator can be written as

$$\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathcal{T}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \mathcal{U}(\boldsymbol{q}). \tag{7.37}$$

Taking the derivatives required by Lagrange equations in (7.3) and recalling that U does not depend on $\dot{\boldsymbol{q}}$ yields

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\xi} \tag{7.38}$$

[5] In the case of link flexibility, one would have an additional contribution due to elastic forces.

where

$$\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \dot{\boldsymbol{B}}(\boldsymbol{q})\dot{\boldsymbol{q}} - \frac{1}{2}\left(\frac{\partial}{\partial \boldsymbol{q}}\left(\dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}}\right)\right)^T + \left(\frac{\partial \mathcal{U}(\boldsymbol{q})}{\partial \boldsymbol{q}}\right)^T.$$

In detail, noticing that $\mathcal{U}$ in (7.36) does not depend on $\dot{\boldsymbol{q}}$ and accounting for (7.31) yields

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) = \frac{d}{dt}\left(\frac{\partial \mathcal{T}}{\partial \dot{q}_i}\right) &= \sum_{j=1}^{n} b_{ij}(\boldsymbol{q})\ddot{q}_j + \sum_{j=1}^{n} \frac{db_{ij}(\boldsymbol{q})}{dt}\dot{q}_j \\ &= \sum_{j=1}^{n} b_{ij}(\boldsymbol{q})\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} \frac{\partial b_{ij}(\boldsymbol{q})}{\partial q_k}\dot{q}_k\dot{q}_j\end{aligned}$$

and

$$\frac{\partial \mathcal{T}}{\partial q_i} = \frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{n} \frac{\partial b_{jk}(\boldsymbol{q})}{\partial q_i}\dot{q}_k\dot{q}_j$$

where the indices of summation have been conveniently switched. Further, in view of (7.14), (7.24), it is

$$\begin{aligned}\frac{\partial \mathcal{U}}{\partial q_i} &= -\sum_{j=1}^{n}\left(m_{\ell_j}\boldsymbol{g}_0^T\frac{\partial \boldsymbol{p}_{\ell_j}}{\partial q_i} + m_{m_j}\boldsymbol{g}_0^T\frac{\partial \boldsymbol{p}_{m_j}}{\partial q_i}\right) \\ &= -\sum_{j=1}^{n}\left(m_{\ell_j}\boldsymbol{g}_0^T\boldsymbol{\jmath}_{Pi}^{(\ell_j)}(\boldsymbol{q}) + m_{m_j}\boldsymbol{g}_0^T\boldsymbol{\jmath}_{Pi}^{(m_j)}(\boldsymbol{q})\right) = g_i(\boldsymbol{q})\end{aligned} \tag{7.39}$$

where, again, the index of summation has been changed.

As a result, the equations of motion are

$$\sum_{j=1}^{n} b_{ij}(\boldsymbol{q})\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} h_{ijk}(\boldsymbol{q})\dot{q}_k\dot{q}_j + g_i(\boldsymbol{q}) = \xi_i \qquad i = 1, \ldots, n. \tag{7.40}$$

where

$$h_{ijk} = \frac{\partial b_{ij}}{\partial q_k} - \frac{1}{2}\frac{\partial b_{jk}}{\partial q_i}. \tag{7.41}$$

A physical interpretation of (7.40) reveals that:

- For the *acceleration terms*:
  - The coefficient b_{ii} represents the moment of inertia at Joint i axis, in the current manipulator configuration, when the other joints are blocked.
  - The coefficient b_{ij} accounts for the effect of acceleration of Joint j on Joint j.
- For the *quadratic velocity terms*:
  - The term $h_{ijj}\dot{q}_j^2$ represents the *centrifugal* effect induced on Joint i by velocity of Joint j; notice that $h_{iii} = 0$, since $\partial b_{ii}/\partial q_i = 0$.

  - The term $h_{ijk}\dot{q}_j\dot{q}_k$ represents the *Coriolis* effect induced on Joint i by velocities of Joints j and k.
- For the *configuration-dependent terms*:
  - The term g_i represents the moment generated at Joint i axis of the manipulator, in the current configuration, by the presence of gravity.

Some joint dynamic couplings, e.g., coefficients b_{ij} and h_{ijk}, may be reduced or zeroed when designing the structure, so as to simplify the control problem.

Regarding the nonconservative forces doing work at the manipulator joints, these are given by the *actuation torques* $\boldsymbol{\tau}$ minus the *viscous friction* torques $\boldsymbol{F}_v\dot{\boldsymbol{q}}$ and the static friction torques $\boldsymbol{f}_s(\boldsymbol{q},\dot{\boldsymbol{q}})$: $\boldsymbol{F}_v$ denotes the $(n\times n)$ diagonal matrix of viscous friction coefficients. As a simplified model of static friction torques, one may consider the *Coulomb friction* torques $\boldsymbol{F}_s\,\mathbf{sgn}\,(\dot{\boldsymbol{q}})$, where $\boldsymbol{F}_s$ is an $(n\times n)$ diagonal matrix and $\mathbf{sgn}\,(\dot{\boldsymbol{q}})$ denotes the $(n\times 1)$ vector whose components are given by the sign functions of the single joint velocities.

If the manipulator's end-effector is in contact with an environment, a portion of the actuation torques is used to balance the torques induced at the joints by the contact forces. According to a relation formally analogous to (3.111), such torques are given by $\boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{h}_e$ where $\boldsymbol{h}_e$ denotes the vector of force and moment exerted by the end-effector on the environment.

In summary, the equations of motion (7.38) can be rewritten in the compact matrix form which represents the *joint space dynamic model*:

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}}+\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}+\boldsymbol{F}_v\dot{\boldsymbol{q}}+\boldsymbol{F}_s\,\mathbf{sgn}\,(\dot{\boldsymbol{q}})+\boldsymbol{g}(\boldsymbol{q})=\boldsymbol{\tau}-\boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{h}_e \tag{7.42}$$

where $\boldsymbol{C}$ is a suitable $(n\times n)$ matrix such that its elements c_{ij} satisfy the equation

$$\sum_{j=1}^{n}c_{ij}\dot{q}_j=\sum_{j=1}^{n}\sum_{k=1}^{n}h_{ijk}\dot{q}_k\dot{q}_j. \tag{7.43}$$

7.2 Notable Properties of Dynamic Model

In the following, two *notable properties* of the dynamic model are presented which will be useful for dynamic parameter identification as well as for deriving control algorithms.

7.2.1 Skew-symmetry of Matrix $\dot{\boldsymbol{B}}-2\boldsymbol{C}$

The choice of the matrix $\boldsymbol{C}$ is not unique, since there exist several matrices $\boldsymbol{C}$ whose elements satisfy (7.43). A particular choice can be obtained by

elaborating the term on the right-hand side of (7.43) and accounting for the expressions of the coefficients h_{ijk} in (7.41). To this end, one has

$$\begin{aligned}\sum_{j=1}^{n}c_{ij}\dot{q}_j&=\sum_{j=1}^{n}\sum_{k=1}^{n}h_{ijk}\dot{q}_k\dot{q}_j\\&=\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{\partial b_{ij}}{\partial q_k}-\frac{1}{2}\frac{\partial b_{jk}}{\partial q_i}\right)\dot{q}_k\dot{q}_j.\end{aligned}$$

Splitting the first term on the right-hand side by an opportune switch of summation between j and k yields

$$\sum_{j=1}^{n}c_{ij}\dot{q}_j=\frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{\partial b_{ij}}{\partial q_k}\dot{q}_k\dot{q}_j+\frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{n}\left(\frac{\partial b_{ik}}{\partial q_j}-\frac{\partial b_{jk}}{\partial q_i}\right)\dot{q}_k\dot{q}_j.$$

As a consequence, the generic element of $\boldsymbol{C}$ is

$$c_{ij}=\sum_{k=1}^{n}c_{ijk}\dot{q}_k \tag{7.44}$$

where the coefficients

$$c_{ijk}=\frac{1}{2}\left(\frac{\partial b_{ij}}{\partial q_k}+\frac{\partial b_{ik}}{\partial q_j}-\frac{\partial b_{jk}}{\partial q_i}\right) \tag{7.45}$$

are termed *Christoffel symbols of the first type*. Notice that, in view of the symmetry of $\boldsymbol{B}$, it is

$$c_{ijk}=c_{ikj}. \tag{7.46}$$

This choice for the matrix $\boldsymbol{C}$ leads to deriving the following notable property of the equations of motion (7.42). The matrix

$$\boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}})=\dot{\boldsymbol{B}}(\boldsymbol{q})-2\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{7.47}$$

is *skew-symmetric*; that is, given any $(n\times 1)$ vector $\boldsymbol{w}$, the following relation holds:

$$\boldsymbol{w}^T\boldsymbol{N}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{w}=0. \tag{7.48}$$

In fact, substituting the coefficient (7.45) into (7.44) gives

$$\begin{aligned}c_{ij}&=\frac{1}{2}\sum_{k=1}^{n}\frac{\partial b_{ij}}{\partial q_k}\dot{q}_k+\frac{1}{2}\sum_{k=1}^{n}\left(\frac{\partial b_{ik}}{\partial q_j}-\frac{\partial b_{jk}}{\partial q_i}\right)\dot{q}_k\\&=\frac{1}{2}\dot{b}_{ij}+\frac{1}{2}\sum_{k=1}^{n}\left(\frac{\partial b_{ik}}{\partial q_j}-\frac{\partial b_{jk}}{\partial q_i}\right)\dot{q}_k\end{aligned}$$

and then the expression of the generic element of the matrix $\boldsymbol{N}$ in (7.47) is

$$n_{ij}=\dot{b}_{ij}-2c_{ij}=\sum_{k=1}^{n}\left(\frac{\partial b_{jk}}{\partial q_i}-\frac{\partial b_{ik}}{\partial q_j}\right)\dot{q}_k.$$

The result follows by observing that

$$n_{ij} = -n_{ji}.$$

An interesting property which is a direct implication of the skew-symmetry of $\boldsymbol{N}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is that, by setting $\boldsymbol{w} = \dot{\boldsymbol{q}}$,

$$\dot{\boldsymbol{q}}^T \boldsymbol{N}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = 0; \tag{7.49}$$

notice that (7.49) does not imply (7.48), since $\boldsymbol{N}$ is a function of $\dot{\boldsymbol{q}}$, too.

It can be shown that (7.49) holds for any choice of the matrix $\boldsymbol{C}$, since it is a result of the principle of conservation of energy (*Hamilton*). By virtue of this principle, the total time derivative of kinetic energy is balanced by the power generated by all the forces acting on the manipulator joints. For the mechanical system at issue, one may write

$$\frac{1}{2}\frac{d}{dt}\big(\dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}}\big) = \dot{\boldsymbol{q}}^T\big(\boldsymbol{\tau} - \boldsymbol{F}_v\dot{\boldsymbol{q}} - \boldsymbol{F}_s\,\mathbf{sgn}\,(\dot{\boldsymbol{q}}) - \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{h}_e\big). \tag{7.50}$$

Taking the derivative on the left-hand side of (7.50) gives

$$\frac{1}{2}\dot{\boldsymbol{q}}^T \dot{\boldsymbol{B}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}}$$

and substituting the expression of $\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}}$ in (7.42) yields

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}\big(\dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}}\big) = {} & \frac{1}{2}\dot{\boldsymbol{q}}^T\big(\dot{\boldsymbol{B}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\big)\dot{\boldsymbol{q}} \\ & + \dot{\boldsymbol{q}}^T\big(\boldsymbol{\tau} - \boldsymbol{F}_v\dot{\boldsymbol{q}} - \boldsymbol{F}_s\,\mathbf{sgn}\,(\dot{\boldsymbol{q}}) - \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{h}_e\big).\end{aligned} \tag{7.51}$$

A direct comparison of the right-hand sides of (7.50) and (7.51) leads to the result established by (7.49).

To summarize, the relation (7.49) holds for any choice of the matrix $\boldsymbol{C}$, since it is a direct consequence of the physical properties of the system, whereas the relation (7.48) holds only for the particular choice of the elements of $\boldsymbol{C}$ as in (7.44), (7.45).

7.2.2 Linearity in the Dynamic Parameters

An important property of the dynamic model is the *linearity* with respect to the *dynamic parameters* characterizing the manipulator links and rotors.

In order to determine such parameters, it is worth associating the kinetic and potential energy contributions of each rotor with those of the link on which it is located. Hence, by considering the union of Link i and Rotor $i+1$ (*augmented Link* i), the kinetic energy contribution is given by

$$\mathcal{T}_i = \mathcal{T}_{\ell_i} + \mathcal{T}_{m_{i+1}} \tag{7.52}$$

where

$$\mathcal{T}_{\ell_i} = \frac{1}{2}m_{\ell i}\dot{\boldsymbol{p}}_{\ell_i}^T\dot{\boldsymbol{p}}_{\ell_i} + \frac{1}{2}\boldsymbol{\omega}_i^T\boldsymbol{I}_{\ell_i}\boldsymbol{\omega}_i \tag{7.53}$$

and

$$\mathcal{T}_{m_{i+1}} = \frac{1}{2}m_{m_{i+1}}\dot{\boldsymbol{p}}_{m_{i+1}}^T\dot{\boldsymbol{p}}_{m_{i+1}} + \frac{1}{2}\boldsymbol{\omega}_{m_{i+1}}^T\boldsymbol{I}_{m_{i+1}}\boldsymbol{\omega}_{m_{i+1}}. \tag{7.54}$$

With reference to the centre of mass of the augmented link, the linear velocities of the link and rotor can be expressed according to (3.26) as

$$\dot{\boldsymbol{p}}_{\ell_i} = \dot{\boldsymbol{p}}_{C_i} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{C_i,\ell_i} \tag{7.55}$$

$$\dot{\boldsymbol{p}}_{m_{i+1}} = \dot{\boldsymbol{p}}_{C_i} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{C_i,m_{i+1}} \tag{7.56}$$

with

$$\boldsymbol{r}_{C_i,\ell_i} = \boldsymbol{p}_{\ell_i} - \boldsymbol{p}_{C_i} \tag{7.57}$$

$$\boldsymbol{r}_{C_i,m_{i+1}} = \boldsymbol{p}_{m_{i+1}} - \boldsymbol{p}_{C_i}, \tag{7.58}$$

where $\boldsymbol{p}_{C_i}$ denotes the position vector of the centre of mass of augmented Link i.

Substituting (7.55) into (7.53) gives

$$\begin{aligned}\mathcal{T}_{\ell_i} = {} & \frac{1}{2}m_{\ell i}\dot{\boldsymbol{p}}_{C_i}^T\dot{\boldsymbol{p}}_{C_i} + \dot{\boldsymbol{p}}_{C_i}^T\boldsymbol{S}(\boldsymbol{\omega}_i)m_{\ell_i}\boldsymbol{r}_{C_i,\ell_i} \\ & + \frac{1}{2}m_{\ell_i}\boldsymbol{\omega}_i^T\boldsymbol{S}^T(\boldsymbol{r}_{C_i,\ell_i})\boldsymbol{S}(\boldsymbol{r}_{C_i,\ell_i})\boldsymbol{\omega}_i + \frac{1}{2}\boldsymbol{\omega}_i^T\boldsymbol{I}_{\ell_i}\boldsymbol{\omega}_i.\end{aligned} \tag{7.59}$$

By virtue of *Steiner theorem*, the matrix

$$\bar{\boldsymbol{I}}_{\ell_i} = \boldsymbol{I}_{\ell_i} + m_{\ell_i}\boldsymbol{S}^T(\boldsymbol{r}_{C_i,\ell_i})\boldsymbol{S}(\boldsymbol{r}_{C_i,\ell_i}) \tag{7.60}$$

represents the inertia tensor relative to the overall centre of mass $\boldsymbol{p}_{C_i}$, which contains an additional contribution due to the translation of the pole with respect to which the tensor is evaluated, as in (7.57). Therefore, (7.59) can be written as

$$\mathcal{T}_{\ell_i} = \frac{1}{2}m_{\ell i}\dot{\boldsymbol{p}}_{C_i}^T\dot{\boldsymbol{p}}_{C_i} + \dot{\boldsymbol{p}}_{C_i}^T\boldsymbol{S}(\boldsymbol{\omega}_i)m_{\ell_i}\boldsymbol{r}_{C_i,\ell_i} + \frac{1}{2}\boldsymbol{\omega}_i^T\bar{\boldsymbol{I}}_{\ell_i}\boldsymbol{\omega}_i. \tag{7.61}$$

In a similar fashion, substituting (7.56) into (7.54) and exploiting (7.23) yields

$$\begin{aligned}\mathcal{T}_{m_{i+1}} = {} & \frac{1}{2}m_{m_{i+1}}\dot{\boldsymbol{p}}_{C_i}^T\dot{\boldsymbol{p}}_{C_i} + \dot{\boldsymbol{p}}_{C_i}^T\boldsymbol{S}(\boldsymbol{\omega}_i)m_{m_{i+1}}\boldsymbol{r}_{C_i,m_{i+1}} + \frac{1}{2}\boldsymbol{\omega}_i^T\bar{\boldsymbol{I}}_{m_{i+1}}\boldsymbol{\omega}_i \\ & + k_{r,i+1}\dot{q}_{i+1}\boldsymbol{z}_{m_{i+1}}^T\boldsymbol{I}_{m_{i+1}}\boldsymbol{\omega}_i + \frac{1}{2}k_{r,i+1}^2\dot{q}_{i+1}^2\boldsymbol{z}_{m_{i+1}}^T\boldsymbol{I}_{m_{i+1}}\boldsymbol{z}_{m_{i+1}},\end{aligned} \tag{7.62}$$

where

$$\bar{\boldsymbol{I}}_{m_{i+1}} = \boldsymbol{I}_{m_{i+1}} + m_{m_{i+1}}\boldsymbol{S}^T(\boldsymbol{r}_{C_i,m_{i+1}})\boldsymbol{S}(\boldsymbol{r}_{C_i,m_{i+1}}). \tag{7.63}$$

Summing the contributions in (7.61), (7.62) as in (7.52) gives the expression of the kinetic energy of augmented Link i in the form

$$\mathcal{T}_i = \frac{1}{2} m_i \dot{\boldsymbol{p}}_{C_i}^T \dot{\boldsymbol{p}}_{C_i} + \frac{1}{2} \boldsymbol{\omega}_i^T \bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i + k_{r,i+1} \dot{q}_{i+1} \boldsymbol{z}_{m_{i+1}}^T \boldsymbol{I}_{m_{i+1}} \boldsymbol{\omega}_i + \frac{1}{2} k_{r,i+1}^2 \dot{q}_{i+1}^2 \boldsymbol{z}_{m_{i+1}}^T \boldsymbol{I}_{m_{i+1}} \boldsymbol{z}_{m_{i+1}}, \tag{7.64}$$

where $m_i = m_{\ell_i} + m_{m_{i+1}}$ and $\bar{\boldsymbol{I}}_i = \bar{\boldsymbol{I}}_{\ell_i} + \bar{\boldsymbol{I}}_{m_{i+1}}$ are respectively the overall mass and inertia tensor. In deriving (7.64), the relations in (7.57), (7.58) have been utilized as well as the following relation between the positions of the centres of mass:

$$m_{\ell_i} \boldsymbol{p}_{\ell_i} + m_{m_{i+1}} \boldsymbol{p}_{m_{i+1}} = m_i \boldsymbol{p}_{C_i}. \tag{7.65}$$

Notice that the first two terms on the right-hand side of (7.64) represent the kinetic energy contribution of the rotor when this is still, whereas the remaining two terms account for the rotor's own motion.

On the assumption that the rotor has a symmetric mass distribution about its axis of rotation, its inertia tensor expressed in a frame $\boldsymbol{R}_{m_i}$ with origin at the centre of mass and axis z_{m_i} aligned with the rotation axis can be written as

$$\boldsymbol{I}_{m_i}^{m_i} = \begin{bmatrix} I_{m_i xx} & 0 & 0 \\ 0 & I_{m_i yy} & 0 \\ 0 & 0 & I_{m_i zz} \end{bmatrix} \tag{7.66}$$

where $I_{m_i yy} = I_{m_i xx}$. As a consequence, the inertia tensor is invariant with respect to any rotation about axis z_{m_i} and is, anyhow, constant when referred to any frame attached to Link $i-1$.

Since the aim is to determine a set of dynamic parameters independent of the manipulator joint configuration, it is worth referring the inertia tensor of the link $\bar{\boldsymbol{I}}_i$ to frame $\boldsymbol{R}_i$ attached to the link and the inertia tensor $\boldsymbol{I}_{m_{i+1}}$ to frame $\boldsymbol{R}_{m_{i+1}}$ so that it is diagonal. In view of (7.66) one has

$$\boldsymbol{I}_{m_{i+1}} \boldsymbol{z}_{m_{i+1}} = \boldsymbol{R}_{m_{i+1}} \boldsymbol{I}_{m_{i+1}}^{m_{i+1}} \boldsymbol{R}_{m_{i+1}}^T \boldsymbol{z}_{m_{i+1}} = I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}} \tag{7.67}$$

where $I_{m_{i+1}} = I_{m_{i+1} zz}$ denotes the constant scalar moment of inertia of the rotor about its rotation axis.

Therefore, the kinetic energy (7.64) becomes

$$\mathcal{T}_i = \frac{1}{2} m_i \dot{\boldsymbol{p}}_{C_i}^{iT} \dot{\boldsymbol{p}}_{C_i}^i + \frac{1}{2} \boldsymbol{\omega}_i^{iT} \bar{\boldsymbol{I}}_i^i \boldsymbol{\omega}_i^i + k_{r,i+1} \dot{q}_{i+1} I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}}^{iT} \boldsymbol{\omega}_i^i + \frac{1}{2} k_{r,i+1}^2 \dot{q}_{i+1}^2 I_{m_{i+1}}. \tag{7.68}$$

According to the linear velocity composition rule for Link i in (3.15), one may write

$$\dot{\boldsymbol{p}}_{C_i}^i = \dot{\boldsymbol{p}}_i^i + \boldsymbol{\omega}_i^i \times \boldsymbol{r}_{i,C_i}^i, \tag{7.69}$$

where all the vectors have been referred to Frame i; note that $\boldsymbol{r}_{i,C_i}^i$ is fixed in such a frame. Substituting (7.69) into (7.68) gives

$$\mathcal{T}_i = \frac{1}{2} m_i \dot{\boldsymbol{p}}_i^{iT} \dot{\boldsymbol{p}}_i^i + \dot{\boldsymbol{p}}_i^{iT} \boldsymbol{S}(\boldsymbol{\omega}_i^i) m_i \boldsymbol{r}_{i,C_i}^i + \frac{1}{2} \boldsymbol{\omega}_i^{iT} \widehat{\boldsymbol{I}}_i^i \boldsymbol{\omega}_i^i + k_{r,i+1} \dot{q}_{i+1} I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}}^{iT} \boldsymbol{\omega}_i^i + \frac{1}{2} k_{r,i+1}^2 \dot{q}_{i+1}^2 I_{m_{i+1}}, \tag{7.70}$$

where

$$\widehat{\boldsymbol{I}}_i^i = \bar{\boldsymbol{I}}_i^i + m_i \boldsymbol{S}^T(\boldsymbol{r}_{i,C_i}^i) \boldsymbol{S}(\boldsymbol{r}_{i,C_i}^i) \tag{7.71}$$

represents the inertia tensor with respect to the origin of Frame i according to Steiner theorem.

Let $\boldsymbol{r}_{i,C_i}^i = [\, \ell_{C_i x} \quad \ell_{C_i y} \quad \ell_{C_i z} \,]^T$. The *first moment of inertia* is

$$m_i \boldsymbol{r}_{i,C_i}^i = \begin{bmatrix} m_i \ell_{C_i x} \\ m_i \ell_{C_i y} \\ m_i \ell_{C_i z} \end{bmatrix}. \tag{7.72}$$

From (7.71) the inertia tensor of augmented Link i is

$$\begin{aligned} \widehat{\boldsymbol{I}}_i^i &= \begin{bmatrix} \bar{I}_{ixx} + m_i(\ell_{C_i y}^2 + \ell_{C_i z}^2) & -\bar{I}_{ixy} - m_i \ell_{C_i x} \ell_{C_i y} & -\bar{I}_{ixz} - m_i \ell_{C_i x} \ell_{C_i z} \\ * & \bar{I}_{iyy} + m_i(\ell_{C_i x}^2 + \ell_{C_i z}^2) & -\bar{I}_{iyz} - m_i \ell_{C_i y} \ell_{C_i z} \\ * & * & \bar{I}_{izz} + m_i(\ell_{C_i x}^2 + \ell_{C_i y}^2) \end{bmatrix} \\ &= \begin{bmatrix} \widehat{I}_{ixx} & -\widehat{I}_{ixy} & -\widehat{I}_{ixz} \\ * & \widehat{I}_{iyy} & -\widehat{I}_{iyz} \\ * & * & \widehat{I}_{izz} \end{bmatrix}. \end{aligned} \tag{7.73}$$

Therefore, the kinetic energy of the augmented link is linear with respect to the dynamic parameters, namely, the *mass*, the *three components of the first moment of inertia* in (7.72), the *six components of the inertia tensor* in (7.73), and the *moment of inertia of the rotor*.

As regards potential energy, it is worth referring to the centre of mass of augmented Link i defined as in (7.65), and thus the single contribution of potential energy can be written as

$$\mathcal{U}_i = -m_i \boldsymbol{g}_0^{iT} \boldsymbol{p}_{C_i}^i \tag{7.74}$$

where the vectors have been referred to Frame i. According to the relation

$$\boldsymbol{p}_{C_i}^i = \boldsymbol{p}_i^i + \boldsymbol{r}_{i,C_i}^i.$$

The expression in (7.74) can be rewritten as

$$\mathcal{U}_i = -\boldsymbol{g}_0^{iT} (m_i \boldsymbol{p}_i^i + m_i \boldsymbol{r}_{i,C_i}^i) \tag{7.75}$$

that is, the potential energy of the augmented link is linear with respect to the mass and the three components of the first moment of inertia in (7.72).

By summing the contributions of kinetic energy and potential energy for all augmented links, the Lagrangian of the system (7.1) can be expressed in the form

$$\mathcal{L} = \sum_{i=1}^{n} (\boldsymbol{\beta}_{\mathcal{T}i}^T - \boldsymbol{\beta}_{\mathcal{U}i}^T)\boldsymbol{\pi}_i \tag{7.76}$$

where $\boldsymbol{\pi}_i$ is the (11×1) vector of dynamic parameters

$$\boldsymbol{\pi}_i = [\, m_i \;\; m_i\ell_{C_ix} \;\; m_i\ell_{C_iy} \;\; m_i\ell_{C_iz} \;\; \widehat{I}_{ixx} \;\; \widehat{I}_{ixy} \;\; \widehat{I}_{ixz} \;\; \widehat{I}_{iyy} \;\; \widehat{I}_{iyz} \;\; \widehat{I}_{izz} \;\; I_{m_i}\,]^T, \tag{7.77}$$

in which the moment of inertia of Rotor i has been associated with the parameters of Link i so as to simplify the notation.

In (7.76), $\boldsymbol{\beta}_{\mathcal{T}i}$ and $\boldsymbol{\beta}_{\mathcal{U}i}$ are two (11×1) vectors that allow the Lagrangian to be written as a function of $\boldsymbol{\pi}_i$. Such vectors are a function of the generalized coordinates of the mechanical system (and also of their derivatives as regards $\boldsymbol{\beta}_{\mathcal{T}i}$). In particular, it can be shown that $\boldsymbol{\beta}_{\mathcal{T}i} = \boldsymbol{\beta}_{\mathcal{T}i}(q_1, q_2, \ldots, q_i, \dot{q}_1, \dot{q}_2, \ldots, \dot{q}_i)$ and $\boldsymbol{\beta}_{\mathcal{U}i} = \boldsymbol{\beta}_{\mathcal{U}i}(q_1, q_2, \ldots, q_i)$, i.e., they do not depend on the variables of the joints subsequent to Link i.

At this point, it should be observed how the derivations required by the Lagrange equations in (7.2) do not alter the property of linearity in the parameters, and then the generalized force at Joint i can be written as

$$\xi_i = \sum_{j=1}^{n} \boldsymbol{y}_{ij}^T \boldsymbol{\pi}_j \tag{7.78}$$

where

$$\boldsymbol{y}_{ij} = \frac{d}{dt}\frac{\partial \boldsymbol{\beta}_{\mathcal{T}j}}{\partial \dot{q}_i} - \frac{\partial \boldsymbol{\beta}_{\mathcal{T}j}}{\partial q_i} + \frac{\partial \boldsymbol{\beta}_{\mathcal{U}j}}{\partial q_i}. \tag{7.79}$$

Since the partial derivatives of $\boldsymbol{\beta}_{\mathcal{T}j}$ and $\boldsymbol{\beta}_{\mathcal{U}j}$ appearing in (7.79) vanish for $j < i$, the following notable result is obtained:

$$\begin{bmatrix} \xi_1 \\ \xi_2 \\ \vdots \\ \xi_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_{11}^T & \boldsymbol{y}_{12}^T & \cdots & \boldsymbol{y}_{1n}^T \\ \boldsymbol{0}^T & \boldsymbol{y}_{22}^T & \cdots & \boldsymbol{y}_{2n}^T \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0}^T & \boldsymbol{0}^T & \cdots & \boldsymbol{y}_{nn}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\pi}_1 \\ \boldsymbol{\pi}_2 \\ \vdots \\ \boldsymbol{\pi}_n \end{bmatrix} \tag{7.80}$$

which yields the property of *linearity of the model* of a manipulator with respect to a suitable set of *dynamic parameters*.

In the simple case of no contact forces ($\boldsymbol{h}_e = \boldsymbol{0}$), it may be worth including the viscous friction coefficient F_{vi} and Coulomb friction coefficient F_{si} in the parameters of the vector $\boldsymbol{\pi}_i$, thus leading to a total number of 13 parameters per joint. In summary, (7.80) can be compactly written as

$$\boldsymbol{\tau} = \boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\boldsymbol{\pi} \tag{7.81}$$

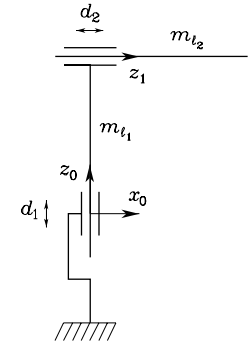

Fig. 7.3. Two-link Cartesian arm

where $\boldsymbol{\pi}$ is a $(p \times 1)$ vector of *constant* parameters and $\boldsymbol{Y}$ is an $(n \times p)$ matrix which is a *function of joint positions, velocities and accelerations*; this matrix is usually called *regressor*. Regarding the dimension of the parameter vector, notice that $p \leq 13n$ since not all the thirteen parameters for each joint may explicitly appear in (7.81).

7.3 Dynamic Model of Simple Manipulator Structures

In the following, three examples of dynamic model computation are illustrated for simple two-DOF manipulator structures. Two DOFs, in fact, are enough to understand the physical meaning of all dynamic terms, especially the joint coupling terms. On the other hand, dynamic model computation for manipulators with more DOFs would be quite tedious and prone to errors, when carried out by paper and pencil. In those cases, it is advisable to perform it with the aid of a symbolic programming software package.

7.3.1 Two-link Cartesian Arm

Consider the two-link Cartesian arm in Fig. 7.3, for which the vector of generalized coordinates is $\boldsymbol{q} = [\, d_1 \quad d_2\,]^T$. Let m_{ℓ_1}, m_{ℓ_2} be the masses of the two links, and m_{m_1}, m_{m_2} the masses of the rotors of the two joint motors. Also let I_{m_1}, I_{m_2} be the moments of inertia with respect to the axes of the two rotors. It is assumed that $\boldsymbol{p}_{m_i} = \boldsymbol{p}_{i-1}$ and $\boldsymbol{z}_{m_i} = \boldsymbol{z}_{i-1}$, for $i = 1, 2$, i.e., the motors are located on the joint axes with centres of mass located at the origins of the respective frames.

With the chosen coordinate frames, computation of the Jacobians in (7.16), (7.18) yields

$$\boldsymbol{J}_P^{(\ell_1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \qquad \boldsymbol{J}_P^{(\ell_2)} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

Obviously, in this case there are no angular velocity contributions for both links.

Computation of the Jacobians in (7.25), (7.26) e (7.28), (7.29) yields

$$\boldsymbol{J}_P^{(m_1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \boldsymbol{J}_P^{(m_2)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}$$

$$\boldsymbol{J}_O^{(m_1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ k_{r1} & 0 \end{bmatrix} \qquad \boldsymbol{J}_O^{(m_2)} = \begin{bmatrix} 0 & k_{r2} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

where k_{ri} is the gear reduction ratio of Motor i. It is obvious to see that $\boldsymbol{z}_1 = [\,1 \quad 0 \quad 0\,]^T$, which greatly simplifies computation of the second term in (4.34).

From (7.32), the inertia matrix is

$$\boldsymbol{B} = \begin{bmatrix} m_{\ell_1} + m_{m_2} + k_{r1}^2 I_{m_1} + m_{\ell_2} & 0 \\ 0 & m_{\ell_2} + k_{r2}^2 I_{m_2} \end{bmatrix}.$$

It is worth observing that $\boldsymbol{B}$ is *constant*, i.e., it does not depend on the arm configuration. This implies also that $\boldsymbol{C} = \boldsymbol{O}$, i.e., there are no contributions of centrifugal and Coriolis forces. As for the gravitational terms, since $\boldsymbol{g}_0 = [\,0 \quad 0 \quad -g\,]^T$ (g is gravity acceleration), (7.39) with the above Jacobians gives

$$g_1 = (m_{\ell_1} + m_{m_2} + m_{\ell_2})g \qquad g_2 = 0.$$

In the absence of friction and tip contact forces, the resulting equations of motion are

$$\begin{aligned} (m_{\ell_1} + m_{m_2} + k_{r1}^2 I_{m_1} + m_{\ell_2})\ddot{d}_1 + (m_{\ell_1} + m_{m_2} + m_{\ell_2})g &= \tau_1 \\ (m_{\ell_2} + k_{r2}^2 I_{m_2})\ddot{d}_2 &= \tau_2 \end{aligned}$$

where τ_1 and τ_2 denote the forces applied to the two joints. Notice that a completely decoupled dynamics has been obtained. This is a consequence not only of the Cartesian structures but also of the particular geometry; in other words, if the second joint axis were not at a right angle with the first joint axis, the resulting inertia matrix would not be diagonal (see Problem 7.1).

7.3.2 Two-link Planar Arm

Consider the two-link planar arm in Fig. 7.4, for which the vector of generalized coordinates is $\boldsymbol{q} = [\,\vartheta_1 \quad \vartheta_2\,]^T$. Let ℓ_1, ℓ_2 be the distances of the centres of mass of the two links from the respective joint axes. Also let m_{ℓ_1}, m_{ℓ_2} be the masses of the two links, and m_{m_1}, m_{m_2} the masses of the rotors of the two joint motors. Finally, let I_{m_1}, I_{m_2} be the moments of inertia with respect to the axes of the two rotors, and I_{ℓ_1}, I_{ℓ_2} the moments of inertia relative to the

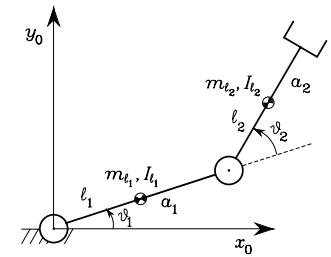

Fig. 7.4. Two-link planar arm

centres of mass of the two links, respectively. It is assumed that $\boldsymbol{p}_{m_i} = \boldsymbol{p}_{i-1}$ and $\boldsymbol{z}_{m_i} = \boldsymbol{z}_{i-1}$, for $i = 1, 2$, i.e., the motors are located on the joint axes with centres of mass located at the origins of the respective frames.

With the chosen coordinate frames, computation of the Jacobians in (7.16), (7.18) yields

$$\boldsymbol{J}_P^{(\ell_1)} = \begin{bmatrix} -\ell_1 s_1 & 0 \\ \ell_1 c_1 & 0 \\ 0 & 0 \end{bmatrix} \qquad \boldsymbol{J}_P^{(\ell_2)} = \begin{bmatrix} -a_1 s_1 - \ell_2 s_{12} & -\ell_2 s_{12} \\ a_1 c_1 + \ell_2 c_{12} & \ell_2 c_{12} \\ 0 & 0 \end{bmatrix},$$

whereas computation of the Jacobians in (7.17), (7.19) yields

$$\boldsymbol{J}_O^{(\ell_1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \qquad \boldsymbol{J}_O^{(\ell_2)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}.$$

Notice that $\boldsymbol{\omega}_i$, for $i = 1, 2$, is aligned with z_0, and thus $\boldsymbol{R}_i$ has *no* effect. It is then possible to refer to the scalar moments of inertia I_{ℓ_i}.

Computation of the Jacobians in (7.25), (7.26) yields

$$\boldsymbol{J}_P^{(m_1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \boldsymbol{J}_P^{(m_2)} = \begin{bmatrix} -a_1 s_1 & 0 \\ a_1 c_1 & 0 \\ 0 & 0 \end{bmatrix},$$

whereas computation of the Jacobians in (7.28), (7.29) yields

$$\boldsymbol{J}_O^{(m_1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ k_{r1} & 0 \end{bmatrix} \qquad \boldsymbol{J}_O^{(m_2)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & k_{r2} \end{bmatrix}$$

where k_{ri} is the gear reduction ratio of Motor i.

From (7.32), the inertia matrix is

$$\boldsymbol{B}(\boldsymbol{q}) = \begin{bmatrix} b_{11}(\vartheta_2) & b_{12}(\vartheta_2) \\ b_{21}(\vartheta_2) & b_{22} \end{bmatrix}$$

$$
\begin{aligned}
b_{11} &= I_{\ell_1} + m_{\ell_1}\ell_1^2 + k_{r1}^2 I_{m_1} + I_{\ell_2} + m_{\ell_2}(a_1^2 + \ell_2^2 + 2a_1\ell_2 c_2) \\
&\quad + I_{m_2} + m_{m_2} a_1^2 \\
b_{12} &= b_{21} = I_{\ell_2} + m_{\ell_2}(\ell_2^2 + a_1\ell_2 c_2) + k_{r2} I_{m_2} \\
b_{22} &= I_{\ell_2} + m_{\ell_2}\ell_2^2 + k_{r2}^2 I_{m_2}.
\end{aligned}
$$

Compared to the previous example, the inertia matrix is now configuration-dependent. Notice that the term $k_{r2}I_{m_2}$ in the off-diagonal term of the inertia matrix derives from having considered the rotational part of the motor kinetic energy as due to the total angular velocity, i.e., its own angular velocity and that of the preceding link in the kinematic chain. At first approximation, especially in the case of high values of the gear reduction ratio, this contribution could be neglected; in the resulting reduced model, motor inertias would appear uniquely in the elements on the diagonal of the inertia matrix with terms of the type $k_{ri}^2 I_{m_i}$.

The computation of Christoffel symbols as in (7.45) gives

$$
\begin{aligned}
c_{111} &= \frac{1}{2}\frac{\partial b_{11}}{\partial q_1} = 0 \\
c_{112} = c_{121} &= \frac{1}{2}\frac{\partial b_{11}}{\partial q_2} = -m_{\ell_2} a_1 \ell_2 s_2 = h \\
c_{122} &= \frac{\partial b_{12}}{\partial q_2} - \frac{1}{2}\frac{\partial b_{22}}{\partial q_1} = h \\
c_{211} &= \frac{\partial b_{21}}{\partial q_1} - \frac{1}{2}\frac{\partial b_{11}}{\partial q_2} = -h \\
c_{212} = c_{221} &= \frac{1}{2}\frac{\partial b_{22}}{\partial q_1} = 0 \\
c_{222} &= \frac{1}{2}\frac{\partial b_{22}}{\partial q_2} = 0,
\end{aligned}
$$

leading to the matrix

$$
\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \begin{bmatrix} h\dot{\vartheta}_2 & h(\dot{\vartheta}_1 + \dot{\vartheta}_2) \\ -h\dot{\vartheta}_1 & 0 \end{bmatrix}.
$$

Computing the matrix $\boldsymbol{N}$ in (7.47) gives

$$
\begin{aligned}
\boldsymbol{N}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \dot{\boldsymbol{B}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\
&= \begin{bmatrix} 2h\dot{\vartheta}_2 & h\dot{\vartheta}_2 \\ h\dot{\vartheta}_2 & 0 \end{bmatrix} - 2\begin{bmatrix} h\dot{\vartheta}_2 & h(\dot{\vartheta}_1 + \dot{\vartheta}_2) \\ -h\dot{\vartheta}_1 & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & -2h\dot{\vartheta}_1 - h\dot{\vartheta}_2 \\ 2h\dot{\vartheta}_1 + h\dot{\vartheta}_2 & 0 \end{bmatrix}
\end{aligned}
$$

that allows the verification of the skew-symmetry property expressed by (7.48). See also Problem 7.2.

As for the gravitational terms, since $\boldsymbol{g}_0 = [\,0 \quad -g \quad 0\,]^T$, (7.39) with the above Jacobians gives

$$
\begin{aligned}
g_1 &= (m_{\ell_1}\ell_1 + m_{m_2} a_1 + m_{\ell_2} a_1) g c_1 + m_{\ell_2}\ell_2 g c_{12} \\
g_2 &= m_{\ell_2}\ell_2 g c_{12}.
\end{aligned}
$$

In the absence of friction and tip contact forces, the resulting equations of motion are

$$
\begin{aligned}
&\left(I_{\ell_1} + m_{\ell_1}\ell_1^2 + k_{r1}^2 I_{m_1} + I_{\ell_2} + m_{\ell_2}(a_1^2 + \ell_2^2 + 2a_1\ell_2 c_2) + I_{m_2} + m_{m_2} a_1^2\right)\ddot{\vartheta}_1 \\
&\quad + \left(I_{\ell_2} + m_{\ell_2}(\ell_2^2 + a_1\ell_2 c_2) + k_{r2} I_{m_2}\right)\ddot{\vartheta}_2 \\
&\quad - 2m_{\ell_2} a_1 \ell_2 s_2 \dot{\vartheta}_1 \dot{\vartheta}_2 - m_{\ell_2} a_1 \ell_2 s_2 \dot{\vartheta}_2^2 \\
&\quad + (m_{\ell_1}\ell_1 + m_{m_2} a_1 + m_{\ell_2} a_1) g c_1 + m_{\ell_2}\ell_2 g c_{12} = \tau_1 \qquad (7.82) \\
&\left(I_{\ell_2} + m_{\ell_2}(\ell_2^2 + a_1\ell_2 c_2) + k_{r2} I_{m_2}\right)\ddot{\vartheta}_1 + \left(I_{\ell_2} + m_{\ell_2}\ell_2^2 + k_{r2}^2 I_{m_2}\right)\ddot{\vartheta}_2 \\
&\quad + m_{\ell_2} a_1 \ell_2 s_2 \dot{\vartheta}_1^2 + m_{\ell_2}\ell_2 g c_{12} = \tau_2
\end{aligned}
$$

where τ_1 and τ_2 denote the torques applied to the joints.

Finally, it is wished to derive a parameterization of the dynamic model (7.82) according to the relation (7.81). By direct inspection of the expressions of the joint torques, it is possible to find the following parameter vector:

$$
\boldsymbol{\pi} = [\,\pi_1 \quad \pi_2 \quad \pi_3 \quad \pi_4 \quad \pi_5 \quad \pi_6 \quad \pi_7 \quad \pi_8\,]^T \qquad (7.83)
$$

$$
\begin{aligned}
\pi_1 &= m_1 = m_{\ell_1} + m_{m_2} \\
\pi_2 &= m_1 \ell_{C_1} = m_{\ell_1}(\ell_1 - a_1) \\
\pi_3 &= \widehat{I}_1 = I_{\ell_1} + m_{\ell_1}(\ell_1 - a_1)^2 + I_{m_2} \\
\pi_4 &= I_{m_1} \\
\pi_5 &= m_2 = m_{\ell_2} \\
\pi_6 &= m_2 \ell_{C_2} = m_{\ell_2}(\ell_2 - a_2) \\
\pi_7 &= \widehat{I}_2 = I_{\ell_2} + m_{\ell_2}(\ell_2 - a_2)^2 \\
\pi_8 &= I_{m_2},
\end{aligned}
$$

where the parameters for the augmented links have been found according to (7.77). It can be recognized that the number of non-null parameters is less than the maximum number of twenty-two parameters allowed in this case.[6] The regressor in (7.81) is

$$
\boldsymbol{Y} = \begin{bmatrix} y_{11} & y_{12} & y_{13} & y_{14} & y_{15} & y_{16} & y_{17} & y_{18} \\ y_{21} & y_{22} & y_{23} & y_{24} & y_{25} & y_{26} & y_{27} & y_{28} \end{bmatrix} \qquad (7.84)
$$

[6] The number of parameters can be further reduced by resorting to a more accurate inspection, which leads to finding a minimum number of five parameters; those turn out to be a linear combination of the parameters in (7.83) (see Problem 7.4).

$$
\begin{aligned}
y_{11} &= a_1^2\ddot{\vartheta}_1 + a_1 g c_1 \\
y_{12} &= 2a_1\ddot{\vartheta}_1 + g c_1 \\
y_{13} &= \ddot{\vartheta}_1 \\
y_{14} &= k_{r1}^2\ddot{\vartheta}_1 \\
y_{15} &= (a_1^2 + 2a_1a_2c_2 + a_2^2)\ddot{\vartheta}_1 + (a_1a_2c_2 + a_2^2)\ddot{\vartheta}_2 - 2a_1a_2s_2\dot{\vartheta}_1\dot{\vartheta}_2 \\
&\quad -a_1a_2s_2\dot{\vartheta}_2^2 + a_1 g c_1 + a_2 g c_{12} \\
y_{16} &= (2a_1c_2 + 2a_2)\ddot{\vartheta}_1 + (a_1c_2 + 2a_2)\ddot{\vartheta}_2 - 2a_1s_2\dot{\vartheta}_1\dot{\vartheta}_2 - a_1s_2\dot{\vartheta}_2^2 \\
&\quad +g c_{12} \\
y_{17} &= \ddot{\vartheta}_1 + \ddot{\vartheta}_2 \\
y_{18} &= k_{r2}\ddot{\vartheta}_2
\end{aligned}
$$

$$
\begin{aligned}
y_{21} &= 0 \\
y_{22} &= 0 \\
y_{23} &= 0 \\
y_{24} &= 0 \\
y_{25} &= (a_1a_2c_2 + a_2^2)\ddot{\vartheta}_1 + a_2^2\ddot{\vartheta}_2 + a_1a_2s_2\dot{\vartheta}_1^2 + a_2 g c_{12} \\
y_{26} &= (a_1c_2 + 2a_2)\ddot{\vartheta}_1 + 2a_2\ddot{\vartheta}_2 + a_1s_2\dot{\vartheta}_1^2 + g c_{12} \\
y_{27} &= \ddot{\vartheta}_1 + \ddot{\vartheta}_2 \\
y_{28} &= k_{r2}\ddot{\vartheta}_1 + k_{r2}^2\ddot{\vartheta}_2.
\end{aligned}
$$

Example 7.2

In order to understand the relative weight of the various torque contributions in the dynamic model (7.82), consider a two-link planar arm with the following data:

$$a_1 = a_2 = 1\,\mathrm{m} \quad \ell_1 = \ell_2 = 0.5\,\mathrm{m} \quad m_{\ell_1} = m_{\ell_2} = 50\,\mathrm{kg} \quad I_{\ell_1} = I_{\ell_2} = 10\,\mathrm{kg{\cdot}m^2}$$

$$k_{r1} = k_{r2} = 100 \quad m_{m_1} = m_{m_2} = 5\,\mathrm{kg} \quad I_{m_1} = I_{m_2} = 0.01\,\mathrm{kg{\cdot}m^2}.$$

The two links have been chosen equal to illustrate better the dynamic interaction between the two joints.

Figure 7.5 shows the time history of positions, velocities, accelerations and torques resulting from joint trajectories with typical triangular velocity profile and equal time duration. The initial arm configuration is so that the tip is located at the point $(0.2, 0)$ m with a lower elbow posture. Both joints make a rotation of $\pi/2$ rad in a time of 0.5 s.

From the time history of the single torque contributions in Fig. 7.6 it can be recognized that:

- The inertia torque at Joint 1 due to Joint 1 acceleration follows the time history of the acceleration.
- The inertia torque at Joint 2 due to Joint 2 acceleration is piecewise constant, since the inertia moment at Joint 2 axis is constant.

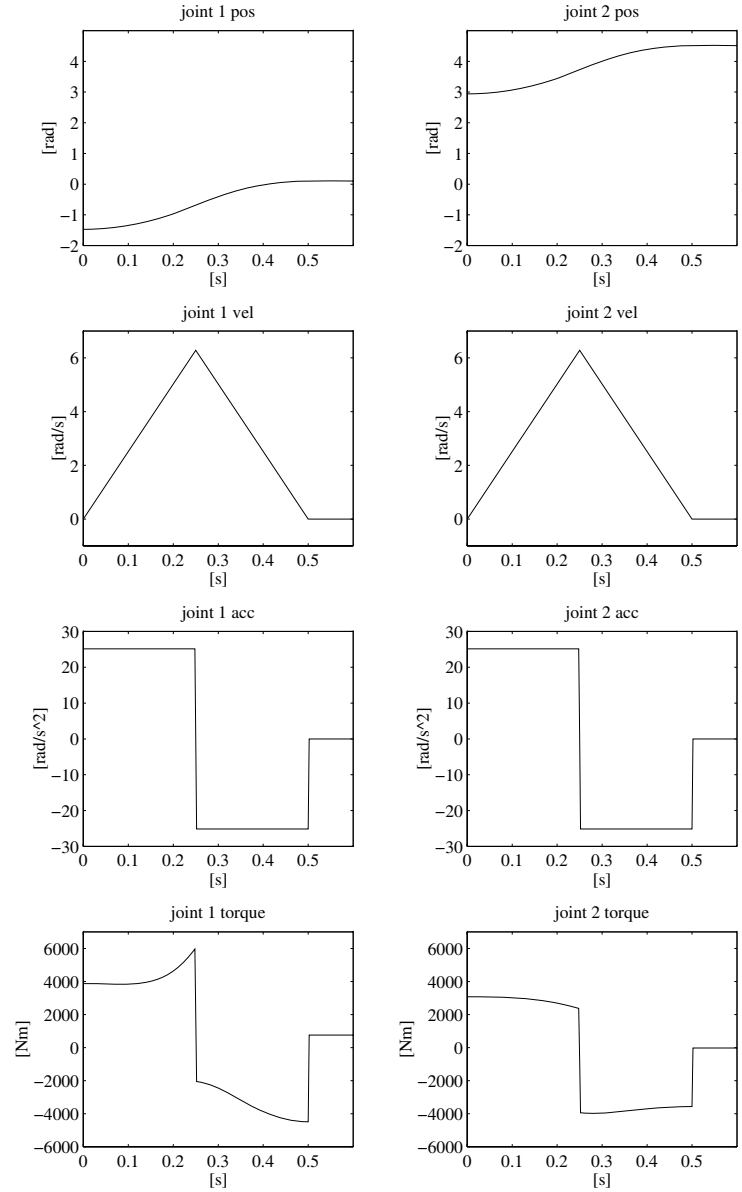

Fig. 7.5. Time history of positions, velocities, accelerations and torques with joint trajectories of equal duration

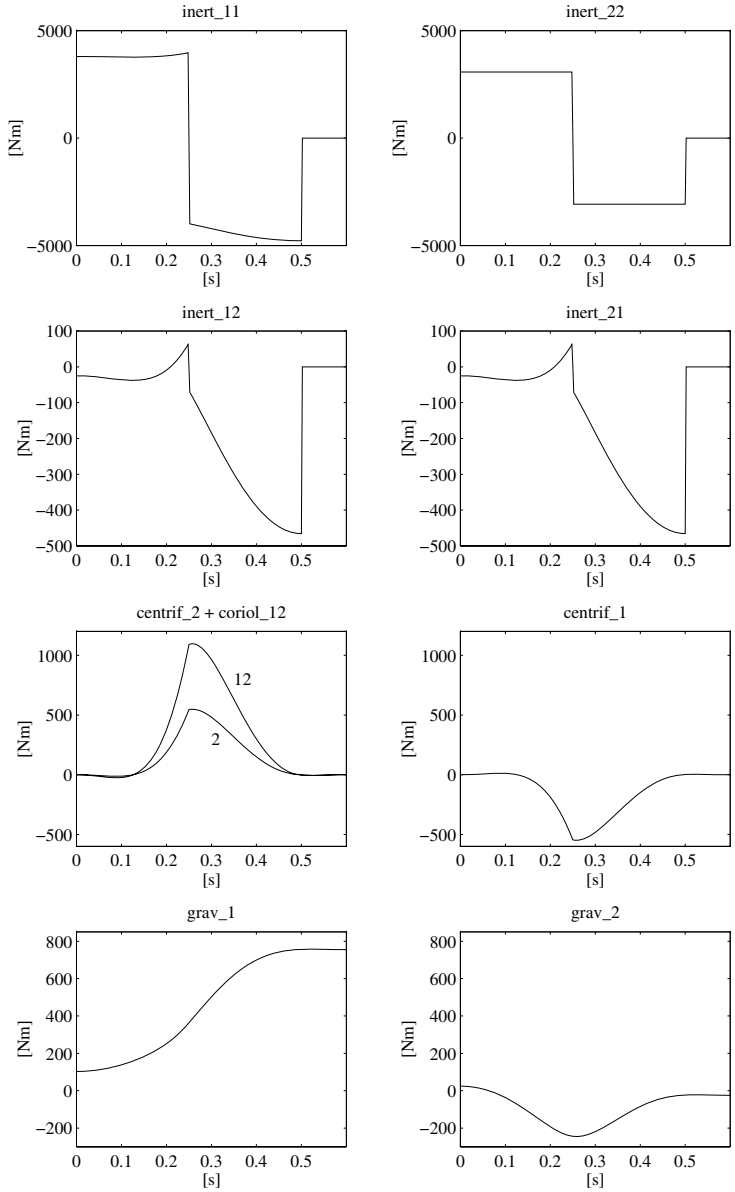

Fig. 7.6. Time history of torque contributions with joint trajectories of equal duration

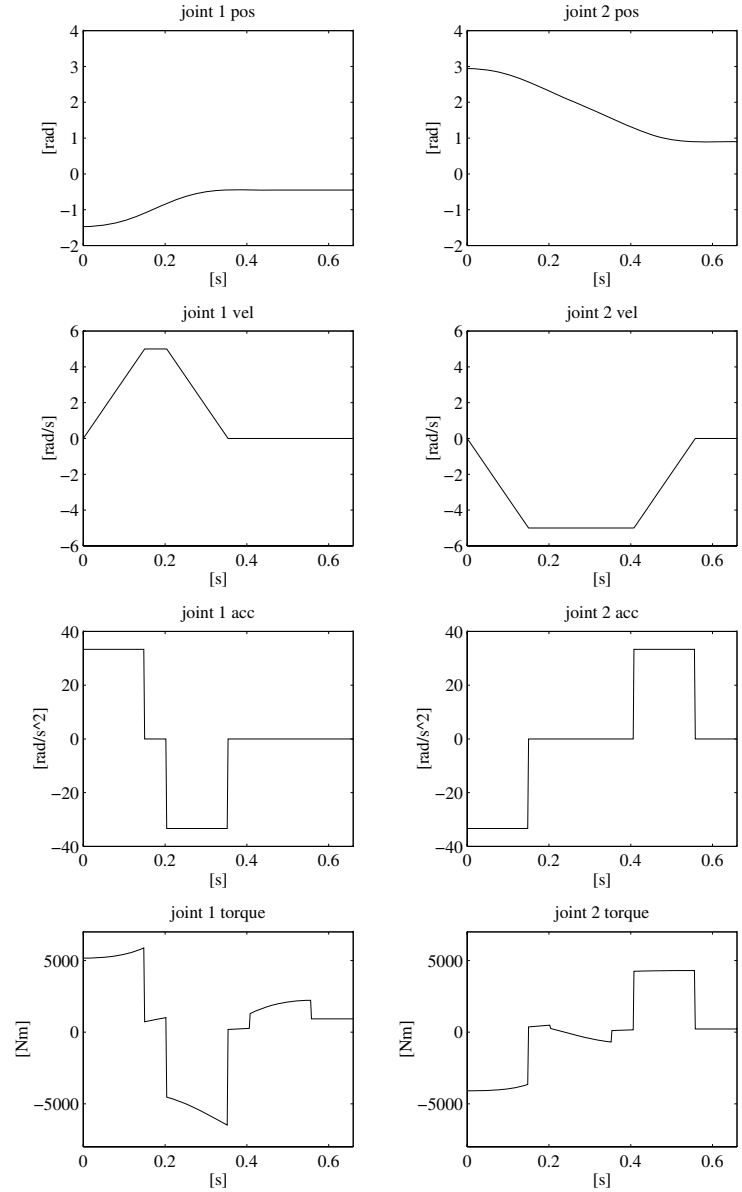

Fig. 7.7. Time history of positions, velocities, accelerations and torques with joint trajectories of different duration

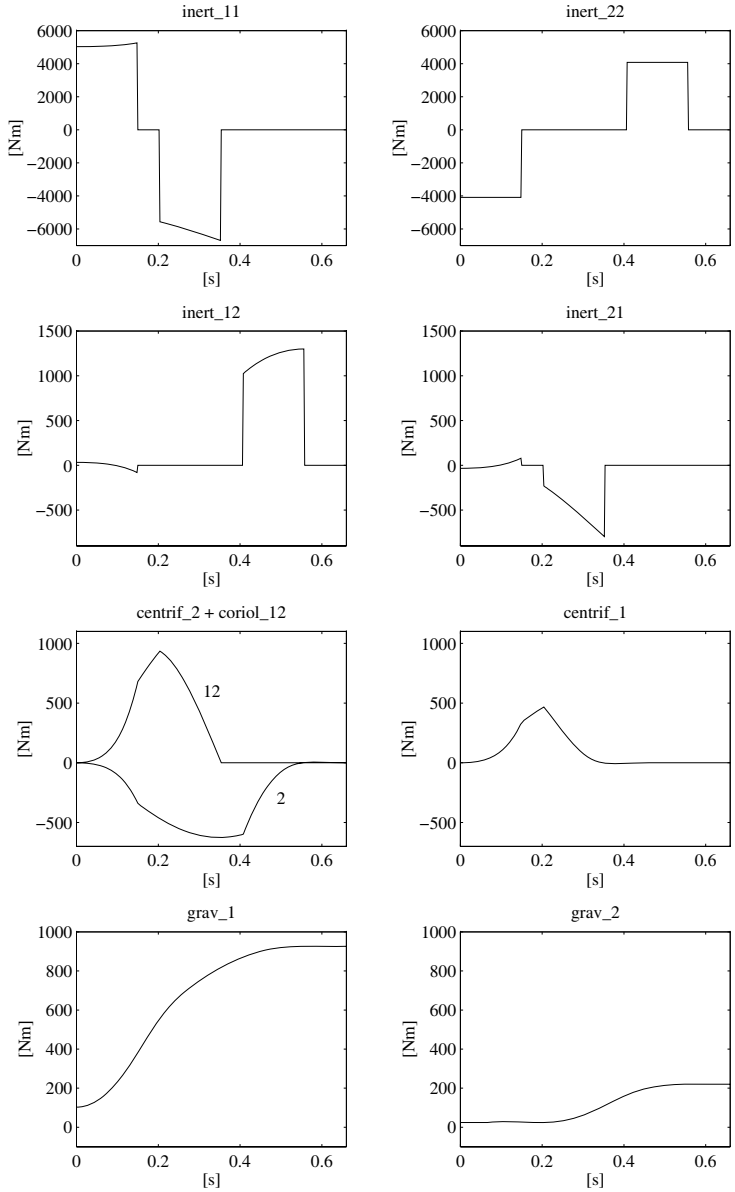

Fig. 7.8. Time history of torque contributions with joint trajectories of different duration

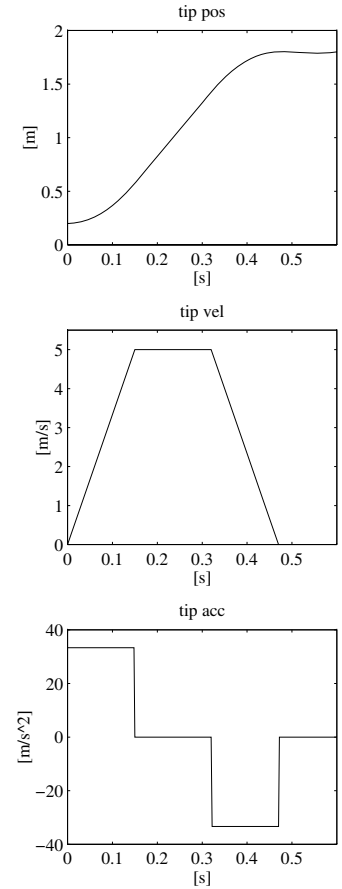

Fig. 7.9. Time history of tip position, velocity and acceleration with a straight line tip trajectory along the horizontal axis

- The inertia torques at each joint due to acceleration of the other joint confirm the symmetry of the inertia matrix, since the acceleration profiles are the same for both joints.
- The Coriolis effect is present only at Joint 1, since the arm tip moves with respect to the mobile frame attached to Link 1 but is fixed with respect to the frame attached to Link 2.
- The centrifugal and Coriolis torques reflect the above symmetry.

Figure 7.7 shows the time history of positions, velocities, accelerations and torques resulting from joint trajectories with typical trapezoidal velocity profile and different time duration. The initial configuration is the same as in the previous case. The two joints make a rotation so as to take the tip to the point $(1.8, 0)$ m. The acceleration time is 0.15 s and the maximum velocity is 5 rad/s for both joints.

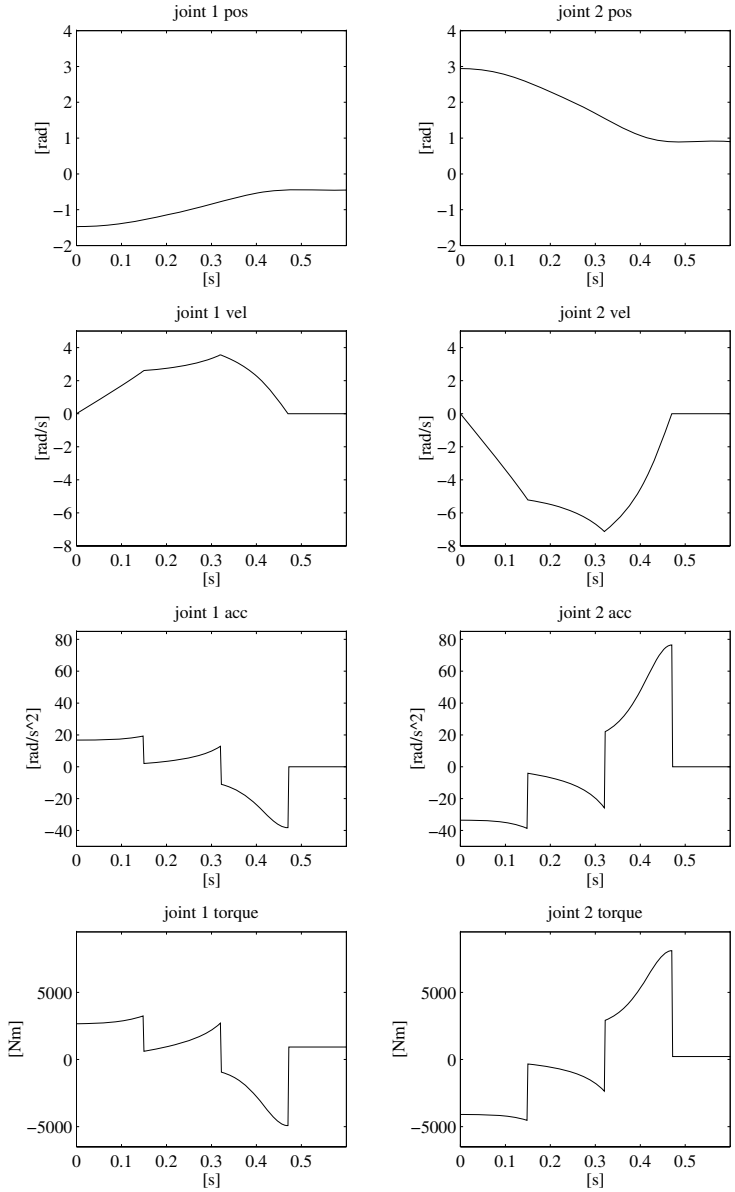

Fig. 7.10. Time history of joint positions, velocities, accelerations, and torques with a straight line tip trajectory along the horizontal axis

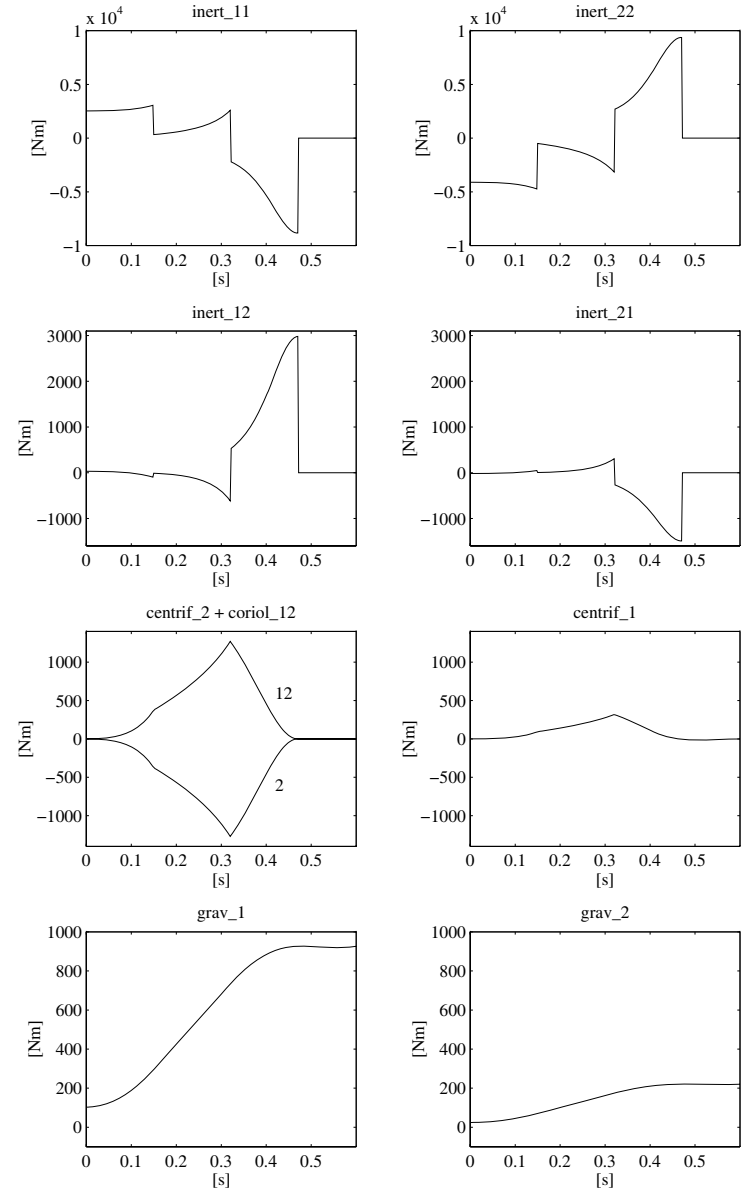

Fig. 7.11. Time history of joint torque contributions with a straight line tip trajectory along the horizontal axis

From the time history of the single torque contributions in Fig. 7.8 it can be recognized that:

- The inertia torque at Joint 1 due to Joint 2 acceleration is opposite to that at Joint 2 due to Joint 1 acceleration in that portion of trajectory when the two accelerations have the same magnitude but opposite sign.
- The different velocity profiles imply that the centrifugal effect induced at Joint 1 by Joint 2 velocity dies out later than the centrifugal effect induced at Joint 2 by Joint 1 velocity.
- The gravitational torque at Joint 2 is practically constant in the first portion of the trajectory, since Link 2 is almost kept in the same posture. As for the gravitational torque at Joint 1, instead, the centre of mass of the articulated system moves away from the origin of the axes.

Finally, Fig. 7.9 shows the time history of tip position, velocity and acceleration for a trajectory with a trapezoidal velocity profile. Starting from the same initial posture as above, the arm tip makes a translation of 1.6 m along the horizontal axis; the acceleration time is 0.15 s and the maximum velocity is 5 m/s.

As a result of an inverse kinematics procedure, the time history of joint positions, velocities and accelerations have been computed which are illustrated in Fig. 7.10, together with the joint torques that are needed to execute the assigned trajectory. It can be noticed that the time history of the represented quantities differs from the corresponding ones in the operational space, in view of the nonlinear effects introduced by kinematic relations.

For what concerns the time history of the individual torque contributions in Fig. 7.11, it is possible to make a number of remarks similar to those made above for trajectories assigned directly in the joint space.

7.3.3 Parallelogram Arm

Consider the parallelogram arm in Fig. 7.12. Because of the presence of the closed chain, the equivalent tree-structured open-chain arm is initially taken into account. Let $\ell_{1'}$, $\ell_{2'}$, $\ell_{3'}$ and $\ell_{1''}$ be the distances of the centres of mass of the three links along one branch of the tree, and of the single link along the other branch, from the respective joint axes. Also let $m_{\ell_{1'}}$, $m_{\ell_{2'}}$, $m_{\ell_{3'}}$ and $m_{\ell_{1''}}$ be the masses of the respective links, and $I_{\ell_{1'}}$, $I_{\ell_{2'}}$, $I_{\ell_{3'}}$ and $I_{\ell_{1''}}$ the moments of inertia relative to the centres of mass of the respective links. For the sake of simplicity, the contributions of the motors are neglected.

With the chosen coordinate frames, computation of the Jacobians in (7.16) (7.18) yields

$$\boldsymbol{J}_P^{(\ell_{1'})} = \begin{bmatrix} -\ell_{1'}s_{1'} & 0 & 0 \\ \ell_{1'}c_{1'} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{J}_P^{(\ell_{2'})} = \begin{bmatrix} -a_{1'}s_{1'} - \ell_2 s_{1'2'} & -\ell_{2'}s_{1'2'} & 0 \\ a_{1'}c_{1'} + \ell_{2'}c_{1'2'} & \ell_2 c_{1'2'} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{J}_P^{(\ell_{3'})} = \begin{bmatrix} -a_{1'}s_{1'} - a_{2'}s_{1'2'} - \ell_{3'}s_{1'2'3'} & -a_{2'}s_{1'2'} - \ell_{3'}s_{1'2'3'} & -\ell_{3'}s_{1'2'3'} \\ a_{1'}c_{1'} + a_{2'}c_{1'2'} + \ell_{3'}c_{1'2'3'} & a_{2'}c_{1'2'} + \ell_{3'}c_{1'2'3'} & \ell_{3'}c_{1'2'3'} \\ 0 & 0 & 0 \end{bmatrix}$$

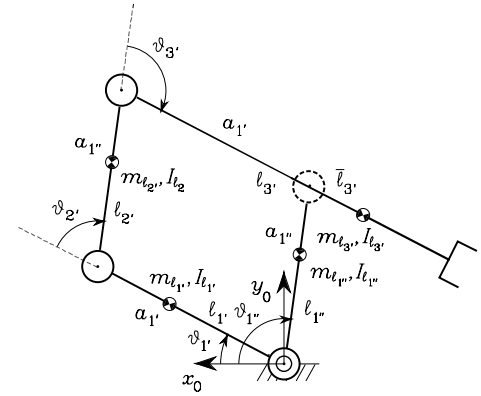

Fig. 7.12. Parallelogram arm

and

$$\boldsymbol{J}_P^{(\ell_{1''})} = \begin{bmatrix} -\ell_{1''}s_{1''} \\ \ell_{1''}c_{1''} \\ 0 \end{bmatrix},$$

whereas computation of the Jacobians in (7.17), (7.19) yields

$$\boldsymbol{J}_O^{(\ell_{1'})} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{J}_O^{(\ell_{2'})} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \qquad \boldsymbol{J}_O^{(\ell_{3'})} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

and

$$\boldsymbol{J}_O^{(\ell_{1''})} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

From (7.32), the inertia matrix of the virtual arm composed of joints $\vartheta_{1'}$, $\vartheta_{2'}$, $\vartheta_{3'}$ is

$$\boldsymbol{B}'(\boldsymbol{q}') = \begin{bmatrix} b_{1'1'}(\vartheta_{2'}, \vartheta_{3'}) & b_{1'2'}(\vartheta_{2'}, \vartheta_{3'}) & b_{1'3'}(\vartheta_{2'}, \vartheta_{3'}) \\ b_{2'1'}(\vartheta_{2'}, \vartheta_{3'}) & b_{2'2'}(\vartheta_{3'}) & b_{2'3'}(\vartheta_{3'}) \\ b_{3'1'}(\vartheta_{2'}, \vartheta_{3'}) & b_{3'2'}(\vartheta_{3'}) & b_{3'3'} \end{bmatrix}$$

$$\begin{aligned}
b_{1'1'} &= I_{\ell_{1'}} + m_{\ell_{1'}}\ell_{1'}^2 + I_{\ell_{2'}} + m_{\ell_{2'}}(a_{1'}^2 + \ell_{2'}^2 + 2a_{1'}\ell_{2'}c_{2'}) + I_{\ell_{3'}} \\
&\quad + m_{\ell_{3'}}(a_{1'}^2 + a_{2'}^2 + \ell_{3'}^2 + 2a_{1'}a_{2'}c_{2'} + 2a_{1'}\ell_{3'}c_{2'3'} + 2a_{2'}\ell_{3'}c_{3'}) \\
b_{1'2'} &= b_{2'1'} = I_{\ell_{2'}} + m_{\ell_{2'}}(\ell_{2'}^2 + a_{1'}\ell_{2'}c_{2'}) + I_{\ell_{3'}} \\
&\quad + m_{\ell_{3'}}(a_{2'}^2 + \ell_{3'}^2 + a_{1'}a_{2'}c_{2'} + a_{1'}\ell_{3'}c_{2'3'} + 2a_{2'}\ell_{3'}c_{3'}) \\
b_{1'3'} &= b_{31} = I_{\ell_{3'}} + m_{\ell_{3'}}(\ell_{3'}^2 + a_{1'}\ell_{3'}c_{2'3'} + a_{2'}\ell_{3'}c_{3'}) \\
b_{2'2'} &= I_{\ell_{2'}} + m_{\ell_{2'}}\ell_{2'}^2 + I_{\ell_{3'}} + m_{\ell_{3'}}(a_{2'}^2 + \ell_{3'}^2 + 2a_{2'}\ell_{3'}c_{3'}) \\
b_{2'3'} &= I_{\ell_{3'}} + m_{\ell_{3'}}(\ell_{3'}^2 + a_{2'}\ell_{3'}c_{3'}) \\
b_{3'3'} &= I_{\ell_{3'}} + m_{\ell_{3'}}\ell_{3'}^2
\end{aligned}$$

while the moment of inertia of the virtual arm composed of just joint $\vartheta_{1''}$ is

$$b_{1''1''} = I_{\ell_{1''}} + m_{\ell_{1''}} \ell_{1''}^2.$$

Therefore, the inertial torque contributions of the two virtual arms are respectively:

$$\tau_{i'} = \sum_{j'=1'}^{3'} b_{i'j'} \ddot{\vartheta}_{j'} \qquad \tau_{1''} = b_{1''1''} \ddot{\vartheta}_{1''}.$$

At this point, in view of (2.64) and (3.121), the inertial torque contributions at the actuated joints for the closed-chain arm turn out to be

$$\boldsymbol{\tau}_a = \boldsymbol{B}_a \ddot{\boldsymbol{q}}_a$$

where $\boldsymbol{q}_a = [\, \vartheta_{1'} \quad \vartheta_{1''} \,]^T$, $\boldsymbol{\tau}_a = [\, \tau_{a1} \quad \tau_{a2} \,]^T$ and

$$\boldsymbol{B}_a = \begin{bmatrix} b_{a11} & b_{a12} \\ b_{a21} & b_{a22} \end{bmatrix}$$

$$\begin{aligned} b_{a11} &= I_{\ell_{1'}} + m_{\ell_{1'}} \ell_{1'}^2 + m_{\ell_{2'}} a_{1'}^2 + I_{\ell_{3'}} + m_{\ell_{3'}} \ell_{3'}^2 + m_{\ell_{3'}} a_{1'}^2 \\ &\quad -2 a_{1'} m_{\ell_{3'}} \ell_{3'} \\ b_{a12} &= b_{a21} = \left(a_{1'} m_{\ell_{2'}} \ell_{2'} + a_{1''} m_{\ell_{3'}} (a_{1'} - \ell_{3'}) \right) \cos(\vartheta_{1''} - \vartheta_{1'}) \\ b_{a22} &= I_{\ell_{1'}} + m_{\ell_{1'}} \ell_{1'}^2 + I_{\ell_{2'}} + m_{\ell_{2'}} \ell_{2'}^2 + m_{\ell_{3'}} a_{1''}^2. \end{aligned}$$

This expression reveals the possibility of obtaining a *configuration-independent* and *decoupled* inertia matrix; to this end it is sufficient to design the four links of the parallelogram so that

$$\frac{m_{\ell_{3'}} \bar{\ell}_{3'}}{m_{\ell_{2'}} \ell_{2'}} = \frac{a_{1'}}{a_{1''}}$$

where $\bar{\ell}_{3'} = \ell_{3'} - a_{1'}$ is the distance of the centre of mass of Link $3'$ from the axis of Joint 4. If this condition is satisfied, then the inertia matrix is diagonal ($b_{a12} = b_{a21} = 0$) with

$$\begin{aligned} b_{a11} &= I_{\ell_{1'}} + m_{\ell_{1'}} \ell_{1'}^2 + m_{\ell_{2'}} a_{1'}^2 \left(1 + \frac{\ell_{2'} \bar{\ell}_{3'}}{a_{1'} a_{1''}} \right) + I_{\ell_{3'}} \\ b_{a22} &= I_{\ell_{1'}} + m_{\ell_{1'}} \ell_{1'}^2 + I_{\ell_{2'}} + m_{\ell_{2'}} \ell_{2'}^2 \left(1 + \frac{a_{1'} a_{1''}}{\ell_{2'} \bar{\ell}_{3'}} \right). \end{aligned}$$

As a consequence, no contributions of Coriolis and centrifugal torques are obtained. Such a result could not be achieved with the previous two-link planar arm, no matter how the design parameters were chosen.

As for the gravitational terms, since $\boldsymbol{g}_0 = [\, 0 \quad -g \quad 0 \,]^T$, (7.39) with the above Jacobians gives

$$\begin{aligned} g_{1'} \;& (m_{\ell_{1'}} \ell_{1'} + m_{\ell_{2'}} a_{1'} + m_{\ell_{3'}} a_{1'}) g c_{1'} + (m_{\ell_{2'}} \ell_{2'} + m_{\ell_{3'}} a_{2'}) g c_{1'2'} \\ & + m_{\ell_{3'}} \ell_{3'} g c_{1'2'3} \\ g_{2'} \;& (m_{\ell_{2'}} \ell_{2'} + m_{\ell_{3'}} a_{2'}) g c_{1'2'} + m_{\ell_{3'}} \ell_{3'} g c_{1'2'3} \\ g_{3'} \;& m_{\ell_{3'}} \ell_{3'} g c_{1'2'3} \end{aligned}$$

and

$$g_{1''} = m_{\ell_{1''}} \ell_{1''} g c_{1''}.$$

Composing the various contributions as done above yields

$$\boldsymbol{g}_a = \begin{bmatrix} (m_{\ell_{1'}} \ell_{1'} + m_{\ell_{2'}} a_{1'} - m_{\ell_{3'}} \bar{\ell}_{3'}) g c_{1'} \\ (m_{\ell_{1''}} \ell_{1''} + m_{\ell_{2'}} \ell_{2'} + m_{\ell_{3'}} a_{1''}) g c_{1''} \end{bmatrix}$$

which, together with the inertial torques, completes the derivation of the sought dynamic model.

A final comment is in order. In spite of its kinematic equivalence with the two-link planar arm, the dynamic model of the parallelogram is remarkably lighter. This property is quite advantageous for trajectory planning and control purposes. For this reason, apart from obvious considerations related to manipulation of heavy payloads, the adoption of closed kinematic chains in the design of industrial robots has received a great deal of attention.

7.4 Dynamic Parameter Identification

The use of the dynamic model for solving simulation and control problems demands the knowledge of the values of dynamic parameters of the manipulator model.

Computing such parameters from the design data of the mechanical structure is not simple. CAD modelling techniques can be adopted which allow the computation of the values of the inertial parameters of the various components (links, actuators and transmissions) on the basis of their geometry and type of materials employed. Nevertheless, the estimates obtained by such techniques are inaccurate because of the simplification typically introduced by geometric modelling; moreover, complex dynamic effects, such as joint friction, cannot be taken into account.

A heuristic approach could be to dismantle the various components of the manipulator and perform a series of measurements to evaluate the inertial parameters. Such technique is not easy to implement and may be troublesome to measure the relevant quantities.

In order to find accurate estimates of dynamic parameters, it is worth resorting to *identification* techniques which conveniently exploit the *property of linearity* (7.81) of the manipulator model with respect to a suitable set of

dynamic parameters. Such techniques allow the computation of the parameter vector $\boldsymbol{\pi}$ from the measurements of joint torques $\boldsymbol{\tau}$ and of relevant quantities for the evaluation of the matrix $\boldsymbol{Y}$, when suitable motion trajectories are imposed to the manipulator.

On the assumption that the kinematic parameters in the matrix $\boldsymbol{Y}$ are known with good accuracy, e.g., as a result of a kinematic calibration, measurements of joint positions $\boldsymbol{q}$, velocities $\dot{\boldsymbol{q}}$ and accelerations $\ddot{\boldsymbol{q}}$ are required. Joint positions and velocities can be actually measured while numerical reconstruction of accelerations is needed; this can be performed on the basis of the position and velocity values recorded during the execution of the trajectories. The reconstructing filter does not work in real time and thus it can also be anti-causal, allowing an accurate reconstruction of the accelerations.

As regards joint torques, in the unusual case of torque sensors at the joint, these can be measured directly. Otherwise, they can be evaluated from either wrist force measurements or current measurements in the case of electric actuators.

If measurements of joint torques, positions, velocities and accelerations have been obtained at given time instants $t_1, \dots, t_N$ along a given trajectory, one may write

$$\bar{\boldsymbol{\tau}} = \begin{bmatrix} \boldsymbol{\tau}(t_1) \\ \vdots \\ \boldsymbol{\tau}(t_N) \end{bmatrix} = \begin{bmatrix} \boldsymbol{Y}(t_1) \\ \vdots \\ \boldsymbol{Y}(t_N) \end{bmatrix} \boldsymbol{\pi} = \bar{\boldsymbol{Y}} \boldsymbol{\pi}. \tag{7.85}$$

The number of time instants sets the number of measurements to perform and should be large enough (typically $Nn \gg p$) so as to avoid ill-conditioning of matrix $\bar{\boldsymbol{Y}}$. Solving (7.85) by a least-squares technique leads to the solution in the form

$$\boldsymbol{\pi} = (\bar{\boldsymbol{Y}}^T \bar{\boldsymbol{Y}})^{-1} \bar{\boldsymbol{Y}}^T \bar{\boldsymbol{\tau}} \tag{7.86}$$

where $(\bar{\boldsymbol{Y}}^T \bar{\boldsymbol{Y}})^{-1} \bar{\boldsymbol{Y}}^T$ is the *left pseudo-inverse* matrix of $\bar{\boldsymbol{Y}}$.

It should be noticed that, in view of the block triangular structure of matrix $\boldsymbol{Y}$ in (7.80), computation of parameter estimates could be simplified by resorting to a sequential procedure. Take the equation $\tau_n = \boldsymbol{y}_{nn}^T \boldsymbol{\pi}_n$ and solve it for $\boldsymbol{\pi}_n$ by specifying τ_n and $\boldsymbol{y}_{nn}^T$ for a given trajectory on Joint n. By iterating the procedure, the manipulator parameters can be identified on the basis of measurements performed joint by joint from the outer link to the base. Such procedure, however, may have the inconvenience to accumulate any error due to ill-conditioning of the matrices involved step by step. It may then be worth operating with a global procedure by imposing motions on all manipulator joints at the same time.

Regarding the rank of matrix $\bar{\boldsymbol{Y}}$, it is possible to identify only the dynamic parameters of the manipulator that contribute to the dynamic model. Example 7.2 has indeed shown that for the two-link planar arm considered, only 8 out of the 22 possible dynamic parameters appear in the dynamic model. Hence, there exist some dynamic parameters which, in view of the disposition

of manipulator links and joints, are *non-identifiable*, since for any trajectory assigned to the structure they do not contribute to the equations of motion. A direct consequence is that the columns of the matrix $\boldsymbol{Y}$ in (7.80) corresponding to such parameters are null and thus they have to be removed from the matrix itself; e.g., the resulting (2×8) matrix in (7.84).

Another issue to consider about determination of the effective number of parameters that can be identified by (7.86) is that some parameters can be *identified in linear combinations* whenever they do not appear isolated in the equations. In such a case, it is necessary, for each linear combination, to remove as many columns of the matrix $\boldsymbol{Y}$ as the number of parameters in the linear combination minus one.

For the determination of the minimum number of identifiable parameters that allow direct application of the least-squares technique based on (7.86), it is possible to inspect directly the equations of the dynamic model, as long as the manipulator has few joints. Otherwise, numerical techniques based on singular value decomposition of matrix $\bar{\boldsymbol{Y}}$ have to be used. If the matrix $\bar{\boldsymbol{Y}}$ resulting from a series of measurements is not full-rank, one has to resort to a *damped least-squares inverse* of $\bar{\boldsymbol{Y}}$ where solution accuracy depends on the weight of the damping factor.

In the above discussion, the type of trajectory imposed to the manipulator joints has not been explicitly addressed. It can be generally ascertained that the choice should be oriented in favor of polynomial type trajectories which are sufficiently *rich* to allow an accurate evaluation of the identifiable parameters. This corresponds to achieving a low condition number of the matrix $\bar{\boldsymbol{Y}}^T \bar{\boldsymbol{Y}}$ along the trajectory. On the other hand, such trajectories should not excite any unmodelled dynamic effects such as joint elasticity or link flexibility that would naturally lead to unreliable estimates of the dynamic parameters to identify.

Finally, it is worth observing that the technique presented above can also be extended to the identification of the dynamic parameters of an unknown payload at the manipulator's end-effector. In such a case, the payload can be regarded as a structural modification of the last link and one may proceed to identify the dynamic parameters of the modified link. To this end, if a force sensor is available at the manipulator's wrist, it is possible to characterize directly the dynamic parameters of the payload starting from force sensor measurements.

7.5 Newton–Euler Formulation

In the Lagrange formulation, the manipulator dynamic model is derived starting from the total Lagrangian of the system. On the other hand, the *Newton–Euler* formulation is based on a balance of all the forces acting on the generic link of the manipulator. This leads to a set of equations whose structure allows a recursive type of solution; a forward recursion is performed for propagating

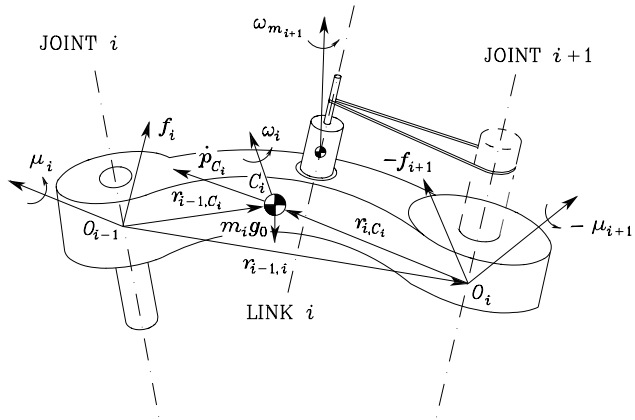

Fig. 7.13. Characterization of Link i for Newton–Euler formulation

link velocities and accelerations, followed by a backward recursion for propagating forces.

Consider the generic *augmented Link* i (Link i plus motor of Joint $i+1$) of the manipulator kinematic chain (Fig. 7.13). According to what was presented in Sect. 7.2.2, one can refer to the centre of mass C_i of the augmented link to characterize the following parameters:

- m_i mass of augmented link,
- $\bar{\boldsymbol{I}}_i$ inertia tensor of augmented link,
- I_{m_i} moment of inertia of rotor,
- $\boldsymbol{r}_{i-1,C_i}$ vector from origin of Frame $(i-1)$ to centre of mass C_i,
- $\boldsymbol{r}_{i,C_i}$ vector from origin of Frame i to centre of mass C_i,
- $\boldsymbol{r}_{i-1,i}$ vector from origin of Frame $(i-1)$ to origin of Frame i.

The velocities and accelerations to be considered are:

- $\dot{\boldsymbol{p}}_{C_i}$ linear velocity of centre of mass C_i,
- $\dot{\boldsymbol{p}}_i$ linear velocity of origin of Frame i,
- $\boldsymbol{\omega}_i$ angular velocity of link,
- $\boldsymbol{\omega}_{m_i}$ angular velocity of rotor,
- $\ddot{\boldsymbol{p}}_{C_i}$ linear acceleration of centre of mass C_i,
- $\ddot{\boldsymbol{p}}_i$ linear acceleration of origin of Frame i,
- $\dot{\boldsymbol{\omega}}_i$ angular acceleration of link,
- $\dot{\boldsymbol{\omega}}_{m_i}$ angular acceleration of rotor,
- $\boldsymbol{g}_0$ gravity acceleration.

The forces and moments to be considered are:

- $\boldsymbol{f}_i$ force exerted by Link $i-1$ on Link i,
- $-\boldsymbol{f}_{i+1}$ force exerted by Link $i+1$ on Link i,

- $\boldsymbol{\mu}_i$ moment exerted by Link $i-1$ on Link i with respect to origin of Frame $i-1$,
- $-\boldsymbol{\mu}_{i+1}$ moment exerted by Link $i+1$ on Link i with respect to origin of Frame i.

Initially, all the vectors and matrices are assumed to be expressed with reference to the *base frame*.

As already anticipated, the Newton–Euler formulation describes the motion of the link in terms of a balance of forces and moments acting on it.

The *Newton* equation for the *translational* motion of the centre of mass can be written as

$$\boldsymbol{f}_i - \boldsymbol{f}_{i+1} + m_i \boldsymbol{g}_0 = m_i \ddot{\boldsymbol{p}}_{C_i}. \tag{7.87}$$

The *Euler* equation for the *rotational* motion of the link (referring moments to the centre of mass) can be written as

$$\boldsymbol{\mu}_i + \boldsymbol{f}_i \times \boldsymbol{r}_{i-1,C_i} - \boldsymbol{\mu}_{i+1} - \boldsymbol{f}_{i+1} \times \boldsymbol{r}_{i,C_i} = \frac{d}{dt}(\bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i + k_{r,i+1} \dot{q}_{i+1} I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}}), \tag{7.88}$$

where (7.67) has been used for the angular momentum of the rotor. Notice that the gravitational force $m_i \boldsymbol{g}_0$ does not generate any moment, since it is concentrated at the centre of mass.

As pointed out in the above Lagrange formulation, it is convenient to express the inertia tensor in the current frame (constant tensor). Hence, according to (7.12), one has $\bar{\boldsymbol{I}}_i = \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T$, where $\boldsymbol{R}_i$ is the rotation matrix from Frame i to the base frame. Substituting this relation in the first term on the right-hand side of (7.88) yields

$$\begin{aligned} \frac{d}{dt}(\bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i) &= \dot{\boldsymbol{R}}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \boldsymbol{\omega}_i + \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \dot{\boldsymbol{R}}_i^T \boldsymbol{\omega}_i + \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \dot{\boldsymbol{\omega}}_i \\ &= \boldsymbol{S}(\boldsymbol{\omega}_i) \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \boldsymbol{\omega}_i + \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \boldsymbol{S}^T(\boldsymbol{\omega}_i) \boldsymbol{\omega}_i + \boldsymbol{R}_i \bar{\boldsymbol{I}}_i^i \boldsymbol{R}_i^T \dot{\boldsymbol{\omega}}_i \\ &= \bar{\boldsymbol{I}}_i \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times (\bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i) \end{aligned} \tag{7.89}$$

where the second term represents the *gyroscopic* torque induced by the dependence of $\bar{\boldsymbol{I}}_i$ on link orientation.[7] Moreover, by observing that the unit vector $\boldsymbol{z}_{m_{i+1}}$ rotates accordingly to Link i, the derivative needed in the second term on the right-hand side of (7.88) is

$$\frac{d}{dt}(\dot{q}_{i+1} I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}}) = \ddot{q}_{i+1} I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}} + \dot{q}_{i+1} I_{m_{i+1}} \boldsymbol{\omega}_i \times \boldsymbol{z}_{m_{i+1}} \tag{7.90}$$

By substituting (7.89), (7.90) in (7.88), the resulting Euler equation is

$$\begin{aligned} \boldsymbol{\mu}_i + \boldsymbol{f}_i \times \boldsymbol{r}_{i-1,C_i} \; -\boldsymbol{\mu}_{i+1} - \boldsymbol{f}_{i+1} \times \boldsymbol{r}_{i,C_i} &= \bar{\boldsymbol{I}}_i \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times (\bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i) \\ + k_{r,i+1} \ddot{q}_{i+1} I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}} &+ k_{r,i+1} \dot{q}_{i+1} I_{m_{i+1}} \boldsymbol{\omega}_i \times \boldsymbol{z}_{m_{i+1}}. \end{aligned} \tag{7.91}$$

[7] In deriving (7.89), the operator $\boldsymbol{S}$ has been introduced to compute the derivative of $\boldsymbol{R}_i$, as in (3.8); also, the property $\boldsymbol{S}^T(\boldsymbol{\omega}_i)\boldsymbol{\omega}_i = \boldsymbol{0}$ has been utilized.

The generalized force at Joint i can be computed by projecting the force $\boldsymbol{f}_i$ for a prismatic joint, or the moment $\boldsymbol{\mu}_i$ for a revolute joint, along the joint axis. In addition, there is the contribution of the rotor inertia torque $k_{ri} I_{m_i} \dot{\boldsymbol{\omega}}_{m_i}^T \boldsymbol{z}_{m_i}$. Hence, the generalized force at Joint i is expressed by

$$\tau_i = \begin{cases} \boldsymbol{f}_i^T \boldsymbol{z}_{i-1} + k_{ri} I_{m_i} \dot{\boldsymbol{\omega}}_{m_i}^T \boldsymbol{z}_{m_i} & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{\mu}_i^T \boldsymbol{z}_{i-1} + k_{ri} I_{m_i} \dot{\boldsymbol{\omega}}_{m_i}^T \boldsymbol{z}_{m_i} & \text{for a } \textit{revolute} \text{ joint.} \end{cases} \tag{7.92}$$

7.5.1 Link Accelerations

The Newton–Euler equations in (7.87), (7.91) and the equation in (7.92) require the computation of linear and angular acceleration of Link i and Rotor i. This computation can be carried out on the basis of the relations expressing the linear and angular velocities previously derived. The equations in (3.21), (3.22), (3.25), (3.26) can be briefly rewritten as

$$\boldsymbol{\omega}_i = \begin{cases} \boldsymbol{\omega}_{i-1} & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{\omega}_{i-1} + \dot{\vartheta}_i \boldsymbol{z}_{i-1} & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{7.93}$$

and

$$\dot{\boldsymbol{p}}_i = \begin{cases} \dot{\boldsymbol{p}}_{i-1} + \dot{d}_i \boldsymbol{z}_{i-1} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i} & \text{for a } \textit{prismatic} \text{ joint} \\ \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i} & \text{for a } \textit{revolute} \text{ joint.} \end{cases} \tag{7.94}$$

As for the angular acceleration of the link, it can be seen that, for a prismatic joint, differentiating (3.21) with respect to time gives

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1}, \tag{7.95}$$

whereas, for a revolute joint, differentiating (3.25) with respect to time gives

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} + \ddot{\vartheta}_i \boldsymbol{z}_{i-1} + \dot{\vartheta}_i \boldsymbol{\omega}_{i-1} \times \boldsymbol{z}_{i-1}. \tag{7.96}$$

As for the linear acceleration of the link, for a prismatic joint, differentiating (3.22) with respect to time gives

$$\begin{aligned} \ddot{\boldsymbol{p}}_i = \ddot{\boldsymbol{p}}_{i-1} + \ddot{d}_i \boldsymbol{z}_{i-1} + \dot{d}_i \boldsymbol{\omega}_{i-1} \times \boldsymbol{z}_{i-1} + \dot{\boldsymbol{\omega}}_i \times \boldsymbol{r}_{i-1,i} \\ + \boldsymbol{\omega}_i \times \dot{d}_i \boldsymbol{z}_{i-1} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_{i-1} \times \boldsymbol{r}_{i-1,i}) \end{aligned} \tag{7.97}$$

where the relation $\dot{\boldsymbol{r}}_{i-1,i} = \dot{d}_i \boldsymbol{z}_{i-1} + \boldsymbol{\omega}_{i-1} \times \boldsymbol{r}_{i-1,i}$ has been used. Hence, in view of (3.21), the equation in (7.97) can be rewritten as

$$\ddot{\boldsymbol{p}}_i = \ddot{\boldsymbol{p}}_{i-1} + \ddot{d}_i \boldsymbol{z}_{i-1} + 2\dot{d}_i \boldsymbol{\omega}_i \times \boldsymbol{z}_{i-1} + \dot{\boldsymbol{\omega}}_i \times \boldsymbol{r}_{i-1,i} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i}). \tag{7.98}$$

Also, for a revolute joint, differentiating (3.26) with respect to time gives

$$\ddot{\boldsymbol{p}}_i = \ddot{\boldsymbol{p}}_{i-1} + \dot{\boldsymbol{\omega}}_i \times \boldsymbol{r}_{i-1,i} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i}). \tag{7.99}$$

In summary, the equations in (7.95), (7.96), (7.98), (7.99) can be compactly rewritten as

$$\dot{\boldsymbol{\omega}}_i = \begin{cases} \dot{\boldsymbol{\omega}}_{i-1} & \text{for a } \textit{prismatic} \text{ joint} \\ \dot{\boldsymbol{\omega}}_{i-1} + \ddot{\vartheta}_i \boldsymbol{z}_{i-1} + \dot{\vartheta}_i \boldsymbol{\omega}_{i-1} \times \boldsymbol{z}_{i-1} & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{7.100}$$

and

$$\ddot{\boldsymbol{p}}_i = \begin{cases} \ddot{\boldsymbol{p}}_{i-1} + \ddot{d}_i \boldsymbol{z}_{i-1} + 2\dot{d}_i \boldsymbol{\omega}_i \times \boldsymbol{z}_{i-1} & \text{for a } \textit{prismatic} \text{ joint} \\ \quad + \dot{\boldsymbol{\omega}}_i \times \boldsymbol{r}_{i-1,i} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i}) & \\ \ddot{\boldsymbol{p}}_{i-1} + \dot{\boldsymbol{\omega}}_i \times \boldsymbol{r}_{i-1,i} & \text{for a } \textit{revolute} \text{ joint.} \\ \quad + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i}) & \end{cases} \tag{7.101}$$

The acceleration of the centre of mass of Link i required by the Newton equation in (7.87) can be derived from (3.15), since $\dot{\boldsymbol{r}}_{i,C_i}^i = \boldsymbol{0}$; by differentiating (3.15) with respect to time, the acceleration of the centre of mass C_i can be expressed as a function of the velocity and acceleration of the origin of Frame i, i.e.,

$$\ddot{\boldsymbol{p}}_{C_i} = \ddot{\boldsymbol{p}}_i + \dot{\boldsymbol{\omega}}_i \times \boldsymbol{r}_{i,C_i} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \boldsymbol{r}_{i,C_i}). \tag{7.102}$$

Finally, the angular acceleration of the rotor can be obtained by time differentiation of (7.23), i.e.,

$$\dot{\boldsymbol{\omega}}_{m_i} = \dot{\boldsymbol{\omega}}_{i-1} + k_{ri} \ddot{q}_i \boldsymbol{z}_{m_i} + k_{ri} \dot{q}_i \boldsymbol{\omega}_{i-1} \times \boldsymbol{z}_{m_i}. \tag{7.103}$$

7.5.2 Recursive Algorithm

It is worth remarking that the resulting Newton–Euler equations of motion are *not* in *closed form*, since the motion of a single link is coupled to the motion of the other links through the kinematic relationship for velocities and accelerations.

Once the joint positions, velocities and accelerations are known, one can compute the link velocities and accelerations, and the Newton–Euler equations can be utilized to find the forces and moments acting on each link in a recursive fashion, starting from the force and moment applied to the end-effector. On the other hand, also link and rotor velocities and accelerations can be computed recursively starting from the velocity and acceleration of the base link. In summary, a computationally *recursive algorithm* can be constructed that features a *forward recursion* relative to the propagation of *velocities and accelerations* and a *backward recursion* for the propagation of *forces and moments* along the structure.

For the forward recursion, once $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$, and the velocity and acceleration of the base link $\boldsymbol{\omega}_0$, $\ddot{\boldsymbol{p}}_0 - \boldsymbol{g}_0$, $\dot{\boldsymbol{\omega}}_0$ are specified, $\boldsymbol{\omega}_i$, $\dot{\boldsymbol{\omega}}_i$, $\ddot{\boldsymbol{p}}_i$, $\ddot{\boldsymbol{p}}_{C_i}$, $\dot{\boldsymbol{\omega}}_{m_i}$ can be computed using (7.93), (7.100), (7.101), (7.102), (7.103), respectively. Notice that the linear acceleration has been taken as $\ddot{\boldsymbol{p}}_0 - \boldsymbol{g}_0$ so as to incorporate the

term $-\boldsymbol{g}_0$ in the computation of the acceleration of the centre of mass $\ddot{\boldsymbol{p}}_{C_i}$ via (7.101), (7.102).

Having computed the velocities and accelerations with the forward recursion from the base link to the end-effector, a backward recursion can be carried out for the forces. In detail, once $\boldsymbol{h}_e = [\,\boldsymbol{f}_{n+1}^T \quad \boldsymbol{\mu}_{n+1}^T\,]^T$ is given (eventually $\boldsymbol{h}_e = \boldsymbol{0}$), the Newton equation in (7.87) to be used for the recursion can be rewritten as

$$\boldsymbol{f}_i = \boldsymbol{f}_{i+1} + m_i \ddot{\boldsymbol{p}}_{C_i} \tag{7.104}$$

since the contribution of gravity acceleration has already been included in $\ddot{\boldsymbol{p}}_{C_i}$. Further, the Euler equation gives

$$\begin{aligned}\boldsymbol{\mu}_i = &-\boldsymbol{f}_i \times (\boldsymbol{r}_{i-1,i} + \boldsymbol{r}_{i,C_i}) + \boldsymbol{\mu}_{i+1} + \boldsymbol{f}_{i+1} \times \boldsymbol{r}_{i,C_i} + \bar{\boldsymbol{I}}_i \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times (\bar{\boldsymbol{I}}_i \boldsymbol{\omega}_i) \\ &+ k_{r,i+1} \ddot{q}_{i+1} I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}} + k_{r,i+1} \dot{q}_{i+1} I_{m_{i+1}} \boldsymbol{\omega}_i \times \boldsymbol{z}_{m_{i+1}}\end{aligned} \tag{7.105}$$

which derives from (7.91), where $\boldsymbol{r}_{i-1,C_i}$ has been expressed as the sum of the two vectors appearing already in the forward recursion. Finally, the generalized forces resulting at the joints can be computed from (7.92) as

$$\tau_i = \begin{cases} \boldsymbol{f}_i^T \boldsymbol{z}_{i-1} + k_{ri} I_{m_i} \dot{\boldsymbol{\omega}}_{m_i}^T \boldsymbol{z}_{m_i} & \\ \quad + F_{vi} \dot{d}_i + F_{si} \operatorname{sgn}(\dot{d}_i) & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{\mu}_i^T \boldsymbol{z}_{i-1} + k_{ri} I_{m_i} \dot{\boldsymbol{\omega}}_{m_i}^T \boldsymbol{z}_{m_i} & \\ \quad + F_{vi} \dot{\vartheta}_i + F_{si} \operatorname{sgn}(\dot{\vartheta}_i) & \text{for a } \textit{revolute} \text{ joint,} \end{cases} \tag{7.106}$$

where joint viscous and Coulomb friction torques have been included.

In the above derivation, it has been assumed that all vectors were referred to the base frame. To simplify greatly computation, however, the recursion is computationally more efficient if all vectors are referred to the current frame on Link i. This implies that all vectors that need to be transformed from Frame $i+1$ into Frame i have to be multiplied by the rotation matrix $\boldsymbol{R}_{i+1}^i$, whereas all vectors that need to be transformed from Frame $i-1$ into Frame i have to be multiplied by the rotation matrix $\boldsymbol{R}_i^{i-1T}$. Therefore, the equations in (7.93), (7.100), (7.101), (7.102), (7.103), (7.104), (7.105), (7.106) can be rewritten as:

$$\boldsymbol{\omega}_i^i = \begin{cases} \boldsymbol{R}_i^{i-1T} \boldsymbol{\omega}_{i-1}^{i-1} & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{R}_i^{i-1T} (\boldsymbol{\omega}_{i-1}^{i-1} + \dot{\vartheta}_i \boldsymbol{z}_0) & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{7.107}$$

$$\dot{\boldsymbol{\omega}}_i^i = \begin{cases} \boldsymbol{R}_i^{i-1T} \dot{\boldsymbol{\omega}}_{i-1}^{i-1} & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{R}_i^{i-1T} (\dot{\boldsymbol{\omega}}_{i-1}^{i-1} + \ddot{\vartheta}_i \boldsymbol{z}_0 + \dot{\vartheta}_i \boldsymbol{\omega}_{i-1}^{i-1} \times \boldsymbol{z}_0) & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{7.108}$$

$$\ddot{\boldsymbol{p}}_i^i = \begin{cases} \boldsymbol{R}_i^{i-1T} (\ddot{\boldsymbol{p}}_{i-1}^{i-1} + \ddot{d}_i \boldsymbol{z}_0) + 2\dot{d}_i \boldsymbol{\omega}_i^i \times \boldsymbol{R}_i^{i-1T} \boldsymbol{z}_0 & \\ \quad + \dot{\boldsymbol{\omega}}_i^i \times \boldsymbol{r}_{i-1,i}^i + \boldsymbol{\omega}_i^i \times (\boldsymbol{\omega}_i^i \times \boldsymbol{r}_{i-1,i}^i) & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{R}_i^{i-1T} \ddot{\boldsymbol{p}}_{i-1}^{i-1} + \dot{\boldsymbol{\omega}}_i^i \times \boldsymbol{r}_{i-1,i}^i & \\ \quad + \boldsymbol{\omega}_i^i \times (\boldsymbol{\omega}_i^i \times \boldsymbol{r}_{i-1,i}^i) & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{7.109}$$

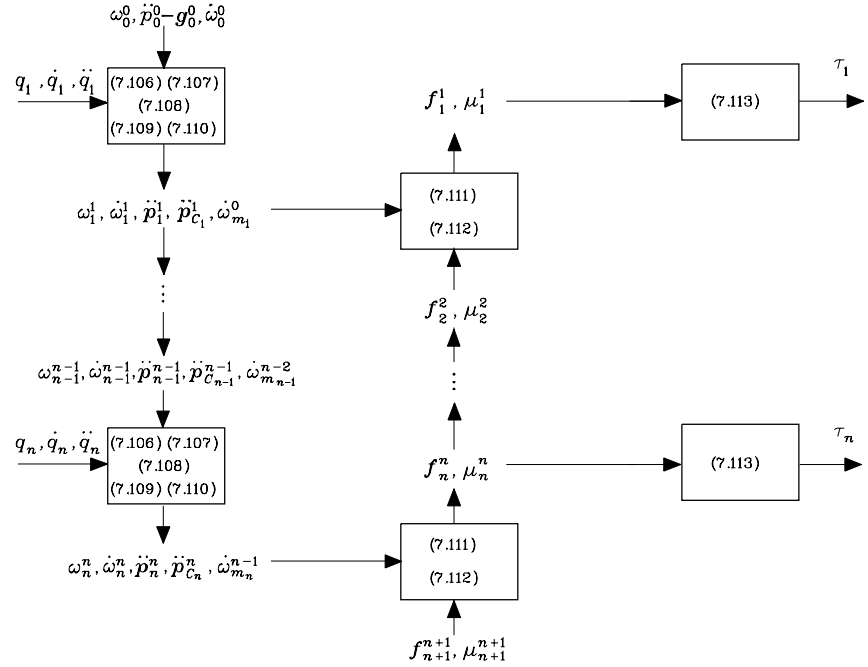

Fig. 7.14. Computational structure of the Newton–Euler recursive algorithm

$$\ddot{\boldsymbol{p}}_{C_i}^i = \ddot{\boldsymbol{p}}_i^i + \dot{\boldsymbol{\omega}}_i^i \times \boldsymbol{r}_{i,C_i}^i + \boldsymbol{\omega}_i^i \times (\boldsymbol{\omega}_i^i \times \boldsymbol{r}_{i,C_i}^i) \tag{7.110}$$

$$\dot{\boldsymbol{\omega}}_{m_i}^{i-1} = \dot{\boldsymbol{\omega}}_{i-1}^{i-1} + k_{ri} \ddot{q}_i \boldsymbol{z}_{m_i}^{i-1} + k_{ri} \dot{q}_i \boldsymbol{\omega}_{i-1}^{i-1} \times \boldsymbol{z}_{m_i}^{i-1} \tag{7.111}$$

$$\boldsymbol{f}_i^i = \boldsymbol{R}_{i+1}^i \boldsymbol{f}_{i+1}^{i+1} + m_i \ddot{\boldsymbol{p}}_{C_i}^i \tag{7.112}$$

$$\begin{aligned}\boldsymbol{\mu}_i^i = &-\boldsymbol{f}_i^i \times (\boldsymbol{r}_{i-1,i}^i + \boldsymbol{r}_{i,C_i}^i) + \boldsymbol{R}_{i+1}^i \boldsymbol{\mu}_{i+1}^{i+1} + \boldsymbol{R}_{i+1}^i \boldsymbol{f}_{i+1}^{i+1} \times \boldsymbol{r}_{i,C_i}^i \\ &+ \bar{\boldsymbol{I}}_i^i \dot{\boldsymbol{\omega}}_i^i + \boldsymbol{\omega}_i^i \times (\bar{\boldsymbol{I}}_i^i \boldsymbol{\omega}_i^i) \\ &+ \boldsymbol{\omega}_i^i \times (\bar{\boldsymbol{I}}_i^i \boldsymbol{\omega}_i^i) + k_{r,i+1} \ddot{q}_{i+1} I_{m_{i+1}} \boldsymbol{z}_{m_{i+1}}^i + k_{r,i+1} \dot{q}_{i+1} I_{m_{i+1}} \boldsymbol{\omega}_i^i \times \boldsymbol{z}_{m_{i+1}}^i\end{aligned} \tag{7.113}$$

$$\tau_i = \begin{cases} \boldsymbol{f}_i^{iT} \boldsymbol{R}_i^{i-1T} \boldsymbol{z}_0 + k_{ri} I_{m_i} \dot{\boldsymbol{\omega}}_{m_i}^{i-1T} \boldsymbol{z}_{m_i}^{i-1} & \\ \quad + F_{vi} \dot{d}_i + F_{si} \operatorname{sgn}(\dot{d}_i) & \text{for a } \textit{prismatic} \text{ joint} \\ \boldsymbol{\mu}_i^{iT} \boldsymbol{R}_i^{i-1T} \boldsymbol{z}_0 + k_{ri} I_{m_i} \dot{\boldsymbol{\omega}}_{m_i}^{i-1T} \boldsymbol{z}_{m_i}^{i-1} & \\ \quad + F_{vi} \dot{\vartheta}_i + F_{si} \operatorname{sgn}(\dot{\vartheta}_i) & \text{for a } \textit{revolute} \text{ joint.} \end{cases} \tag{7.114}$$

The above equations have the advantage that the quantities $\bar{\boldsymbol{I}}_i^i$, $\boldsymbol{r}_{i,C_i}^i$, $\boldsymbol{z}_{m_i}^{i-1}$ are *constant*; further, it is $\boldsymbol{z}_0 = [\,0 \quad 0 \quad 1\,]^T$.

To summarize, for given joint positions, velocities and accelerations, the recursive algorithm is carried out in the following two phases:

- With known initial conditions $\boldsymbol{\omega}_0^0$, $\ddot{\boldsymbol{p}}_0^0 - \boldsymbol{g}_0^0$, and $\dot{\boldsymbol{\omega}}_0^0$, use (7.107), (7.108), (7.109), (7.110), (7.111), for $i = 1, \ldots, n$, to compute $\boldsymbol{\omega}_i^i$, $\dot{\boldsymbol{\omega}}_i^i$, $\ddot{\boldsymbol{p}}_i^i$, $\ddot{\boldsymbol{p}}_{C_i}^i$, $\dot{\boldsymbol{\omega}}_{m_i}^{i-1}$.
- With known terminal conditions $\boldsymbol{f}_{n+1}^{n+1}$ and $\boldsymbol{\mu}_{n+1}^{n+1}$, use (7.112), (7.113), for $i = n, \ldots, 1$, to compute $\boldsymbol{f}_i^i$, $\boldsymbol{\mu}_i^i$, and then (7.114) to compute τ_i.

The computational structure of the algorithm is schematically illustrated in Fig. 7.14.

7.5.3 Example

In the following, an example to illustrate the single steps of the Newton–Euler algorithm is developed. Consider the two-link planar arm whose dynamic model has already been derived in Example 7.2.

Start by imposing the initial conditions for the velocities and accelerations:

$$\ddot{\boldsymbol{p}}_0^0 - \boldsymbol{g}_0^0 = [\,0 \quad g \quad 0\,]^T \qquad \boldsymbol{\omega}_0^0 = \dot{\boldsymbol{\omega}}_0^0 = \boldsymbol{0},$$

and the terminal conditions for the forces:

$$\boldsymbol{f}_3^3 = \boldsymbol{0} \qquad \boldsymbol{\mu}_3^3 = \boldsymbol{0}.$$

All quantities are referred to the current link frame. As a consequence, the following constant vectors are obtained:

$$\boldsymbol{r}_{1,C_1}^1 = \begin{bmatrix} \ell_{C_1} \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{r}_{0,1}^1 = \begin{bmatrix} a_1 \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{r}_{2,C_2}^2 = \begin{bmatrix} \ell_{C_2} \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{r}_{1,2}^2 = \begin{bmatrix} a_2 \\ 0 \\ 0 \end{bmatrix}$$

where ℓ_{C_1} and ℓ_{C_2} are both negative quantities. The rotation matrices needed for vector transformation from one frame to another are

$$\boldsymbol{R}_i^{i-1} = \begin{bmatrix} c_i & -s_i & 0 \\ s_i & c_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad i = 1, 2 \qquad \boldsymbol{R}_3^2 = \boldsymbol{I}.$$

Further, it is assumed that the axes of rotation of the two rotors coincide with the respective joint axes, i.e., $\boldsymbol{z}_{m_i}^{i-1} = \boldsymbol{z}_0 = [\,0 \quad 0 \quad 1\,]^T$ for $i = 1, 2$.

According to (7.107)–(7.114), the Newton–Euler algorithm requires the execution of the following steps:

- Forward recursion: Link 1

$$\boldsymbol{\omega}_1^1 = \begin{bmatrix} 0 \\ 0 \\ \dot{\vartheta}_1 \end{bmatrix}$$

$$\dot{\boldsymbol{\omega}}_1^1 = \begin{bmatrix} 0 \\ 0 \\ \ddot{\vartheta}_1 \end{bmatrix}$$

$$\ddot{\boldsymbol{p}}_1^1 = \begin{bmatrix} -a_1\dot{\vartheta}_1^2 + gs_1 \\ a_1\ddot{\vartheta}_1 + gc_1 \\ 0 \end{bmatrix}$$

$$\ddot{\boldsymbol{p}}_{C_1}^1 = \begin{bmatrix} -(\ell_{C_1} + a_1)\dot{\vartheta}_1^2 + gs_1 \\ (\ell_{C_1} + a_1)\ddot{\vartheta}_1 + gc_1 \\ 0 \end{bmatrix}$$

$$\dot{\boldsymbol{\omega}}_{m_1}^0 = \begin{bmatrix} 0 \\ 0 \\ k_{r1}\ddot{\vartheta}_1 \end{bmatrix}.$$

- Forward recursion: Link 2

$$\boldsymbol{\omega}_2^2 = \begin{bmatrix} 0 \\ 0 \\ \dot{\vartheta}_1 + \dot{\vartheta}_2 \end{bmatrix}$$

$$\dot{\boldsymbol{\omega}}_2^2 = \begin{bmatrix} 0 \\ 0 \\ \ddot{\vartheta}_1 + \ddot{\vartheta}_2 \end{bmatrix}$$

$$\ddot{\boldsymbol{p}}_2^2 = \begin{bmatrix} a_1 s_2\ddot{\vartheta}_1 - a_1 c_2\dot{\vartheta}_1^2 - a_2(\dot{\vartheta}_1 + \dot{\vartheta}_2)^2 + gs_{12} \\ a_1 c_2\ddot{\vartheta}_1 + a_2(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) + a_1 s_2\dot{\vartheta}_1^2 + gc_{12} \\ 0 \end{bmatrix}$$

$$\ddot{\boldsymbol{p}}_{C_2}^2 = \begin{bmatrix} a_1 s_2\ddot{\vartheta}_1 - a_1 c_2\dot{\vartheta}_1^2 - (\ell_{C_2} + a_2)(\dot{\vartheta}_1 + \dot{\vartheta}_2)^2 + gs_{12} \\ a_1 c_2\ddot{\vartheta}_1 + (\ell_{C_2} + a_2)(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) + a_1 s_2\dot{\vartheta}_1^2 + gc_{12} \\ 0 \end{bmatrix}$$

$$\dot{\boldsymbol{\omega}}_{m_2}^1 = \begin{bmatrix} 0 \\ 0 \\ \ddot{\vartheta}_1 + k_{r2}\ddot{\vartheta}_2 \end{bmatrix}.$$

- Backward recursion: Link 2

$$\boldsymbol{f}_2^2 = \begin{bmatrix} m_2\big(a_1 s_2 \ddot{\vartheta}_1 - a_1 c_2 \dot{\vartheta}_1^2 - (\ell_{C_2} + a_2)(\dot{\vartheta}_1 + \dot{\vartheta}_2)^2 + g s_{12}\big) \\ m_2\big(a_1 c_2 \ddot{\vartheta}_1 + (\ell_{C_2} + a_2)(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) + a_1 s_2 \dot{\vartheta}_1^2 + g c_{12}\big) \\ 0 \end{bmatrix}$$

$$\boldsymbol{\mu}_2^2 = \begin{bmatrix} * \\ * \\ \bar{I}_{2zz}(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) + m_2(\ell_{C_2} + a_2)^2(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) + m_2 a_1(\ell_{C_2} + a_2) c_2 \ddot{\vartheta}_1 \\ \qquad + m_2 a_1(\ell_{C_2} + a_2) s_2 \dot{\vartheta}_1^2 + m_2(\ell_{C_2} + a_2) g c_{12} \end{bmatrix}$$

$$\begin{aligned} \tau_2 = {} & \big(\bar{I}_{2zz} + m_2\left((\ell_{C_2} + a_2)^2 + a_1(\ell_{C_2} + a_2)c_2\right) + k_{r2} I_{m_2}\big)\ddot{\vartheta}_1 \\ & + \big(\bar{I}_{2zz} + m_2(\ell_{C_2} + a_2)^2 + k_{r2}^2 I_{m_2}\big)\,\ddot{\vartheta}_2 \\ & + m_2 a_1(\ell_{C_2} + a_2) s_2 \dot{\vartheta}_1^2 + m_2(\ell_{C_2} + a_2) g c_{12}. \end{aligned}$$

- Backward recursion: Link 1

$$\boldsymbol{f}_1^1 = \begin{bmatrix} -m_2(\ell_{C_2} + a_2) s_2(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) - m_1(\ell_{C_1} + a_1)\dot{\vartheta}_1^2 - m_2 a_1 \dot{\vartheta}_1^2 \\ -m_2(\ell_{C_2} + a_2) c_2(\dot{\vartheta}_1 + \dot{\vartheta}_2)^2 + (m_1 + m_2) g s_1 \\ m_1(\ell_{C_1} + a_1)\ddot{\vartheta}_1 + m_2 a_1 \ddot{\vartheta}_1 + m_2(\ell_{C_2} + a_2) c_2(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) \\ -m_2(\ell_{C_2} + a_2) s_2(\dot{\vartheta}_1 + \dot{\vartheta}_2)^2 + (m_1 + m_2) g c_1 \\ 0 \end{bmatrix}$$

$$\boldsymbol{\mu}_1^1 = \begin{bmatrix} * \\ * \\ \bar{I}_{1zz}\ddot{\vartheta}_1 + m_2 a_1^2 \ddot{\vartheta}_1 + m_1(\ell_{C_1} + a_1)^2 \ddot{\vartheta}_1 + m_2 a_1(\ell_{C_2} + a_2) c_2 \ddot{\vartheta}_1 \\ \qquad + \bar{I}_{2zz}(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) + m_2 a_1(\ell_{C_2} + a_2) c_2(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) \\ \qquad + m_2(\ell_{C_2} + a_2)^2(\ddot{\vartheta}_1 + \ddot{\vartheta}_2) + k_{r2} I_{m_2} \ddot{\vartheta}_2 \\ \qquad + m_2 a_1(\ell_{C_2} + a_2) s_2 \dot{\vartheta}_1^2 - m_2 a_1(\ell_{C_2} + a_2) s_2(\dot{\vartheta}_1 + \dot{\vartheta}_2)^2 \\ \qquad + m_1(\ell_{C_1} + a_1) g c_1 + m_2 a_1 g c_1 + m_2(\ell_{C_2} + a_2) g c_{12} \end{bmatrix}$$

$$\begin{aligned} \tau_1 = {} & \Big(\bar{I}_{1zz} + m_1(\ell_{C_1} + a_1)^2 + k_{r1}^2 I_{m_1} + \bar{I}_{2zz} \\ & \qquad + m_2\left(a_1^2 + (\ell_{C_2} + a_2)^2 + 2a_1(\ell_{C_2} + a_2)c_2\right)\Big)\ddot{\vartheta}_1 \\ & + \Big(\bar{I}_{2zz} + m_2\left((\ell_{C_2} + a_2)^2 + a_1(\ell_{C_2} + a_2)c_2\right) + k_{r2} I_{m_2}\Big)\ddot{\vartheta}_2 \\ & - 2m_2 a_1(\ell_{C_2} + a_2) s_2 \dot{\vartheta}_1 \dot{\vartheta}_2 - m_2 a_1(\ell_{C_2} + a_2) s_2 \dot{\vartheta}_2^2 \\ & + \big(m_1(\ell_{C_1} + a_1) + m_2 a_1\big) g c_1 + m_2(\ell_{C_2} + a_2) g c_{12}. \end{aligned}$$

As for the moment components, those marked by the symbol '$*$' have not been computed, since they are not related to the joint torques τ_2 and τ_1.

Expressing the dynamic parameters in the above torques as a function of the link and rotor parameters as in (7.83) yields

$$\begin{aligned} m_1 &= m_{\ell_1} + m_{m_2} \\ m_1 \ell_{C_1} &= m_{\ell_1}(\ell_1 - a_1) \\ \bar{I}_{1zz} + m_1 \ell_{C_1}^2 = \widehat{I}_1 &= I_{\ell_1} + m_{\ell_1}(\ell_1 - a_1)^2 + I_{m_2} \\ m_2 &= m_{\ell_2} \\ m_2 \ell_{C_2} &= m_{\ell_2}(\ell_2 - a_2) \\ \bar{I}_{2zz} + m_2 \ell_{C_2}^2 = \widehat{I}_2 &= I_{\ell_2} + m_{\ell_2}(\ell_2 - a_2)^2. \end{aligned}$$

On the basis of these relations, it can be verified that the resulting dynamic model coincides with the model derived in (7.82) with Lagrange formulation.

7.6 Direct Dynamics and Inverse Dynamics

Both Lagrange formulation and Newton–Euler formulation allow the computation of the relationship between the joint torques — and, if present, the end-effector forces — and the motion of the structure. A comparison between the two approaches reveals what follows. The *Lagrange* formulation has the following advantages:

- It is *systematic* and of immediate comprehension.
- It provides the equations of motion in a compact *analytical form* containing the inertia matrix, the matrix in the centrifugal and Coriolis forces, and the vector of gravitational forces. Such a form is advantageous for *control design*.
- It is effective if it is wished to include more complex mechanical effects such as flexible link deformation.

The *Newton–Euler* formulation has the following fundamental advantage:

- It is an inherently *recursive* method that is computationally efficient.

In the study of dynamics, it is relevant to find a solution to two kinds of problems concerning computation of direct dynamics and inverse dynamics.

The *direct dynamics* problem consists of determining, for $t > t_0$, the joint accelerations $\ddot{\boldsymbol{q}}(t)$ (and thus $\dot{\boldsymbol{q}}(t)$, $\boldsymbol{q}(t)$) resulting from the given joint torques $\boldsymbol{\tau}(t)$ — and the possible end-effector forces $\boldsymbol{h}_e(t)$ — once the initial positions $\boldsymbol{q}(t_0)$ and velocities $\dot{\boldsymbol{q}}(t_0)$ are known (initial state of the system).

The *inverse dynamics* problem consists of determining the joint torques $\boldsymbol{\tau}(t)$ which are needed to generate the motion specified by the joint accelerations $\ddot{\boldsymbol{q}}(t)$, velocities $\dot{\boldsymbol{q}}(t)$, and positions $\boldsymbol{q}(t)$ — once the possible end-effector forces $\boldsymbol{h}_e(t)$ are known.

Solving the direct dynamics problem is useful for manipulator *simulation*. Direct dynamics allows the motion of the real physical system to be described in terms of the joint accelerations, when a set of assigned joint torques is applied to the manipulator; joint velocities and positions can be obtained by integrating the system of nonlinear differential equations.

Since the equations of motion obtained with Lagrange formulation give the analytical relationship between the joint torques (and the end-effector forces) and the joint positions, velocities and accelerations, these can be computed from (7.42) as

$$\ddot{\boldsymbol{q}} = \boldsymbol{B}^{-1}(\boldsymbol{q})(\boldsymbol{\tau} - \boldsymbol{\tau}') \tag{7.115}$$

where

$$\boldsymbol{\tau}'(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}_v\dot{\boldsymbol{q}} + \boldsymbol{F}_s\,\mathbf{sgn}\,(\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{h}_e \tag{7.116}$$

denotes the torque contributions depending on joint positions and velocities. Therefore, for simulation of manipulator motion, once the state at the time instant t_k is known in terms of the position $\boldsymbol{q}(t_k)$ and velocity $\dot{\boldsymbol{q}}(t_k)$, the acceleration $\ddot{\boldsymbol{q}}(t_k)$ can be computed by (7.115). Then using a numerical integration method, e.g., Runge–Kutta, with integration step Δt, the velocity $\dot{\boldsymbol{q}}(t_{k+1})$ and position $\boldsymbol{q}(t_{k+1})$ at the instant $t_{k+1} = t_k + \Delta t$ can be computed.

If the equations of motion are obtained with Newton–Euler formulation, it is possible to compute direct dynamics by using a computationally more efficient method. In fact, for given $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$, the torques $\boldsymbol{\tau}'(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in (7.116) can be computed as the torques given by the algorithm of Fig. 7.14 with $\ddot{\boldsymbol{q}} = \boldsymbol{0}$. Further, column $\boldsymbol{b}_i$ of matrix $\boldsymbol{B}(\boldsymbol{q})$ can be computed as the torque vector given by the algorithm of Fig. 7.14 with $\boldsymbol{g}_0 = \boldsymbol{0}$, $\dot{\boldsymbol{q}} = \boldsymbol{0}$, $\ddot{q}_i = 1$ and $\ddot{q}_j = 0$ for $j \neq i$; iterating this procedure for $i = 1, \ldots, n$ leads to constructing the matrix $\boldsymbol{B}(\boldsymbol{q})$. Hence, from the current values of $\boldsymbol{B}(\boldsymbol{q})$ and $\boldsymbol{\tau}'(\boldsymbol{q}, \dot{\boldsymbol{q}})$, and the given $\boldsymbol{\tau}$, the equations in (7.115) can be integrated as illustrated above.

Solving the inverse dynamics problem is useful for manipulator trajectory planning and control algorithm implementation. Once a joint trajectory is specified in terms of positions, velocities and accelerations (typically as a result of an inverse kinematics procedure), and if the end-effector forces are known, inverse dynamics allows computation of the torques to be applied to the joints to obtain the desired motion. This computation turns out to be useful both for verifying feasibility of the imposed trajectory and for compensating nonlinear terms in the dynamic model of a manipulator. To this end, Newton–Euler formulation provides a computationally efficient recursive method for on-line computation of inverse dynamics. Nevertheless, it can be shown that also Lagrange formulation is liable to a computationally efficient recursive implementation, though with a nonnegligible reformulation effort.

For an n-joint manipulator the *number of operations* required is:[8]

[8] See Sect. E.1 for the definition of computational complexity of an algorithm.

- $O(n^2)$ for computing *direct dynamics*,
- $O(n)$ for computing *inverse dynamics*.

7.7 Dynamic Scaling of Trajectories

The existence of *dynamic constraints* to be taken into account for trajectory generation has been mentioned in Sect. 4.1. In practice, with reference to the given trajectory time or path shape (segments with high curvature), the trajectories that can be obtained with any of the previously illustrated methods may impose too severe dynamic performance for the manipulator. A typical case is that when the required torques to generate the motion are larger than the maximum torques the actuators can supply. In this case, an infeasible trajectory has to be suitably time-scaled.

Suppose a trajectory has been generated for all the manipulator joints as $\boldsymbol{q}(t)$, for $t \in [0, t_f]$. Computing inverse dynamics allows the evaluation of the time history of the torques $\boldsymbol{\tau}(t)$ required for the execution of the given motion. By comparing the obtained torques with the *torque limits* available at the actuators, it is easy to check whether or not the trajectory is actually executable. The problem is then to seek an automatic trajectory *dynamic scaling* technique — avoiding inverse dynamics recomputation — so that the manipulator can execute the motion on the specified path with a proper timing law without exceeding the torque limits.

Consider the manipulator dynamic model as given in (7.42) with $\boldsymbol{F}_v = \boldsymbol{O}$, $\boldsymbol{F}_s = \boldsymbol{O}$ and $\boldsymbol{h}_e = \boldsymbol{0}$, for simplicity. The term $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ accounting for centrifugal and Coriolis forces has a quadratic dependence on joint velocities, and thus it can be formally rewritten as

$$\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{\Gamma}(\boldsymbol{q})[\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}], \tag{7.117}$$

where $[\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}]$ indicates the symbolic notation of the $(n(n+1)/2 \times 1)$ vector

$$[\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}] = [\,\dot{q}_1^2 \quad \dot{q}_1\dot{q}_2 \quad \ldots \quad \dot{q}_{n-1}\dot{q}_n \quad \dot{q}_n^2\,]^T;$$

$\boldsymbol{\Gamma}(\boldsymbol{q})$ is a proper $(n \times n(n+1)/2)$ matrix that satisfies (7.117). In view of such position, the manipulator dynamic model can be expressed as

$$\boldsymbol{B}(\boldsymbol{q}(t))\ddot{\boldsymbol{q}}(t) + \boldsymbol{\Gamma}(\boldsymbol{q}(t))[\dot{\boldsymbol{q}}(t)\dot{\boldsymbol{q}}(t)] + \boldsymbol{g}(\boldsymbol{q}(t)) = \boldsymbol{\tau}(t), \tag{7.118}$$

where the explicit dependence on time t has been shown.

Consider the new variable $\bar{\boldsymbol{q}}(r(t))$ satisfying the equation

$$\boldsymbol{q}(t) = \bar{\boldsymbol{q}}(r(t)), \tag{7.119}$$

where $r(t)$ is a strictly increasing scalar function of time with $r(0) = 0$ and $r(t_f) = \bar{t}_f$.

Differentiating (7.119) twice with respect to time provides the following relations:

$$\dot{\boldsymbol{q}} = \dot{r}\bar{\boldsymbol{q}}'(r) \tag{7.120}$$

$$\ddot{\boldsymbol{q}} = \dot{r}^2\bar{\boldsymbol{q}}''(r) + \ddot{r}\bar{\boldsymbol{q}}'(r) \tag{7.121}$$

where the prime denotes the derivative with respect to r. Substituting (7.120), (7.121) into (7.118) yields

$$\dot{r}^2\Big(\boldsymbol{B}(\bar{\boldsymbol{q}}(r))\bar{\boldsymbol{q}}''(r) + \boldsymbol{\Gamma}(\bar{\boldsymbol{q}}(r))[\bar{\boldsymbol{q}}'(r)\bar{\boldsymbol{q}}'(r)]\Big) + \ddot{r}\boldsymbol{B}(\bar{\boldsymbol{q}}(r))\bar{\boldsymbol{q}}'(r) + \boldsymbol{g}(\bar{\boldsymbol{q}}(r)) = \boldsymbol{\tau}. \tag{7.122}$$

In (7.118) it is possible to identify the term

$$\boldsymbol{\tau}_s(t) = \boldsymbol{B}(\boldsymbol{q}(t))\ddot{\boldsymbol{q}}(t) + \boldsymbol{\Gamma}(\boldsymbol{q}(t))[\dot{\boldsymbol{q}}(t)\dot{\boldsymbol{q}}(t)], \tag{7.123}$$

representing the torque contribution that depends on velocities and accelerations. Correspondingly, in (7.122) one can set

$$\boldsymbol{\tau}_s(t) = \dot{r}^2\Big(\boldsymbol{B}(\bar{\boldsymbol{q}}(r))\bar{\boldsymbol{q}}''(r) + \boldsymbol{\Gamma}(\bar{\boldsymbol{q}}(r))[\bar{\boldsymbol{q}}'(r)\bar{\boldsymbol{q}}'(r)]\Big) + \ddot{r}\boldsymbol{B}(\bar{\boldsymbol{q}}(r))\bar{\boldsymbol{q}}'(r). \tag{7.124}$$

By analogy with (7.123), it can be written

$$\bar{\boldsymbol{\tau}}_s(r) = \boldsymbol{B}(\bar{\boldsymbol{q}}(r))\bar{\boldsymbol{q}}''(r) + \boldsymbol{\Gamma}(\bar{\boldsymbol{q}}(r))[\bar{\boldsymbol{q}}'(r)\bar{\boldsymbol{q}}'(r)] \tag{7.125}$$

and then (7.124) becomes

$$\boldsymbol{\tau}_s(t) = \dot{r}^2\bar{\boldsymbol{\tau}}_s(r) + \ddot{r}\boldsymbol{B}(\bar{\boldsymbol{q}}(r))\bar{\boldsymbol{q}}'(r). \tag{7.126}$$

The expression in (7.126) gives the relationship between the torque contributions depending on velocities and accelerations required by the manipulator when this is subject to motions having the same path but different timing laws, obtained through a time scaling of joint variables as in (7.119).

Gravitational torques have not been considered, since they are a function of the joint positions only, and thus their contribution is not influenced by time scaling.

The simplest choice for the scaling function $r(t)$ is certainly the *linear* function

$$r(t) = ct$$

with c a positive constant. In this case, (7.126) becomes

$$\boldsymbol{\tau}_s(t) = c^2\bar{\boldsymbol{\tau}}_s(ct),$$

which reveals that a linear time scaling by c causes a scaling of the magnitude of the torques by the coefficient c^2. Let $c > 1$: (7.119) shows that the trajectory described by $\bar{\boldsymbol{q}}(r(t))$, assuming $r = ct$ as the independent variable, has a duration $\bar{t}_f > t_f$ to cover the entire path specified by $\boldsymbol{q}$. Correspondingly, the

torque contributions $\bar{\boldsymbol{\tau}}_s(ct)$ computed as in (7.125) are scaled by the factor c^2 with respect to the torque contributions $\boldsymbol{\tau}_s(t)$ required to execute the original trajectory $\boldsymbol{q}(t)$.

With the use of a recursive algorithm for inverse dynamics computation, it is possible to check whether the torques exceed the allowed limits during trajectory execution; obviously, limit violation should not be caused by the sole gravity torques. It is necessary to find the joint for which the torque has exceeded the limit more than the others, and to compute the torque contribution subject to scaling, which in turn determines the factor c^2. It is then possible to compute the time-scaled trajectory as a function of the new time variable $r = ct$ which no longer exceeds torque limits. It should be pointed out, however, that with this kind of linear scaling the entire trajectory may be penalized, even when a torque limit on a single joint is exceeded only for a short interval of time.

7.8 Operational Space Dynamic Model

As an alternative to the joint space dynamic model, the equations of motion of the system can be expressed directly in the operational space; to this end it is necessary to find a *dynamic model* which describes the relationship between the generalized forces acting on the manipulator and the number of minimal variables chosen to describe the end-effector position and orientation in the *operational space*.

Similar to kinematic description of a manipulator in the operational space, the presence of redundant DOFs and/or kinematic and representation singularities deserves careful attention in the derivation of an operational space dynamic model.

The determination of the dynamic model with Lagrange formulation using operational space variables allows a complete description of the system motion only in the case of a *nonredundant* manipulator, when the above variables constitute a set of *generalized coordinates* in terms of which the kinetic energy, the potential energy, and the nonconservative forces doing work on them can be expressed.

This way of proceeding does not provide a complete description of dynamics for a *redundant* manipulator; in this case, in fact, it is reasonable to expect the occurrence of *internal motions* of the structure caused by those joint generalized forces which do not affect the end-effector motion.

To develop an operational space model which can be adopted for both redundant and nonredundant manipulators, it is then convenient to start from the joint space model which is in all general. In fact, solving (7.42) for the joint accelerations, and neglecting the joint friction torques for simplicity, yields

$$\ddot{\boldsymbol{q}} = -\boldsymbol{B}^{-1}(\boldsymbol{q})\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{B}^{-1}(\boldsymbol{q})\boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{B}^{-1}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})(\boldsymbol{\gamma}_e - \boldsymbol{h}_e), \tag{7.127}$$

where the joint torques $\boldsymbol{\tau}$ have been expressed in terms of the equivalent end-effector forces $\boldsymbol{\gamma}$ according to (3.111). It is worth noting that $\boldsymbol{h}$ represents the contribution of the end-effector forces due to contact with the environment, whereas $\boldsymbol{\gamma}$ expresses the contribution of the end-effector forces due to joint actuation.

On the other hand, the second-order differential kinematics equation in (3.98) describes the relationship between joint space and operational space accelerations, i.e.,

$$\ddot{\boldsymbol{x}}_e = \boldsymbol{J}_A(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_A(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}.$$

The solution in (7.127) features the geometric Jacobian $\boldsymbol{J}$, whereas the analytical Jacobian $\boldsymbol{J}_A$ appears in (3.98). For notation uniformity, in view of (3.66), one can set

$$\boldsymbol{T}_A^T(\boldsymbol{x}_e)\boldsymbol{\gamma}_e = \boldsymbol{\gamma}_A \qquad \boldsymbol{T}_A^T(\boldsymbol{x}_e)\boldsymbol{h}_e = \boldsymbol{h}_A \tag{7.128}$$

where $\boldsymbol{T}_A$ is the transformation matrix between the two Jacobians. Substituting (7.127) into (3.98) and accounting for (7.128) gives

$$\ddot{\boldsymbol{x}}_e = -\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{g} + \dot{\boldsymbol{J}}_A\dot{\boldsymbol{q}} + \boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{J}_A^T(\boldsymbol{\gamma}_A - \boldsymbol{h}_A). \tag{7.129}$$

where the dependence on $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ has been omitted. With the positions

$$\boldsymbol{B}_A = (\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{J}_A^T)^{-1} \tag{7.130}$$

$$\boldsymbol{C}_A\dot{\boldsymbol{x}}_e = \boldsymbol{B}_A\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{B}_A\dot{\boldsymbol{J}}_A\dot{\boldsymbol{q}} \tag{7.131}$$

$$\boldsymbol{g}_A = \boldsymbol{B}_A\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{g}, \tag{7.132}$$

the expression in (7.129) can be rewritten as

$$\boldsymbol{B}_A(\boldsymbol{x}_e)\ddot{\boldsymbol{x}}_e + \boldsymbol{C}_A(\boldsymbol{x}_e, \dot{\boldsymbol{x}}_e)\dot{\boldsymbol{x}}_e + \boldsymbol{g}_A(\boldsymbol{x}_e) = \boldsymbol{\gamma}_A - \boldsymbol{h}_A, \tag{7.133}$$

which is formally analogous to the joint space dynamic model (7.42). Notice that the matrix $\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{J}_A^T$ is invertible if and only if $\boldsymbol{J}_A$ is full-rank, that is, in the absence of both kinematic and representation singularities.

For a nonredundant manipulator in a nonsingular configuration, the expressions in (7.130)–(7.132) become:

$$\boldsymbol{B}_A = \boldsymbol{J}_A^{-T}\boldsymbol{B}\boldsymbol{J}_A^{-1} \tag{7.134}$$

$$\boldsymbol{C}_A\dot{\boldsymbol{x}}_e = \boldsymbol{J}_A^{-T}\boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{B}_A\dot{\boldsymbol{J}}_A\dot{\boldsymbol{q}} \tag{7.135}$$

$$\boldsymbol{g}_A = \boldsymbol{J}_A^{-T}\boldsymbol{g}. \tag{7.136}$$

As anticipated above, the main feature of the obtained model is its formal validity also for a redundant manipulator, even though the variables $\boldsymbol{x}_e$ do not constitute a set of generalized coordinates for the system; in this case, the matrix $\boldsymbol{B}_A$ is representative of a *kinetic pseudo-energy*.

In the following, the utility of the operational space dynamic model in (7.133) for solving direct and inverse dynamics problems is investigated. The

following derivation is meaningful for redundant manipulators; for a nonredundant manipulator, in fact, using (7.133) does not pose specific problems as long as $\boldsymbol{J}_A$ is nonsingular ((7.134)–(7.136)).

With reference to operational space, the *direct dynamics* problem consists of determining the resulting end-effector accelerations $\ddot{\boldsymbol{x}}_e(t)$ (and thus $\dot{\boldsymbol{x}}_e(t)$, $\boldsymbol{x}_e(t)$) from the given joint torques $\boldsymbol{\tau}(t)$ and end-effector forces $\boldsymbol{h}_e(t)$. For a redundant manipulator, (7.133) cannot be directly used, since (3.111) has a solution in $\boldsymbol{\gamma}_e$ only if $\boldsymbol{\tau} \in \mathcal{R}(\boldsymbol{J}^T)$. It follows that for simulation purposes, the solution to the problem is naturally obtained in the joint space; in fact, the expression in (7.42) allows the computation of $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$ which, substituted into the direct kinematics equations in ((2.82), (3.62), (3.98), give $\boldsymbol{x}_e$, $\dot{\boldsymbol{x}}_e$, $\ddot{\boldsymbol{x}}_e$, respectively.

Formulation of an *inverse dynamics* problem in the operational space requires the determination of the joint torques $\boldsymbol{\tau}(t)$ that are needed to generate a specific motion assigned in terms of $\ddot{\boldsymbol{x}}_e(t)$, $\dot{\boldsymbol{x}}_e(t)$, $\boldsymbol{x}_e(t)$, for given end-effector forces $\boldsymbol{h}_e(t)$. A possible way of solution is to solve a complete inverse kinematics problem for (2.82), (3.62), (3.98), and then compute the required torques with the joint space inverse dynamics as in (7.42). Hence, for redundant manipulators, redundancy resolution is performed at kinematic level.

An alternative solution to the inverse dynamics problem consists of computing $\boldsymbol{\gamma}_A$ as in (7.133) and the joint torques $\boldsymbol{\tau}$ as in (3.111). In this way, however, the presence of redundant DOFs is not exploited at all, since the computed torques do not generate internal motions of the structure.

If it is desired to find a formal solution that allows redundancy resolution at dynamic level, it is necessary to determine those torques corresponding to the equivalent end-effector forces computed as in (7.133). By analogy with the differential kinematics solution (3.54), the expression of the torques to be determined will feature the presence of a minimum-norm term and a homogeneous term. Since the joint torques have to be computed, it is convenient to express the model (7.133) in terms of $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$. By recalling the positions (7.131), (7.132), the expression in (7.133) becomes

$$\boldsymbol{B}_A(\ddot{\boldsymbol{x}}_e - \dot{\boldsymbol{J}}_A\dot{\boldsymbol{q}}) + \boldsymbol{B}_A\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{B}_A\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{g} = \boldsymbol{\gamma}_A - \boldsymbol{h}_A$$

and, in view of (3.98),

$$\boldsymbol{B}_A\boldsymbol{J}_A\ddot{\boldsymbol{q}} + \boldsymbol{B}_A\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{B}_A\boldsymbol{J}_A\boldsymbol{B}^{-1}\boldsymbol{g} = \boldsymbol{\gamma}_A - \boldsymbol{h}_A. \tag{7.137}$$

By setting

$$\bar{\boldsymbol{J}}_A(\boldsymbol{q}) = \boldsymbol{B}^{-1}(\boldsymbol{q})\boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{B}_A(\boldsymbol{q}), \tag{7.138}$$

the expression in (7.137) becomes

$$\bar{\boldsymbol{J}}_A^T(\boldsymbol{B}\ddot{\boldsymbol{q}} + \boldsymbol{C}\dot{\boldsymbol{q}} + \boldsymbol{g}) = \boldsymbol{\gamma}_A - \boldsymbol{h}_A. \tag{7.139}$$

At this point, from the joint space dynamic model in (7.42), it is easy to recognize that (7.139) can be written as

$$\bar{\boldsymbol{J}}_A^T(\boldsymbol{\tau} - \boldsymbol{J}_A^T\boldsymbol{h}_A) = \boldsymbol{\gamma}_A - \boldsymbol{h}_A$$

from which

$$\bar{\boldsymbol{J}}_A^T \boldsymbol{\tau} = \boldsymbol{\gamma}_A. \tag{7.140}$$

The general solution to (7.140) is of the form (see Problem 7.10)

$$\boldsymbol{\tau} = \boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{\gamma}_A + \big(\boldsymbol{I}_n - \boldsymbol{J}_A^T(\boldsymbol{q})\bar{\boldsymbol{J}}_A^T(\boldsymbol{q})\big)\boldsymbol{\tau}_0, \tag{7.141}$$

that can be derived by observing that $\boldsymbol{J}_A^T$ in (7.138) is a *right pseudo-inverse* of $\bar{\boldsymbol{J}}_A^T$ weighted by the inverse of the inertia matrix $\boldsymbol{B}^{-1}$. The $(n \times 1)$ vector of arbitrary torques $\boldsymbol{\tau}_0$ in (7.141) does not contribute to the end-effector forces, since it is projected in the null space of $\bar{\boldsymbol{J}}_A^T$.

To summarize, for given $\boldsymbol{x}_e$, $\dot{\boldsymbol{x}}_e$, $\ddot{\boldsymbol{x}}_e$ and $\boldsymbol{h}_A$, the expression in (7.133) allows the computation of $\boldsymbol{\gamma}_A$. Then, (7.141) gives the torques $\boldsymbol{\tau}$ which, besides executing the assigned end-effector motion, generate internal motions of the structure to be employed for handling redundancy at dynamic level through a suitable choice of $\boldsymbol{\tau}_0$.

7.9 Dynamic Manipulability Ellipsoid

The availability of the dynamic model allows formulation of the *dynamic manipulability ellipsoid* which provides a useful tool for manipulator dynamic performance analysis. This can be used for mechanical structure design as well as for seeking optimal manipulator configurations.

Consider the set of joint torques of constant (unit) norm

$$\boldsymbol{\tau}^T \boldsymbol{\tau} = 1 \tag{7.142}$$

describing the points on the surface of a sphere. It is desired to describe the operational space accelerations that can be generated by the given set of joint torques.

For studying dynamic manipulability, suppose to consider the case of a manipulator standing still ($\dot{\boldsymbol{q}} = \boldsymbol{0}$), not in contact with the environment ($\boldsymbol{h}_e = \boldsymbol{0}$). The simplified model is

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}. \tag{7.143}$$

The joint accelerations $\ddot{\boldsymbol{q}}$ can be computed from the second-order differential kinematics that can be obtained by differentiating (3.39), and imposing successively $\dot{\boldsymbol{q}} = \boldsymbol{0}$, leading to

$$\dot{\boldsymbol{v}}_e = \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}}. \tag{7.144}$$

Solving for minimum-norm accelerations only, for a *nonsingular Jacobian*, and substituting in (7.143) yields the expression of the torques

$$\boldsymbol{\tau} = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{J}^{\dagger}(\boldsymbol{q})\dot{\boldsymbol{v}}_e + \boldsymbol{g}(\boldsymbol{q}) \tag{7.145}$$

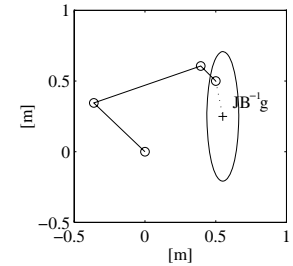

Fig. 7.15. Effect of gravity on the dynamic manipulability ellipsoid for a three-link planar arm

needed to derive the ellipsoid. In fact, substituting (7.145) into (7.142) gives

$$\big(\boldsymbol{B}(\boldsymbol{q})\boldsymbol{J}^{\dagger}(\boldsymbol{q})\dot{\boldsymbol{v}}_e + \boldsymbol{g}(\boldsymbol{q})\big)^T \big(\boldsymbol{B}(\boldsymbol{q})\boldsymbol{J}^{\dagger}(\boldsymbol{q})\dot{\boldsymbol{v}}_e + \boldsymbol{g}(\boldsymbol{q})\big) = 1.$$

The vector on the right-hand side of (7.145) can be rewritten as

$$\begin{aligned} \boldsymbol{B}\boldsymbol{J}^{\dagger}\dot{\boldsymbol{v}}_e + \boldsymbol{g} &= \boldsymbol{B}(\boldsymbol{J}^{\dagger}\dot{\boldsymbol{v}}_e + \boldsymbol{B}^{-1}\boldsymbol{g}) \\ &= \boldsymbol{B}(\boldsymbol{J}^{\dagger}\dot{\boldsymbol{v}}_e + \boldsymbol{B}^{-1}\boldsymbol{g} + \boldsymbol{J}^{\dagger}\boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g} - \boldsymbol{J}^{\dagger}\boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g}) \\ &= \boldsymbol{B}\big(\boldsymbol{J}^{\dagger}\dot{\boldsymbol{v}}_e + \boldsymbol{J}^{\dagger}\boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g} + (\boldsymbol{I}_n - \boldsymbol{J}^{\dagger}\boldsymbol{J})\boldsymbol{B}^{-1}\boldsymbol{g}\big), \end{aligned} \tag{7.146}$$

where the dependence on $\boldsymbol{q}$ has been omitted. According to what was done for solving (7.144), one can neglect the contribution of the accelerations given by $\boldsymbol{B}^{-1}\boldsymbol{g}$ which are in the null space of $\boldsymbol{J}$ and then produce no end-effector acceleration. Hence, (7.146) becomes

$$\boldsymbol{B}\boldsymbol{J}^{\dagger}\dot{\boldsymbol{v}}_e + \boldsymbol{g} = \boldsymbol{B}\boldsymbol{J}^{\dagger}(\dot{\boldsymbol{v}}_e + \boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g}) \tag{7.147}$$

and the dynamic manipulability ellipsoid can be expressed in the form

$$(\dot{\boldsymbol{v}}_e + \boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g})^T \boldsymbol{J}^{\dagger T}\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{J}^{\dagger}(\dot{\boldsymbol{v}}_e + \boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g}) = 1. \tag{7.148}$$

The core of the quadratic form $\boldsymbol{J}^{\dagger T}\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{J}^{\dagger}$ depends on the geometrical and inertial characteristics of the manipulator and determines the volume and principal axes of the ellipsoid. The vector $-\boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g}$, describing the contribution of gravity, produces a constant translation of the centre of the ellipsoid (for each manipulator configuration) with respect to the origin of the reference frame; see the example in Fig. 7.15 for a three-link planar arm.

The meaning of the dynamic manipulability ellipsoid is conceptually similar to that of the ellipsoids considered with reference to kineto-statics duality. In fact, the distance of a point on the surface of the ellipsoid from the end-effector gives a measure of the accelerations which can be imposed to the end-effector along the given direction, with respect to the constraint (7.142). With reference to Fig. 7.15, it is worth noticing how the presence of gravity

acceleration allows the execution of larger accelerations downward, as natural to predict.

In the case of a nonredundant manipulator, the ellipsoid reduces to

$$(\dot{\boldsymbol{v}}_e + \boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g})^T \boldsymbol{J}^{-T}\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{J}^{-1}(\dot{\boldsymbol{v}}_e + \boldsymbol{J}\boldsymbol{B}^{-1}\boldsymbol{g}) = 1. \tag{7.149}$$

Bibliography

The derivation of the dynamic model for rigid manipulators can be found in several classical robotics texts, such as [180, 10, 248, 53, 217, 111].

The first works on the computation of the dynamic model of open-chain manipulators based on the Lagrange formulation are [234, 19, 221, 236]. A computationally efficient formulation is presented in [96].

Dynamic model computation for robotic systems having a closed-chain or a tree kinematic structure can be found in [11, 144] and [112], respectively. Joint friction models are analyzed in [9].

The notable properties of the dynamic model deriving from the principle of energy conservation are underlined in [213], on the basis of the work in [119]. Algorithms to find the parameterization of the dynamic model in terms of a minimum number of parameters are considered in [115], which utilizes the results in [166]. Methods for symbolic computation of those parameters are presented in [85] for open kinematic chains and [110] for closed kinematic chains. Parameter identification methods based on least-squares techniques are given in [13].

The Newton–Euler formulation is proposed in [172], and a computationally efficient version for inverse dynamics can be found in [142]; an analogous formulation is employed for direct dynamics computation in [237]. The Lagrange and Newton–Euler formulations are compared by a computational viewpoint in [211], while they are utilized in [201] for dynamic model computation with inclusion of inertial and gyroscopic effects of actuators. Efficient algorithms for direct dynamics computation are given in [76, 77].

The trajectory dynamic scaling technique is presented in [97]. The operational space dynamic model is illustrated in [114] and the concept of weighted pseudo-inverse of the inertia matrix is introduced in [78]. The manipulability ellipsoids are analyzed in [246, 38].

Problems

7.1. Find the dynamic model of a two-link Cartesian arm in the case when the second joint axis forms an angle of $\pi/4$ with the first joint axis; compare the result with the model of the manipulator in Fig. 7.3.

7.2. For the two-link planar arm of Sect. 7.3.2, prove that with a different choice of the matrix $\boldsymbol{C}$, (7.49) holds true while (7.48) does not.

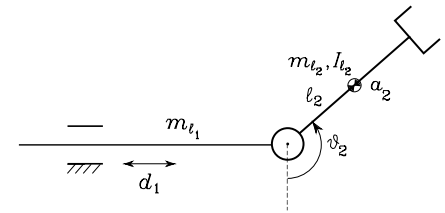

Fig. 7.16. Two-link planar arm with a prismatic joint and a revolute joint

7.3. Find the dynamic model of the SCARA manipulator in Fig. 2.36.

7.4. For the planar arm of Sect. 7.3.2, find a minimal parameterization of the dynamic model in (7.82).

7.5. Find the dynamic model of the two-link planar arm with a prismatic joint and a revolute joint in Fig. 7.16 with the Lagrange formulation. Then, consider the addition of a concentrated tip payload of mass m_L, and express the resulting model in a linear form with respect to a suitable set of dynamic parameters as in (7.81).

7.6. For the two-link planar arm of Fig. 7.4, find the dynamic model with the Lagrange formulation when the absolute angles with respect to the base frame are chosen as generalized coordinates. Discuss the result in view of a comparison with the model derived in (7.82).

7.7. Compute the joint torques for the two-link planar arm of Fig. 7.4 with the data and along the trajectories of Example 7.2, in the case of tip forces $\boldsymbol{f} = [\,500 \quad 500\,]^T$ N.

7.8. Find the dynamic model of the two-link planar arm with a prismatic joint and a revolute joint in Fig. 7.16 by using the recursive Newton–Euler algorithm.

7.9. Show that for the operational space dynamic model (7.133) a skew-symmetry property holds which is analogous to (7.48).

7.10. Show how to obtain the general solution to (7.140) in the form (7.141).

7.11. For a nonredundant manipulator, compute the relationship between the dynamic manipulability measure that can be defined for the dynamic manipulability ellipsoid and the manipulability measure defined in (3.56).

8

Motion Control

In Chap. 4, trajectory planning techniques have been presented which allow the generation of the reference inputs to the motion control system. The problem of controlling a manipulator can be formulated as that to determine the time history of the generalized forces (forces or torques) to be developed by the joint actuators, so as to guarantee execution of the commanded task while satisfying given transient and steady-state requirements. The task may regard either the execution of specified motions for a manipulator operating in free space, or the execution of specified motions and contact forces for a manipulator whose end-effector is constrained by the environment. In view of problem complexity, the two aspects will be treated separately; first, motion control in free space, and then control of the interaction with the environment. The problem of *motion control* of a manipulator is the topic of this chapter. A number of *joint space* control techniques are presented. These can be distinguished between *decentralized control* schemes, i.e., when the single manipulator joint is controlled independently of the others, and *centralized control* schemes, i.e., when the dynamic interaction effects between the joints are taken into account. Finally, as a premise to the interaction control problem, the basic features of *operational space* control schemes are illustrated.

8.1 The Control Problem

Several techniques can be employed for controlling a manipulator. The technique followed, as well as the way it is implemented, may have a significant influence on the manipulator performance and then on the possible range of applications. For instance, the need for trajectory tracking control in the operational space may lead to hardware/software implementations, which differ from those allowing point-to-point control, where only reaching of the final position is of concern.

On the other hand, the manipulator mechanical design has an influence on the kind of control scheme utilized. For instance, the control problem of

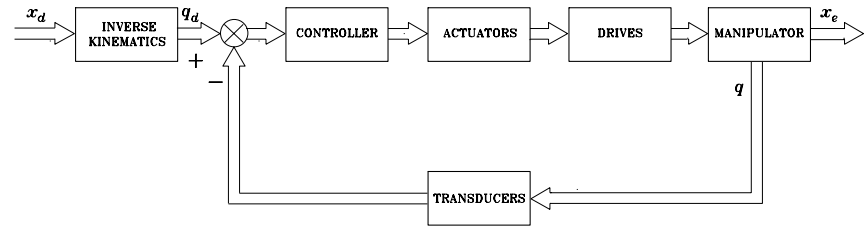

Fig. 8.1. General scheme of joint space control

a Cartesian manipulator is substantially different from that of an anthropomorphic manipulator.

The driving system of the joints also has an effect on the type of control strategy used. If a manipulator is actuated by electric motors with reduction gears of high ratios, the presence of gears tends to linearize system dynamics, and thus to decouple the joints in view of the reduction of nonlinearity effects. The price to pay, however, is the occurrence of joint friction, elasticity and backlash that may limit system performance more than it is due to configuration-dependent inertia, Coriolis and centrifugal forces, and so forth. On the other hand, a robot actuated with direct drives eliminates the drawbacks due to friction, elasticity and backlash, but the weight of nonlinearities and couplings between the joints becomes relevant. As a consequence, different control strategies have to be thought of to obtain high performance.

Without any concern to the specific type of mechanical manipulator, it is worth remarking that task specification (end-effector motion and forces) is usually carried out in the operational space, whereas control actions (joint actuator generalized forces) are performed in the joint space. This fact naturally leads to considering two kinds of general control schemes, namely, a *joint space control* scheme (Fig. 8.1) and an *operational space control* scheme (Fig. 8.2). In both schemes, the control structure has closed loops to exploit the good features provided by feedback, i.e., robustness to modelling uncertainties and reduction of disturbance effects. In general terms, the following considerations should be made.

The *joint space control* problem is actually articulated in two subproblems. First, manipulator inverse kinematics is solved to transform the motion requirements $\boldsymbol{x}_d$ from the operational space into the corresponding motion $\boldsymbol{q}_d$ in the joint space. Then, a joint space control scheme is designed that allows the actual motion $\boldsymbol{q}$ to track the reference inputs. However, this solution has the drawback that a joint space control scheme does not influence the operational space variables $\boldsymbol{x}_e$ which are controlled in an open-loop fashion through the manipulator mechanical structure. It is then clear that any uncertainty of the structure (construction tolerance, lack of calibration, gear backlash, elasticity) or any imprecision in the knowledge of the end-effector pose relative

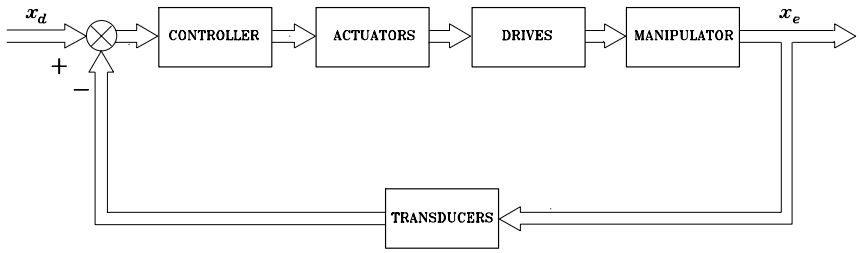

Fig. 8.2. General scheme of operational space control

to an object to manipulate causes a loss of accuracy on the operational space variables.

The *operational space control* problem follows a global approach that requires a greater algorithmic complexity; notice that inverse kinematics is now embedded into the feedback control loop. Its conceptual advantage regards the possibility of acting directly on operational space variables; this is somewhat only a potential advantage, since measurement of operational space variables is often performed not directly, but through the evaluation of direct kinematics functions starting from measured joint space variables.

On the above premises, in the following, joint space control schemes for manipulator motion in the free space are presented first. In the sequel, operational space control schemes will be illustrated which are logically at the basis of control of the interaction with the environment.

8.2 Joint Space Control

In Chap. 7, it was shown that the equations of motion of a manipulator in the absence of external end-effector forces and, for simplicity, of static friction (difficult to model accurately) are described by

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}_v\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} \tag{8.1}$$

with obvious meaning of the symbols. To control the motion of the manipulator in free space means to determine the n components of generalized forces — torques for revolute joints, forces for prismatic joints — that allow execution of a motion $\boldsymbol{q}(t)$ so that

$$\boldsymbol{q}(t) = \boldsymbol{q}_d(t),$$

as closely as possible, where $\boldsymbol{q}_d(t)$ denotes the vector of desired joint trajectory variables.

The generalized forces are supplied by the actuators through proper transmissions to transform the motion characteristics. Let $\boldsymbol{q}_m$ denote the vector of joint actuator displacements; the transmissions — assumed to be rigid and with no backlash — establish the following relationship:

$$\boldsymbol{K}_r\boldsymbol{q} = \boldsymbol{q}_m, \tag{8.2}$$

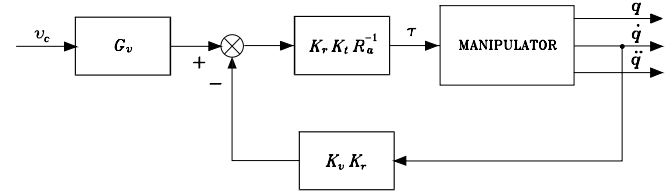

Fig. 8.3. Block scheme of the manipulator and drives system as a voltage-controlled system

where $\boldsymbol{K}_r$ is an $(n \times n)$ diagonal matrix, whose elements are defined in (7.22) and are much greater than unity.[1]

In view of (8.2), if $\boldsymbol{\tau}_m$ denotes the vector of actuator driving torques, one can write

$$\boldsymbol{\tau}_m = \boldsymbol{K}_r^{-1}\boldsymbol{\tau}. \tag{8.3}$$

With reference to (5.1)–(5.4), the n driving systems can be described in compact matrix form by the equations:

$$\boldsymbol{K}_r^{-1}\boldsymbol{\tau} = \boldsymbol{K}_t\boldsymbol{i}_a \tag{8.4}$$

$$\boldsymbol{v}_a = \boldsymbol{R}_a\boldsymbol{i}_a + \boldsymbol{K}_v\dot{\boldsymbol{q}}_m \tag{8.5}$$

$$\boldsymbol{v}_a = \boldsymbol{G}_v\boldsymbol{v}_c. \tag{8.6}$$

In (8.4), $\boldsymbol{K}_t$ is the diagonal matrix of torque constants and $\boldsymbol{i}_a$ is the vector of armature currents of the n motors; in (8.5), $\boldsymbol{v}_a$ is the vector of armature voltages, $\boldsymbol{R}_a$ is the diagonal matrix of armature resistances,[2] and $\boldsymbol{K}_v$ is the diagonal matrix of voltage constants of the n motors; in (8.6), $\boldsymbol{G}_v$ is the diagonal matrix of gains of the n amplifiers and $\boldsymbol{v}_c$ is the vector of control voltages of the n servomotors.

On reduction of (8.1), (8.2), (8.4), (8.5), (8.6), the dynamic model of the system given by the manipulator and drives is described by

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u} \tag{8.7}$$

where the following positions have been made:

$$\boldsymbol{F} = \boldsymbol{F}_v + \boldsymbol{K}_r\boldsymbol{K}_t\boldsymbol{R}_a^{-1}\boldsymbol{K}_v\boldsymbol{K}_r \tag{8.8}$$

$$\boldsymbol{u} = \boldsymbol{K}_r\boldsymbol{K}_t\boldsymbol{R}_a^{-1}\boldsymbol{G}_v\boldsymbol{v}_c. \tag{8.9}$$

From (8.1), (8.7), (8.8), (8.9) it is

$$\boldsymbol{K}_r\boldsymbol{K}_t\boldsymbol{R}_a^{-1}\boldsymbol{G}_v\boldsymbol{v}_c = \boldsymbol{\tau} + \boldsymbol{K}_r\boldsymbol{K}_t\boldsymbol{R}_a^{-1}\boldsymbol{K}_v\boldsymbol{K}_r\dot{\boldsymbol{q}} \tag{8.10}$$

[1] Assuming a diagonal $\boldsymbol{K}_r$ leads to excluding the presence of kinematic couplings in the transmission, that is the motion of each actuator does not induce motion on a joint other than that actuated.

[2] The contribution of the inductance has been neglected.

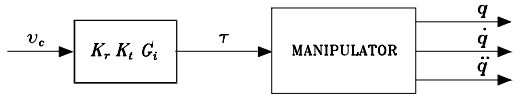

Fig. 8.4. Block scheme of the manipulator and drives system as a torque-controlled system

and thus

$$\boldsymbol{\tau} = \boldsymbol{K}_r \boldsymbol{K}_t \boldsymbol{R}_a^{-1} (\boldsymbol{G}_v \boldsymbol{v}_c - \boldsymbol{K}_v \boldsymbol{K}_r \dot{\boldsymbol{q}}). \tag{8.11}$$

The overall system is then *voltage-controlled* and the corresponding block scheme is illustrated in Fig. 8.3. If the following assumptions hold:

- the elements of matrix $\boldsymbol{K}_r$, characterizing the transmissions, are much greater than unity;
- the elements of matrix $\boldsymbol{R}_a$ are very small, which is typical in the case of high-efficiency servomotors;
- the values of the torques $\boldsymbol{\tau}$ required for the execution of the desired motions are not too large;

then it can be assumed that

$$\boldsymbol{G}_v \boldsymbol{v}_c \approx \boldsymbol{K}_v \boldsymbol{K}_r \dot{\boldsymbol{q}}. \tag{8.12}$$

The proportionality relationship obtained between $\dot{\boldsymbol{q}}$ and $\boldsymbol{v}_c$ is independent of the values attained by the manipulator parameters; the smaller the joint velocities and accelerations, the more valid this assumption. Hence, velocity (or voltage) control shows an inherent robustness with respect to parameter variations of the manipulator model, which is enhanced by the values of the gear reduction ratios.

In this case, the scheme illustrated in Fig. 8.3 can be taken as the reference structure for the design of the control system. Having assumed that

$$\boldsymbol{v}_c \approx \boldsymbol{G}_v^{-1} \boldsymbol{K}_v \boldsymbol{K}_r \dot{\boldsymbol{q}} \tag{8.13}$$

implies that the velocity of the i-th joint depends only on the i-th control voltage, since the matrix $\boldsymbol{G}_v^{-1} \boldsymbol{K}_v \boldsymbol{K}_r$ is diagonal. Therefore, the joint position control system can be designed according to a *decentralized control structure*, since each joint can be controlled independently of the others. The results, evaluated in the terms of the tracking accuracy of the joint variables with respect to the desired trajectories, are improved in the case of higher gear reduction ratios and less demanding values of required speeds and accelerations.

On the other hand, if the desired manipulator motion requires large joint speeds and/or accelerations, the approximation (8.12) no longer holds, in view of the magnitude of the required driving torques; this occurrence is even more evident for direct-drive actuation ($\boldsymbol{K}_r = \boldsymbol{I}$).

In this case, by resorting to an inverse dynamics technique, it is possible to find the joint torques $\boldsymbol{\tau}(t)$ needed to track any specified motion in terms of the joint accelerations $\ddot{\boldsymbol{q}}(t)$, velocities $\dot{\boldsymbol{q}}(t)$ and positions $\boldsymbol{q}(t)$. Obviously, this solution requires the accurate knowledge of the manipulator dynamic model. The determination of the torques to be generated by the drive system can thus refer to a *centralized control structure*, since to compute the torque history at the i-th joint it is necessary to know the time evolution of the motion of all the joints. By recalling that

$$\boldsymbol{\tau} = \boldsymbol{K}_r \boldsymbol{K}_t \boldsymbol{i}_a, \tag{8.14}$$

to find a relationship between the torques $\boldsymbol{\tau}$ and the control voltages $\boldsymbol{v}_c$, using (8.5), (8.6) leads to

$$\boldsymbol{\tau} = \boldsymbol{K}_r \boldsymbol{K}_t \boldsymbol{R}_a^{-1} \boldsymbol{G}_v \boldsymbol{v}_c - \boldsymbol{K}_r \boldsymbol{K}_t \boldsymbol{R}_a^{-1} \boldsymbol{K}_v \boldsymbol{K}_r \dot{\boldsymbol{q}}. \tag{8.15}$$

If the actuators have to provide torque contributions computed on the basis of the manipulator dynamic model, the control voltages — to be determined according to (8.15) — depend on the torque values and also on the joint velocities; this relationship depends on the matrices $\boldsymbol{K}_t$, $\boldsymbol{K}_v$ and $\boldsymbol{R}_a^{-1}$, whose elements are influenced by the operating conditions of the motors. To reduce sensitivity to parameter variations, it is worth considering driving systems characterized by a current control rather than by a voltage control. In this case the actuators behave as torque-controlled generators; the equation in (8.5) becomes meaningless and is replaced by

$$\boldsymbol{i}_a = \boldsymbol{G}_i \boldsymbol{v}_c, \tag{8.16}$$

which gives a proportional relation between the armature currents $\boldsymbol{i}_a$ (and thus the torques $\boldsymbol{\tau}$) and the control voltages $\boldsymbol{v}_c$ established by the constant matrix $\boldsymbol{G}_i$. As a consequence, (8.9) becomes

$$\boldsymbol{\tau} = \boldsymbol{u} = \boldsymbol{K}_r \boldsymbol{K}_t \boldsymbol{G}_i \boldsymbol{v}_c \tag{8.17}$$

which shows a reduced dependence of $\boldsymbol{u}$ on the motor parameters. The overall system is now *torque-controlled* and the resulting block scheme is illustrated in Fig. 8.4.

The above presentation suggests resorting for the decentralized structure — where the need for robustness prevails — to feedback control systems, while for the centralized structure — where the computation of inverse dynamics is needed — it is necessary to refer to control systems with feedforward actions. Nevertheless, it should be pointed out that centralized control still requires the use of error contributions between the desired and the actual trajectory, no matter whether they are implemented in a feedback or in a feedforward fashion. This is a consequence of the fact that the considered dynamic model, even though a quite complex one, is anyhow an idealization of reality which

does not include effects, such as joint Coulomb friction, gear backlash, dimension tolerance, and the simplifying assumptions in the model, e.g., link rigidity, and so on.

As already pointed out, the drive systems is anyhow inserted into a feedback control system. In the case of decentralized control, the drive will be characterized by the model describing its behaviour as a velocity-controlled generator. Instead, in the case of centralized control, since the driving torque is to be computed on a complete or reduced manipulator dynamic model, the drive will be characterized as a torque-controlled generator.

8.3 Decentralized Control

The simplest control strategy that can be thought of is one that regards the manipulator as formed by n independent systems (the n joints) and controls each joint axis as a *single-input/single-output system*. Coupling effects between joints due to varying configurations during motion are treated as *disturbance* inputs.

In order to analyze various control schemes and their performance, it is worth considering the model of the system manipulator with drives in terms of mechanical quantities at the motor side; in view of (8.2), (8.3), it is

$$\boldsymbol{K}_r^{-1}\boldsymbol{B}(\boldsymbol{q})\boldsymbol{K}_r^{-1}\ddot{\boldsymbol{q}}_m + \boldsymbol{K}_r^{-1}\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{K}_r^{-1}\dot{\boldsymbol{q}}_m + \boldsymbol{K}_r^{-1}\boldsymbol{F}_v\boldsymbol{K}_r^{-1} + \boldsymbol{K}_r^{-1}\boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}_m. \tag{8.18}$$

By observing that the diagonal elements of $\boldsymbol{B}(\boldsymbol{q})$ are formed by constant terms and configuration-dependent terms (functions of sine and cosine for revolute joints), one can set

$$\boldsymbol{B}(\boldsymbol{q}) = \bar{\boldsymbol{B}} + \Delta\boldsymbol{B}(\boldsymbol{q}) \tag{8.19}$$

where $\bar{\boldsymbol{B}}$ is the *diagonal* matrix whose constant elements represent the resulting average inertia at each joint. Substituting (8.19) into (8.1) yields

$$\boldsymbol{K}_r^{-1}\bar{\boldsymbol{B}}\boldsymbol{K}_r^{-1}\ddot{\boldsymbol{q}}_m + \boldsymbol{F}_m\dot{\boldsymbol{q}}_m + \boldsymbol{d} = \boldsymbol{\tau}_m \tag{8.20}$$

where

$$\boldsymbol{F}_m = \boldsymbol{K}_r^{-1}\boldsymbol{F}_v\boldsymbol{K}_r^{-1} \tag{8.21}$$

represents the matrix of viscous friction coefficients about the motor axes, and

$$\boldsymbol{d} = \boldsymbol{K}_r^{-1}\Delta\boldsymbol{B}(\boldsymbol{q})\boldsymbol{K}_r^{-1}\ddot{\boldsymbol{q}}_m + \boldsymbol{K}_r^{-1}\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{K}_r^{-1}\dot{\boldsymbol{q}}_m + \boldsymbol{K}_r^{-1}\boldsymbol{g}(\boldsymbol{q}) \tag{8.22}$$

represents the contribution depending on the configuration.

As illustrated by the block scheme of Fig. 8.5, the system of manipulator with drives is actually constituted by two subsystems; one has $\boldsymbol{\tau}_m$ as input and $\boldsymbol{q}_m$ as output, the other has $\boldsymbol{q}_m$, $\dot{\boldsymbol{q}}_m$, $\ddot{\boldsymbol{q}}_m$ as inputs, and $\boldsymbol{d}$ as output. The former is *linear* and *decoupled*, since each component of $\boldsymbol{\tau}_m$ influences only the corresponding component of $\boldsymbol{q}_m$. The latter is *nonlinear* and *coupled*, since

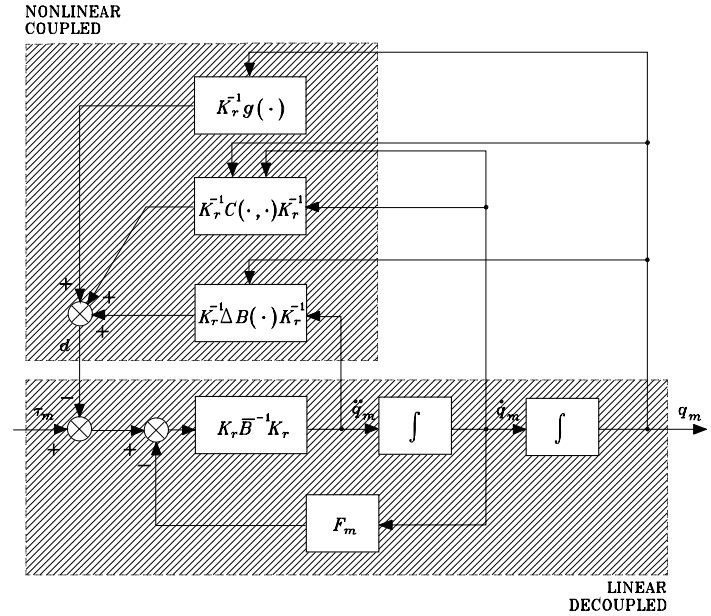

Fig. 8.5. Block scheme of the system of manipulator with drives

it accounts for all those nonlinear and coupling terms of manipulator joint dynamics.

On the basis of the above scheme, several control algorithms can be derived with reference to the detail of knowledge of the dynamic model. The simplest approach that can be followed, in case of high-gear reduction ratios and/or limited performance in terms of required velocities and accelerations, is to consider the component of the nonlinear interacting term $\boldsymbol{d}$ as a *disturbance* for the single joint servo.

The design of the control algorithm leads to a *decentralized control structure*, since each joint is considered independently of the others. The joint controller must guarantee good performance in terms of high disturbance rejection and enhanced trajectory tracking capabilities. The resulting control structure is substantially based on the error between the desired and actual output, while the input control torque at actuator i depends only on the error of output i.

Therefore, the system to control is Joint i drive corresponding to the single-input/single-output system of the decoupled and linear part of the scheme in Fig. 8.5. The interaction with the other joints is described by component i of the vector $\boldsymbol{d}$ in (8.22).

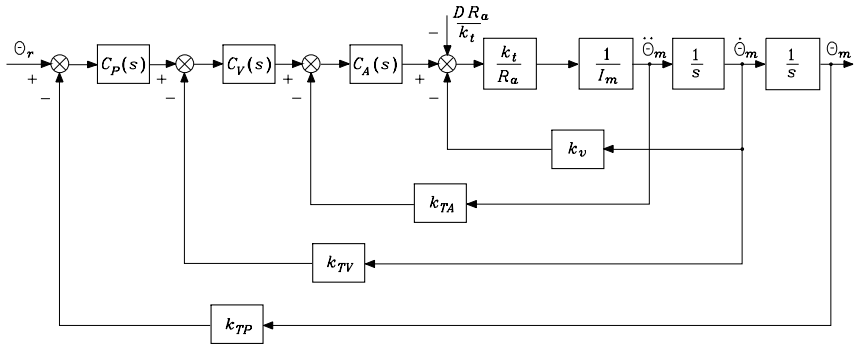

Fig. 8.6. Block scheme of general independent joint control

Assumed that the actuator is a rotary electric DC motor, the general scheme of drive control is that in Fig. 5.9 where I_m is the average inertia reported to the motor axis ($I_{mi} = \bar{b}_{ii}/k_{ri}^2$).[3]

8.3.1 Independent Joint Control

To guide selection of the controller structure, start noticing that an effective rejection of the disturbance d on the output ϑ_m is ensured by:

- a large value of the amplifier gain before the point of intervention of the disturbance,
- the presence of an integral action in the controller so as to cancel the effect of the gravitational component on the output at steady state (constant ϑ_m).

These requisites clearly suggest the use of a *proportional-integral* (PI) control action in the forward path whose transfer function is

$$C(s) = K_c \frac{1 + sT_c}{s}; \tag{8.23}$$

this yields zero error at steady state for a constant disturbance, and the presence of the real zero at $s = -1/T_c$ offers a stabilizing action. To improve dynamic performance, it is worth choosing the controller as a cascade of elementary actions with local feedback loops closed around the disturbance.

Besides closure of a position feedback loop, the most general solution is obtained by closing inner loops on velocity and acceleration. This leads to the scheme in Fig. 8.6, where $C_P(s)$, $C_V(s)$, $C_A(s)$ respectively represent *position*, *velocity*, *acceleration* controllers, and the inmost controller should

[3] Subscript i is to be dropped for notation compactness.

be of PI type as in (8.23) so as to obtain zero error at steady state for a constant disturbance. Further, k_{TP}, k_{TV}, k_{TA} are the respective transducer constants, and the amplifier gain G_v has been embedded in the gain of the inmost controller. In the scheme of Fig. 8.6, notice that ϑ_r is the reference input, which is related to the desired output ϑ_{md} as

$$\vartheta_r = k_{TP}\vartheta_{md}.$$

Further, the disturbance torque D has been suitably transformed into a voltage by the factor R_a/k_t.

In the following, a number of possible solutions that can be derived from the general scheme of Fig. 8.6 are presented; at this stage, the issue arising from possible lack of measurement of physical variables is not considered yet. Three case studies are considered which differ in the number of active feedback loops.[4]

Position feedback

In this case, the control action is characterized by

$$C_P(s) = K_P \frac{1 + sT_P}{s} \qquad C_V(s) = 1 \qquad C_A(s) = 1$$

$$k_{TV} = k_{TA} = 0.$$

With these positions, the structure of the control scheme in Fig. 8.6 leads to the scheme illustrated in Fig. 5.10. From this scheme the transfer function of the forward path is

$$P(s) = \frac{k_m K_P (1 + sT_P)}{s^2(1 + sT_m)},$$

while that of the return path is

$$H(s) = k_{TP}.$$

A root locus analysis can be performed as a function of the gain of the position loop $k_m K_P k_{TP} T_P / T_m$. Three situations are illustrated for the poles of the closed-loop system with reference to the relation between T_P and T_m (Fig. 8.7). Stability of the closed-loop feedback system imposes some constraints on the choice of the parameters of the PI controller. If $T_P < T_m$, the system is inherently unstable (Fig. 8.7a). Then, it must be $T_P > T_m$ (Fig. 8.7b). As T_P increases, the absolute value of the real part of the two roots of the locus tending towards the asymptotes increases too, and the system has faster time response. Hence, it is convenient to render $T_P \gg T_m$ (Fig. 8.7c). In any case, the real part of the dominant poles cannot be less than $-1/2T_m$.

[4] See Appendix C for a brief brush-up on control of linear single-input/single-output systems.

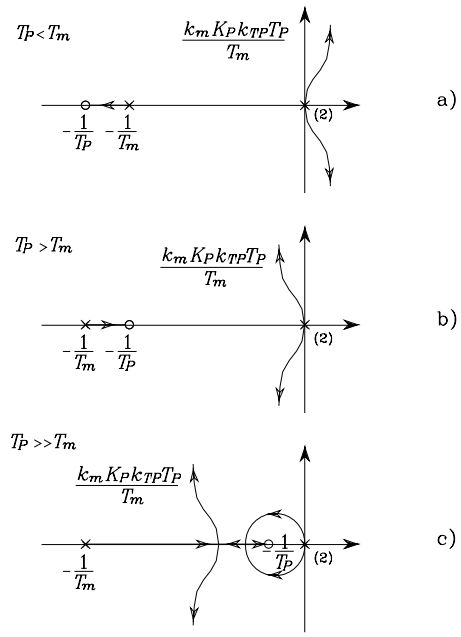

Fig. 8.7. Root loci for the position feedback control scheme

The closed-loop input/output transfer function is

$$\frac{\Theta_m(s)}{\Theta_r(s)} = \frac{\dfrac{1}{k_{TP}}}{1 + \dfrac{s^2(1+sT_m)}{k_m K_P k_{TP}(1+sT_P)}}, \tag{8.24}$$

which can be expressed in the form

$$W(s) = \frac{\dfrac{1}{k_{TP}}(1+sT_P)}{\left(1 + \dfrac{2\zeta s}{\omega_n} + \dfrac{s^2}{\omega_n^2}\right)(1+s\tau)},$$

where ω_n and ζ are respectively the natural frequency and damping ratio of the pair of complex poles and $-1/\tau$ locates the real pole. These values are assigned to define the joint drive dynamics as a function of the constant T_P; if $T_P > T_m$, then $1/\zeta\omega_n > T_P > \tau$ (Fig. 8.7b); if $T_P \gg T_m$ (Fig. 8.7c), for large values of the loop gain, then $\zeta\omega_n > 1/\tau \approx 1/T_P$ and the zero at $-1/T_P$ in the transfer function $W(s)$ tends to cancel the effect of the real pole.

The closed-loop disturbance/output transfer function is

$$\frac{\Theta_m(s)}{D(s)} = -\frac{\dfrac{sR_a}{k_t K_P k_{TP}(1+sT_P)}}{1 + \dfrac{s^2(1+sT_m)}{k_m K_P k_{TP}(1+sT_P)}}, \tag{8.25}$$

which shows that it is worth increasing K_P to reduce the effect of disturbance on the output during the transient. The function in (8.25) has two complex poles $(-\zeta\omega_n, \pm j\sqrt{1-\zeta^2}\omega_n)$, a real pole $(-1/\tau)$, and a zero at the origin. The zero is due to the PI controller and allows the cancellation of the effects of gravity on the angular position when ϑ_m is a constant.

In (8.25), it can be recognized that the term $K_P k_{TP}$ is the reduction factor imposed by the feedback gain on the amplitude of the output due to disturbance; hence, the quantity

$$X_R = K_P k_{TP} \tag{8.26}$$

can be interpreted as the *disturbance rejection factor*, which in turn is determined by the gain K_P. However, it is not advisable to increase K_P too much, because small damping ratios would result leading to unacceptable oscillations of the output. An estimate T_R of the *output recovery time* needed by the control system to recover the effects of the disturbance on the angular position can be evaluated by analyzing the modes of evolution of (8.25). Since $\tau \approx T_P$, such estimate is expressed by

$$T_R = \max\left\{T_P, \frac{1}{\zeta\omega_n}\right\}. \tag{8.27}$$

Position and velocity feedback

In this case, the control action is characterized by

$$C_P(s) = K_P \qquad C_V(s) = K_V\frac{1+sT_V}{s} \qquad C_A(s) = 1$$

$$k_{TA} = 0;$$

with these positions, the structure of the control scheme in Fig. 8.6 leads to scheme illustrated in Fig. 5.11. To carry out a root locus analysis as a function of the velocity feedback loop gain, it is worth reducing the velocity loop in parallel to the position loop by following the usual rules for moving blocks. From the scheme in Fig. 5.11 the transfer function of the forward path is

$$P(s) = \frac{k_m K_P K_V(1+sT_V)}{s^2(1+sT_m)},$$

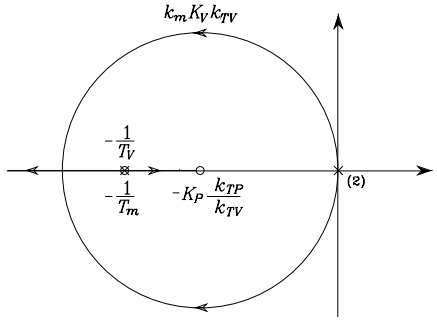

Fig. 8.8. Root locus for the position and velocity feedback control scheme

while that of the return path is

$$H(s) = k_{TP}\left(1 + s\frac{k_{TV}}{K_P k_{TP}}\right).$$

The zero of the controller at $s = -1/T_V$ can be chosen so as to cancel the effects of the real pole of the motor at $s = -1/T_m$. Then, by setting

$$T_V = T_m,$$

the poles of the closed-loop system move on the root locus as a function of the loop gain $k_m K_V k_{TV}$, as shown in Fig. 8.8. By increasing the position feedback gain K_P, it is possible to confine the closed-loop poles into a region of the complex plane with large absolute values of the real part. Then, the actual location can be established by a suitable choice of K_V.

The closed-loop input/output transfer function is

$$\frac{\Theta_m(s)}{\Theta_r(s)} = \frac{\dfrac{1}{k_{TP}}}{1 + \dfrac{sk_{TV}}{K_P k_{TP}} + \dfrac{s^2}{k_m K_P k_{TP} K_V}}, \tag{8.28}$$

which can be compared with the typical transfer function of a second-order system

$$W(s) = \frac{\dfrac{1}{k_{TP}}}{1 + \dfrac{2\zeta s}{\omega_n} + \dfrac{s^2}{\omega_n^2}}. \tag{8.29}$$

It can be recognized that, with a suitable choice of the gains, it is possible to obtain any value of natural frequency ω_n and damping ratio ζ. Hence, if ω_n and ζ are given as design requirements, the following relations can be found:

$$K_V k_{TV} = \frac{2\zeta\omega_n}{k_m} \tag{8.30}$$

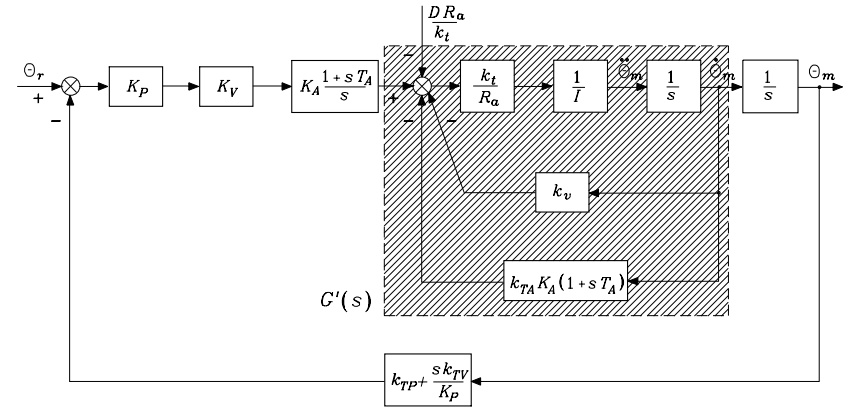

Fig. 8.9. Block scheme of position, velocity and acceleration feedback control

$$K_P k_{TP} K_V = \frac{\omega_n^2}{k_m}. \tag{8.31}$$

For given transducer constants k_{TP} and k_{TV}, once K_V has been chosen to satisfy (8.30), the value of K_P is obtained from (8.31).

The closed-loop disturbance/output transfer function is

$$\frac{\Theta_m(s)}{D(s)} = -\frac{\dfrac{sR_a}{k_t K_P k_{TP} K_V (1 + sT_m)}}{1 + \dfrac{sk_{TV}}{K_P k_{TP}} + \dfrac{s^2}{k_m K_P k_{TP} K_V}}, \tag{8.32}$$

which shows that the *disturbance rejection factor* is

$$X_R = K_P k_{TP} K_V \tag{8.33}$$

and is fixed, once K_P and K_V have been chosen via (8.30), (8.31). Concerning disturbance dynamics, the presence of a zero at the origin introduced by the PI, of a real pole at $s = -1/T_m$, and of a pair of complex poles having real part $-\zeta\omega_n$ should be noticed. Hence, in this case, an estimate of the *output recovery time* is given by the time constant

$$T_R = \max\left\{T_m, \frac{1}{\zeta\omega_n}\right\}; \tag{8.34}$$

which reveals an improvement with respect to the previous case in (8.27), since $T_m \ll T_P$ and the real part of the dominant poles is not constrained by the inequality $\zeta\omega_n < 1/2T_m$.

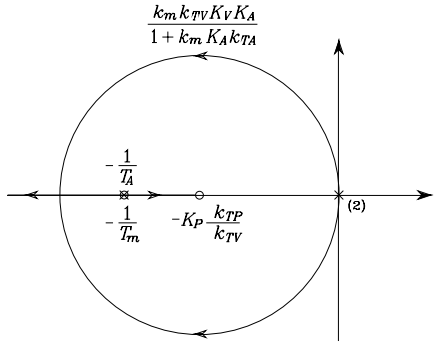

Fig. 8.10. Root locus for the position, velocity and acceleration feedback control scheme

Position, velocity and acceleration feedback

In this case, the control action is characterized by

$$C_P(s) = K_P \qquad C_V(s) = K_V \qquad C_A(s) = K_A \frac{1 + sT_A}{s}.$$

After some manipulation, the block scheme of Fig. 8.6 can be reduced to that of Fig. 8.9 where $G'(s)$ indicates the following transfer function:

$$G'(s) = \frac{k_m}{(1 + k_m K_A k_{TA}) \left(1 + \dfrac{sT_m \left(1 + k_m K_A k_{TA} \dfrac{T_A}{T_m} \right)}{(1 + k_m K_A k_{TA})} \right)}.$$

The transfer function of the forward path is

$$P(s) = \frac{K_P K_V K_A (1 + sT_A)}{s^2} G'(s),$$

while that of the return path is

$$H(s) = k_{TP} \left(1 + \frac{s k_{TV}}{K_P k_{TP}} \right).$$

Also in this case, a suitable pole cancellation is worthy which can be achieved either by setting

$$T_A = T_m,$$

or by making

$$k_m K_A k_{TA} T_A \gg T_m \qquad k_m K_A k_{TA} \gg 1.$$

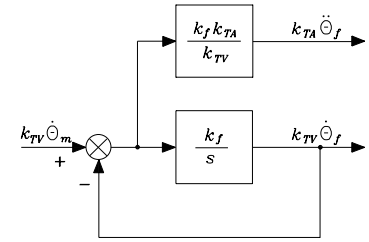

Fig. 8.11. Block scheme of a first-order filter

The two solutions are equivalent as regards dynamic performance of the control system. In both cases, the poles of the closed-loop system are constrained to move on the root locus as a function of the loop gain $k_m K_P K_V K_A/(1 + k_m K_A k_{TA})$ (Fig. 8.10). A close analogy with the previous scheme can be recognized, in that the resulting closed-loop system is again of second-order type.

The closed-loop input/output transfer function is

$$\frac{\Theta_m(s)}{\Theta_r(s)} = \frac{\dfrac{1}{k_{TP}}}{1 + \dfrac{s k_{TV}}{K_P k_{TP}} + \dfrac{s^2 (1 + k_m K_A k_{TA})}{k_m K_P k_{TP} K_V K_A}}, \tag{8.35}$$

while the closed-loop disturbance/output transfer function is

$$\frac{\Theta_m(s)}{D(s)} = -\frac{\dfrac{sR_a}{k_t K_P k_{TP} K_V K_A (1 + sT_A)}}{1 + \dfrac{s k_{TV}}{K_P k_{TP}} + \dfrac{s^2 (1 + k_m K_A k_{TA})}{k_m K_P k_{TP} K_V K_A}}. \tag{8.36}$$

The resulting *disturbance rejection factor* is given by

$$X_R = K_P k_{TP} K_V K_A, \tag{8.37}$$

while the *output recovery time* is given by the time constant

$$T_R = \max \left\{ T_A, \frac{1}{\zeta \omega_n} \right\} \tag{8.38}$$

where T_A can be made less than T_m, as pointed out above.

With reference to the transfer function in (8.29), the following relations can be established for design purposes, once ζ, ω_n, X_R have been specified:

$$\frac{2 K_P k_{TP}}{k_{TV}} = \frac{\omega_n}{\zeta} \tag{8.39}$$

$$k_m K_A k_{TA} = \frac{k_m X_R}{\omega_n^2} - 1 \tag{8.40}$$

$$K_P k_{TP} K_V K_A = X_R. \tag{8.41}$$

For given k_{TP}, k_{TV}, k_{TA}, K_P is chosen to satisfy (8.39), K_A is chosen to satisfy (8.40), and then K_V is obtained from (8.41). Notice how admissible solutions for the controller typically require large values for the rejection factor X_R. Hence, in principle, not only does the acceleration feedback allow the achievement of any desired dynamic behaviour but, with respect to the previous case, it also allows the prescription of the disturbance rejection factor as long as $k_m X_R/\omega_n^2 > 1$.

In deriving the above control schemes, the issue of measurement of feedback variables was not considered explicitly. With reference to the typical position control servos that are implemented in industrial practice, there is no problem of measuring position and velocity, while a direct measurement of acceleration, in general, either is not available or is too expensive to obtain. Therefore, for the scheme of Fig. 8.9, an indirect measurement can be obtained by reconstructing acceleration from direct velocity measurement through a first-order *filter* (Fig. 8.11). The filter is characterized by a bandwidth $\omega_{3f} = k_f$. By choosing this bandwidth wide enough, the effects due to measurement lags are not appreciable, and then it is feasible to take the acceleration filter output as the quantity to feed back. Some problem may occur concerning the noise superimposed on the filtered acceleration signal, though.

Resorting to a filtering technique may be useful when only the direct position measurement is available. In this case, by means of a second-order state variable filter, it is possible to reconstruct velocity and acceleration. However, the greater lags induced by the use of a second-order filter typically degrade the performance with respect to the use of a first-order filter, because of limitations imposed on the filter bandwidth by numerical implementation of the controller and filter.

Notice that the above derivation is based on an ideal dynamic model, i.e., when the effects of transmission elasticity as well as those of amplifier and motor electrical time constants are neglected. This implies that satisfaction of design requirements imposing large values of feedback gains may not be verified in practice, since the existence of unmodelled dynamics — such as electric dynamics, elastic dynamics due to non-perfectly rigid transmissions, filter dynamics for the third scheme — might lead to degrading the system and eventually driving it to instability. In summary, the above solutions constitute design guidelines whose limits should be emphasized with regard to the specific application.

8.3.2 Decentralized Feedforward Compensation

When the joint control servos are required to track reference trajectories with high values of speed and acceleration, the tracking capabilities of the scheme in Fig. 8.6 are unavoidably degraded. The adoption of a *decentralized feedforward compensation* allows a reduction of the tracking error. Therefore, in view of the closed-loop input/output transfer functions in (8.24), (8.28), (8.35),

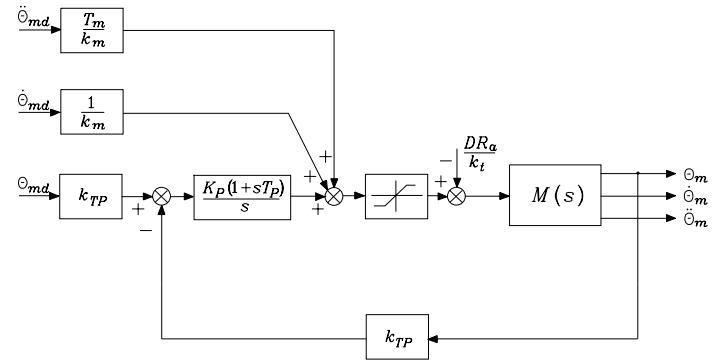

Fig. 8.12. Block scheme of position feedback control with decentralized feedforward compensation

the reference inputs to the three control structures analyzed in the previous section can be respectively modified into

$$\Theta'_r(s) = \left(k_{TP} + \frac{s^2(1+sT_m)}{k_m K_P(1+sT_P)}\right)\Theta_{md}(s) \tag{8.42}$$

$$\Theta'_r(s) = \left(k_{TP} + \frac{sk_{TV}}{K_P} + \frac{s^2}{k_m K_P K_V}\right)\Theta_{md}(s) \tag{8.43}$$

$$\Theta'_r(s) = \left(k_{TP} + \frac{sk_{TV}}{K_P} + \frac{s^2(1+k_m K_A k_{TA})}{k_m K_P K_V K_A}\right)\Theta_{md}(s); \tag{8.44}$$

in this way, tracking of the desired joint position $\Theta_{md}(s)$ is achieved, if not for the effect of disturbances. Notice that computing time derivatives of the desired trajectory is not a problem, once $\vartheta_{md}(t)$ is known analytically. The tracking control schemes, resulting from simple manipulation of (8.42), (8.43), (8.44) are reported respectively in Figs. 8.12, 8.13, 8.14, where $M(s)$ indicates the motor transfer function in (5.11), with k_m and T_m as in (5.12).

All the solutions allow the input trajectory to be tracked within the range of validity and linearity of the models employed. It is worth noticing that, as the number of nested feedback loops increases, a less accurate knowledge of the system model is required to perform feedforward compensation. In fact, T_m and k_m are required for the scheme of Fig. 8.12, only k_m is required for the scheme of Fig. 8.13, and k_m again — but with reduced weight — for the scheme of Fig. 8.14.

It is worth recalling that *perfect* tracking can be obtained only under the assumption of exact matching of the controller and feedforward compensation parameters with the process parameters, as well as of exact modelling and linearity of the physical system. Deviations from the ideal values cause a performance degradation that should be analyzed case by case.

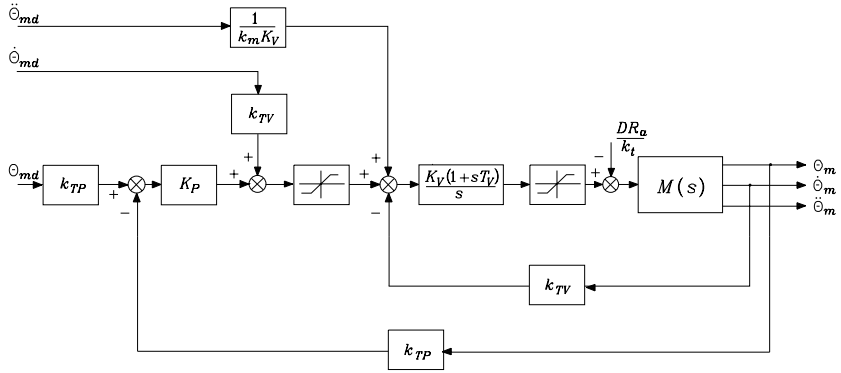

Fig. 8.13. Block scheme of position and velocity feedback control with decentralized feedforward compensation

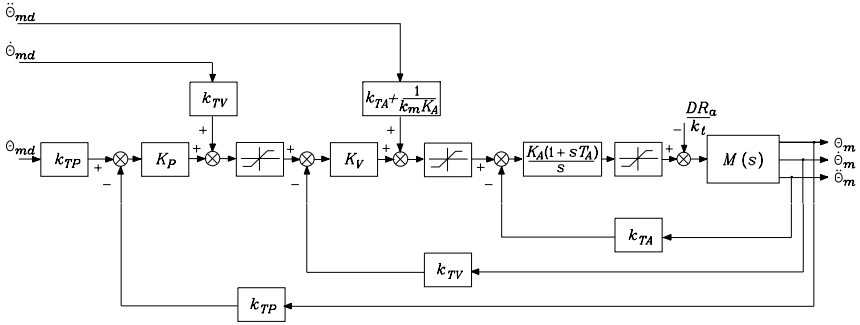

Fig. 8.14. Block scheme of position, velocity and acceleration feedback control with decentralized feedforward compensation

The presence of saturation blocks in the schemes of Figs. 8.12, 8.13, 8.14 is to be intended as intentional nonlinearities whose function is to limit relevant physical quantities during transients; the greater the number of feedback loops, the greater the number of quantities that can be limited (velocity, acceleration, and motor voltage). To this end, notice that trajectory tracking is obviously lost whenever any of the above quantities saturates. This situation often occurs for industrial manipulators required to execute point-to-point motions; in this case, there is less concern about the actual trajectories followed, and the actuators are intentionally taken to operate at the current limits so as to realize the fastest possible motions.

After simple block reduction on the above schemes, it is possible to determine equivalent control structures that utilize position feedback only and *regulators with standard actions*. It should be emphasized that the two solutions are equivalent in terms of disturbance rejection and trajectory tracking.

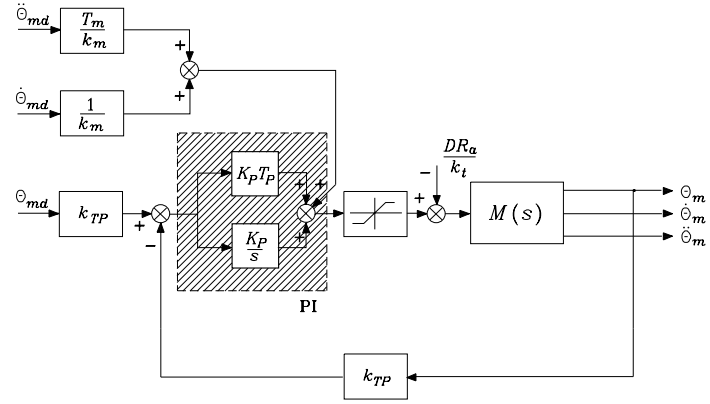

Fig. 8.15. Equivalent control scheme of PI type

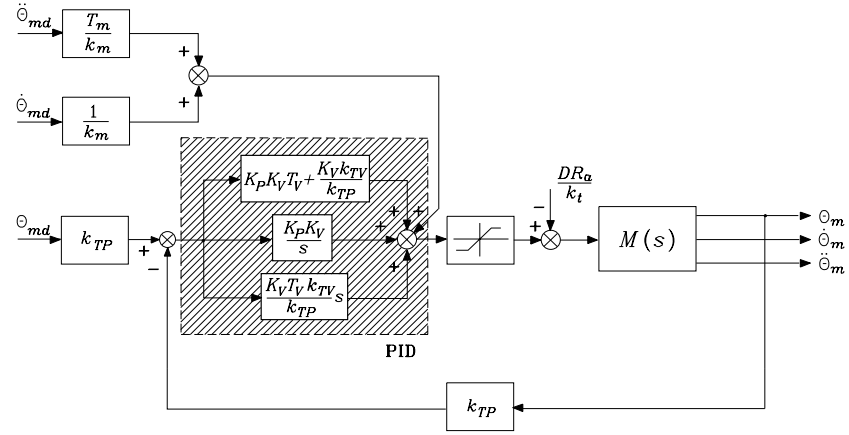

Fig. 8.16. Equivalent control scheme of PID type

However, tuning of regulator parameters is less straightforward, and the elimination of inner feedback loops prevents the possibility of setting saturations on velocity and/or acceleration. The control structures equivalent to those of Figs. 8.12, 8.13, 8.14 are illustrated in Figs. 8.15, 8.16, 8.17, respectively; control actions of PI, PID, PIDD^2 type are illustrated which are respectively equivalent to the cases of: position feedback; position and velocity feedback; position, velocity and acceleration feedback.

It is worth noticing that the equivalent control structures in Figs. 8.15–8.17 are characterized by the presence of the feedforward action $(T_m/k_m)\ddot{\vartheta}_{md} + (1/k_m)\dot{\vartheta}_{md}$. If the motor is current-controlled and not voltage-controlled, by recalling (5.13), the feedforward action is equal to $(k_i/k_t)(I_m\ddot{\vartheta}_{md} + F_m\dot{\vartheta}_{md})$. If $\dot{\vartheta}_m \approx \dot{\vartheta}_{md}$, $\ddot{\vartheta}_m \approx \ddot{\vartheta}_{md}$ and the disturbance is negligible, the term $I_m\ddot{\vartheta}_d +$

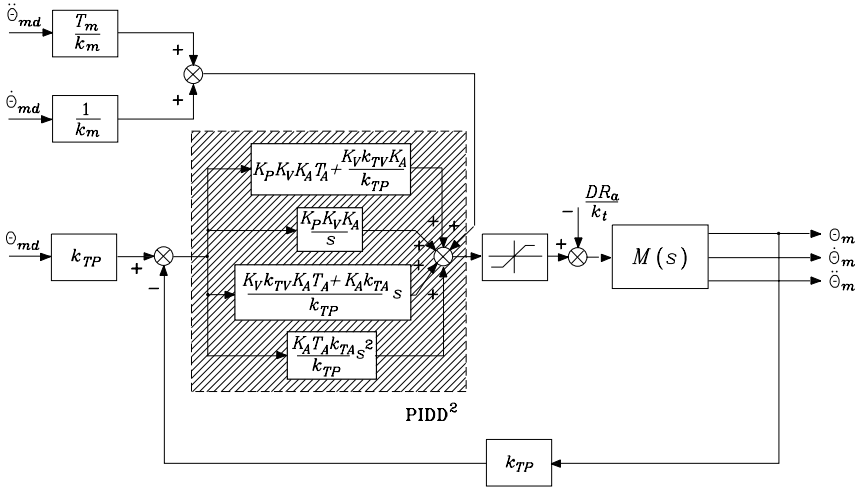

Fig. 8.17. Equivalent control scheme of PIDD^2 type

$F_m\dot{\vartheta}_d$ represents the driving torque providing the desired velocity and acceleration, as indicated by (5.3). By setting

$$i_{ad} = \frac{1}{k_t}(I_m\ddot{\vartheta}_{md} + F_m\dot{\vartheta}_{md}),$$

the feedforward action can be rewritten in the form $k_i i_{ad}$. This shows that, in the case the drive is current-controlled, it is possible to replace the acceleration and velocity feedforward actions with a current and thus a torque feedforward action, which is to be properly computed with reference to the desired motion.

This equivalence is illustrated in Fig. 8.18, where $M(s)$ has been replaced by the block scheme of an electric drive of Fig. 5.2, where the parameters of the current loop are chosen so as to realize a torque-controlled generator. The feedforward action represents a reference for the motor current, which imposes the generation of the nominal torque to execute the desired motion; the presence of the position reference allows the closure of a feedback loop which, in view of the adoption of a standard regulator with transfer function $C_R(s)$, confers robustness to the presented control structure. In summary, the performance that can be achieved with velocity and acceleration feedforward actions and voltage-controlled actuator can be achieved with a current-controlled actuator and a desired torque feedforward action.

The above schemes can incorporate the typical structure of the controllers actually implemented in the control architectures of industrial robots. In these systems it is important to choose the largest possible gains so that model inaccuracy and coupling terms do not appreciably affect positions of the single joints. As pointed out above, the upper limit on the gains is imposed by

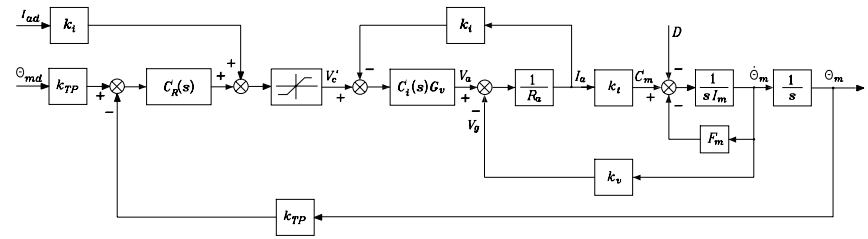

Fig. 8.18. Control scheme with current-controlled drive and current feedforward action

all those factors that have not been modelled, such as implementation of discrete-time controllers in lieu of the continuous-time controllers analyzed in theory, presence of finite sampling time, neglected dynamic effects (e.g., joint elasticity, structural resonance, finite transducer bandwidth), and sensor noise. In fact, the influence of such factors in the implementation of the above controllers may cause a severe system performance degradation for much too large values of feedback gains.

8.4 Computed Torque Feedforward Control

Define the tracking error $e(t) = \vartheta_{md}(t) - \vartheta_m(t)$. With reference to the most general scheme (Fig. 8.17), the output of the PIDD^2 regulator can be written as

$$a_2\ddot{e} + a_1\dot{e} + a_0 e + a_{-1}\int^t e(\varsigma)d\varsigma$$

which describes the time evolution of the error. The constant coefficients a_2, a_1, a_0, a_{-1} are determined by the particular solution adopted. Summing the contribution of the feedforward actions and of the disturbance to this expression yields

$$\frac{T_m}{k_m}\ddot{\vartheta}_{md} + \frac{1}{k_m}\dot{\vartheta}_{md} - \frac{R_a}{k_t}d,$$

where

$$\frac{T_m}{k_m} = \frac{I_m R_a}{k_t} \qquad k_m = \frac{1}{k_v}.$$

The input to the motor (Fig. 8.6) has then to satisfy the following equation:

$$a_2\ddot{e} + a_1\dot{e} + a_0 e + a_{-1}\int^t e(\varsigma)d\varsigma + \frac{T_m}{k_m}\ddot{\vartheta}_{md} + \frac{1}{k_m}\dot{\vartheta}_{md} - \frac{R_a}{k_t}d = \frac{T_m}{k_m}\ddot{\vartheta}_m + \frac{1}{k_m}\dot{\vartheta}_m.$$

With a suitable change of coefficients, this can be rewritten as

$$a_2'\ddot{e} + a_1'\dot{e} + a_0' e + a_{-1}'\int^t e(\varsigma)d\varsigma = \frac{R_a}{k_t}d.$$

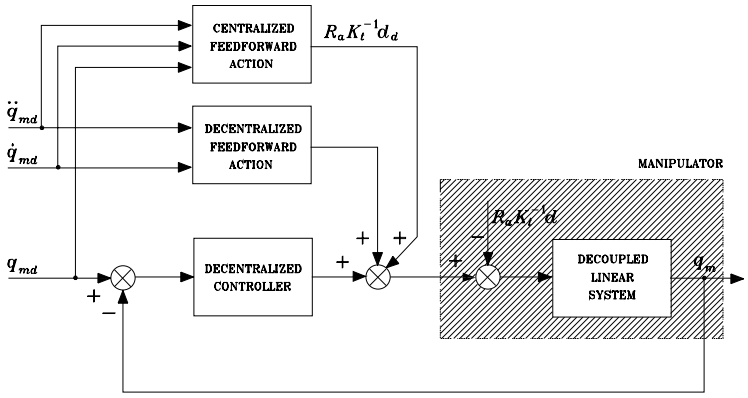

Fig. 8.19. Block scheme of computed torque feedforward control

This equation describes the *error dynamics* and shows that any physically executable trajectory is asymptotically tracked only if the disturbance term $d(t) = 0$. With the term *physically executable* it is meant that the saturation limits on the physical quantities, e.g., current and voltage in electric motors, are not violated in the execution of the desired trajectory.

The presence of the term $d(t)$ causes a tracking error whose magnitude is reduced as much as the disturbance frequency content is located off to the left of the lower limit of the bandwidth of the error system. The disturbance/error transfer function is given by

$$\frac{E(s)}{D(s)} = \frac{\dfrac{R_a}{k_t} s}{a'_2 s^3 + a'_1 s^2 + a'_0 s + a'_{-1}},$$

and thus the adoption of loop gains which are not realizable for the above discussed reasons is often required.

Nevertheless, even if the term $d(t)$ has been introduced as a disturbance, its expression is given by (8.22). It is then possible to add a further term to the previous feedforward actions which is able to compensate the disturbance itself rather than its effects. In other words, by taking advantage of model knowledge, the rejection effort of an independent joint control scheme can be lightened with notable simplification from the implementation viewpoint.

Let $\boldsymbol{q}_d(t)$ be the desired joint trajectory and $\boldsymbol{q}_{md}(t)$ the corresponding actuator trajectory as in (8.2). By adopting an *inverse model* strategy, the *feedforward* action $\boldsymbol{R}_a \boldsymbol{K}_t^{-1} \boldsymbol{d}_d$ can be introduced with

$$\boldsymbol{d}_d = \boldsymbol{K}_r^{-1} \boldsymbol{\Delta B}(\boldsymbol{q}_d) \boldsymbol{K}_r^{-1} \ddot{\boldsymbol{q}}_{md} + \boldsymbol{K}_r^{-1} \boldsymbol{C}(\boldsymbol{q}_d, \dot{\boldsymbol{q}}_d) \boldsymbol{K}_r^{-1} \dot{\boldsymbol{q}}_{md} + \boldsymbol{K}_r^{-1} \boldsymbol{g}(\boldsymbol{q}_d), \quad (8.45)$$

where $\boldsymbol{R}_a$ and $\boldsymbol{K}_t$ denote the diagonal matrices of armature resistances and torque constants of the actuators. This action tends to compensate the actual

disturbance expressed by (8.22) and in turn allows the control system to operate in a better condition.

This solution is illustrated in the scheme of Fig. 8.19, which conceptually describes the control system of a manipulator with *computed torque* control. The feedback control system is representative of the n independent joint control servos; it is *decentralized*, since controller i elaborates references and measurements that refer to single Joint i. The interactions between the various joints, expressed by $\boldsymbol{d}$, are compensated by a *centralized* action whose function is to generate a feedforward action that depends on the joint references as well as on the manipulator dynamic model. This action compensates the nonlinear coupling terms due to inertial, Coriolis, centrifugal, and gravitational forces that depend on the structure and, as such, vary during manipulator motion.

Although the residual disturbance term $\widetilde{\boldsymbol{d}} = \boldsymbol{d}_d - \boldsymbol{d}$ vanishes only in the ideal case of perfect tracking ($\boldsymbol{q} = \boldsymbol{q}_d$) and exact dynamic modelling, $\widetilde{\boldsymbol{d}}$ is representative of interaction disturbances of considerably reduced magnitude with respect to $\boldsymbol{d}$. Hence, the computed torque technique has the advantage to alleviate the disturbance rejection task for the feedback control structure and in turn allows limited gains. Notice that expression (8.45) in general imposes a computationally demanding burden on the centralized part of the controller. Therefore, in those applications where the desired trajectory is generated in real time with regard to exteroceptive sensory data and commands from higher hierarchical levels of the robot control architecture,[5] on-line computation of the centralized feedforward action may require too much time.[6]

Since the actual controller is to be implemented on a computer with a finite sampling time, torque computation has to be carried out during this interval of time; in order not to degrade dynamic system performance, typical sampling times are of the order of the millisecond.

Therefore, it may be worth performing only a *partial* feedforward action so as to compensate those terms of (8.45) that give the most relevant contributions during manipulator motion. Since inertial and gravitational terms dominate velocity-dependent terms (at operational joint speeds not greater than a few radians per second), a partial compensation can be achieved by computing only the gravitational torques and the inertial torques due to the diagonal elements of the inertia matrix. In this way, only the terms depending on the global manipulator configuration are compensated while those deriving from motion interaction with the other joints are not.

Finally, it should be pointed out that, for repetitive trajectories, the above compensating contributions can be computed off-line and properly stored on the basis of a trade-off solution between memory capacity and computational requirements of the control architecture.

[5] See also Chap. 6.

[6] In this regard, the problem of real-time computation of compensating torques can be solved by resorting to efficient recursive formulations of manipulator inverse dynamics, such as the Newton–Euler algorithm presented in Chap. 7.

8.5 Centralized Control

In the previous sections several techniques have been discussed that allow the design of independent joint controllers. These are based on a single-input/single-output approach, since interaction and coupling effects between the joints have been considered as disturbances acting on each single joint drive system.

On the other hand, when large operational speeds are required or direct-drive actuation is employed ($\boldsymbol{K}_r = \boldsymbol{I}$), the nonlinear coupling terms strongly influence system performance. Therefore, considering the effects of the components of $\boldsymbol{d}$ as a disturbance may generate large tracking errors. In this case, it is advisable to design control algorithms that take advantage of a detailed knowledge of manipulator dynamics so as to compensate for the nonlinear coupling terms of the model. In other words, it is necessary to eliminate the causes rather than to reduce the effects induced by them; that is, to generate compensating torques for the nonlinear terms in (8.22). This leads to *centralized control* algorithms that are based on the (partial or complete) knowledge of the manipulator dynamic model.

Whenever the robot is endowed with the torque sensors at the joint motors presented in Sect. 5.4.1, those measurements can be conveniently utilized to generate the compensation action, thus avoiding the on-line computation of the terms of the dynamic model.

As shown by the dynamic model (8.1), the manipulator is not a set of n decoupled system but it is a multivariable system with n inputs (joint torques) and n outputs (joint positions) interacting between them by means of nonlinear relations.[7]

In order to follow a methodological approach which is consistent with control design, it is necessary to treat the control problem in the context of nonlinear multivariable systems. This approach will obviously account for the manipulator *dynamic model* and lead to finding *nonlinear centralized control* laws, whose implementation is needed for high manipulator dynamic performance. On the other hand, the above computed torque control can be interpreted in this framework, since it provides a model-based nonlinear control term to enhance trajectory tracking performance. Notice, however, that this action is inherently performed off line, as it is computed on the time history of the desired trajectory and not of the actual one.

In the following, the problem of the determination of the control law $\boldsymbol{u}$ ensuring a given performance to the system of manipulator with drives is tackles. Since (8.17) can be considered as a proportional relationship between $\boldsymbol{v}_c$ and $\boldsymbol{u}$, the centralized control schemes below refer directly to the generation of control toques $\boldsymbol{u}$.

[7] See Appendix C for the basic concepts on control of nonlinear mechanical systems.

8.5.1 PD Control with Gravity Compensation

Let a *constant* equilibrium posture be assigned for the system as the vector of desired joint variables $\boldsymbol{q}_d$. It is desired to find the structure of the controller which ensures global asymptotic stability of the above posture.

The determination of the control input which stabilizes the system around the equilibrium posture is based on the Lyapunov direct method.

Take the vector $[\,\tilde{\boldsymbol{q}}^T \quad \dot{\boldsymbol{q}}^T\,]^T$ as the system state, where

$$\tilde{\boldsymbol{q}} = \boldsymbol{q}_d - \boldsymbol{q} \tag{8.46}$$

represents the error between the desired and the actual posture. Choose the following positive definite quadratic form as Lyapunov function candidate:

$$V(\dot{\boldsymbol{q}}, \tilde{\boldsymbol{q}}) = \frac{1}{2}\dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}} + \frac{1}{2}\tilde{\boldsymbol{q}}^T \boldsymbol{K}_P \tilde{\boldsymbol{q}} > 0 \qquad \forall \dot{\boldsymbol{q}}, \tilde{\boldsymbol{q}} \neq \boldsymbol{0} \tag{8.47}$$

where $\boldsymbol{K}_P$ is an $(n \times n)$ symmetric positive definite matrix. An energy-based interpretation of (8.47) reveals a first term expressing the system kinetic energy and a second term expressing the potential energy stored in the system of equivalent stiffness $\boldsymbol{K}_P$ provided by the n position feedback loops.

Differentiating (8.47) with respect to time, and recalling that $\boldsymbol{q}_d$ is constant, yields

$$\dot{V} = \dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \frac{1}{2}\dot{\boldsymbol{q}}^T \dot{\boldsymbol{B}}(\boldsymbol{q})\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}^T \boldsymbol{K}_P \tilde{\boldsymbol{q}}. \tag{8.48}$$

Solving (8.7) for $\boldsymbol{B}\ddot{\boldsymbol{q}}$ and substituting it in (8.48) gives

$$\dot{V} = \frac{1}{2}\dot{\boldsymbol{q}}^T \big(\dot{\boldsymbol{B}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\big)\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}^T \boldsymbol{F}\dot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T \big(\boldsymbol{u} - \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{K}_P \tilde{\boldsymbol{q}}\big). \tag{8.49}$$

The first term on the right-hand side is null since the matrix $\boldsymbol{N} = \dot{\boldsymbol{B}} - 2\boldsymbol{C}$ satisfies (7.49). The second term is negative definite. Then, the choice

$$\boldsymbol{u} = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{K}_P \tilde{\boldsymbol{q}}, \tag{8.50}$$

describing a controller with compensation of gravitational terms and a proportional action, leads to a negative semi-definite $\dot{V}$ since

$$\dot{V} = 0 \qquad \dot{\boldsymbol{q}} = \boldsymbol{0}, \forall \tilde{\boldsymbol{q}}.$$

This result can be obtained also by taking the control law

$$\boldsymbol{u} = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{K}_P \tilde{\boldsymbol{q}} - \boldsymbol{K}_D \dot{\boldsymbol{q}}, \tag{8.51}$$

with $\boldsymbol{K}_D$ positive definite, corresponding to a *nonlinear compensation action of gravitational terms* with a *linear proportional-derivative* (PD) *action*. In fact, substituting (8.51) into (8.49) gives

$$\dot{V} = -\dot{\boldsymbol{q}}^T (\boldsymbol{F} + \boldsymbol{K}_D)\dot{\boldsymbol{q}}, \tag{8.52}$$

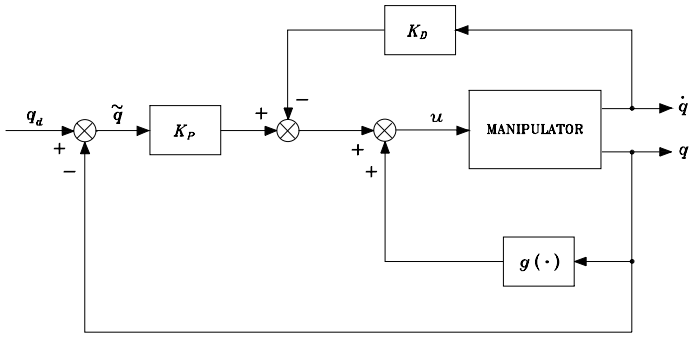

Fig. 8.20. Block scheme of joint space PD control with gravity compensation

which reveals that the introduction of the derivative term causes an increase of the absolute values of $\dot{V}$ along the system trajectories, and then it gives an improvement of system time response. Notice that the inclusion of a derivative action in the controller, as in (8.51), is crucial when direct-drive manipulators are considered. In that case, in fact, mechanical viscous damping is practically null, and current control does not allow the exploitation of the electrical viscous damping provided by voltage-controlled actuators.

According to the above, the function candidate V decreases as long as $\dot{\boldsymbol{q}} \neq \boldsymbol{0}$ for all system trajectories. It can be shown that the system reaches an *equilibrium posture*. To find such posture, notice that $\dot{V} \equiv 0$ only if $\dot{\boldsymbol{q}} \equiv \boldsymbol{0}$. System dynamics under control (8.51) is given by

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{K}_P\tilde{\boldsymbol{q}} - \boldsymbol{K}_D\dot{\boldsymbol{q}}. \tag{8.53}$$

At the equilibrium ($\dot{\boldsymbol{q}} \equiv \boldsymbol{0}$, $\ddot{\boldsymbol{q}} \equiv \boldsymbol{0}$) it is

$$\boldsymbol{K}_P\tilde{\boldsymbol{q}} = \boldsymbol{0} \tag{8.54}$$

and then

$$\tilde{\boldsymbol{q}} = \boldsymbol{q}_d - \boldsymbol{q} \equiv \boldsymbol{0}$$

is the sought equilibrium posture. The above derivation rigorously shows that any manipulator equilibrium posture is *globally asymptotically stable* under a controller with a PD linear action and a nonlinear gravity compensating action. Stability is ensured for any choice of $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$, as long as these are positive definite matrices. The resulting block scheme is shown in Fig. 8.20.

The control law requires the on-line computation of the term $\boldsymbol{g}(\boldsymbol{q})$. If compensation is imperfect, the above discussion does not lead to the same result; this aspect will be revisited later with reference to robustness of controllers performing nonlinear compensation.

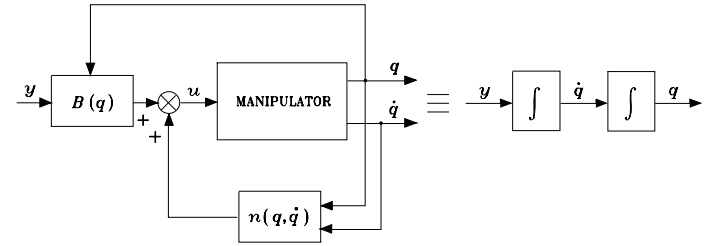

Fig. 8.21. Exact linearization performed by inverse dynamics control

8.5.2 Inverse Dynamics Control

Consider now the problem of tracking a joint space trajectory. The reference framework is that of control of nonlinear multivariable systems. The dynamic model of an n-joint manipulator is expressed by (8.7) which can be rewritten as

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{u}, \tag{8.55}$$

where for simplicity it has been set

$$\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}). \tag{8.56}$$

The approach that follows is founded on the idea to find a control vector $\boldsymbol{u}$, as a function of the system state, which is capable of realizing an input/output relationship of linear type; in other words, it is desired to perform not an approximate linearization but an *exact linearization* of system dynamics obtained by means of a *nonlinear state feedback*. The possibility of finding such a linearizing controller is guaranteed by the particular form of system dynamics. In fact, the equation in (8.55) is linear in the control $\boldsymbol{u}$ and has a full-rank matrix $\boldsymbol{B}(\boldsymbol{q})$ which can be inverted for any manipulator configuration.

Taking the control $\boldsymbol{u}$ as a function of the manipulator state in the form

$$\boldsymbol{u} = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{y} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}), \tag{8.57}$$

leads to the system described by

$$\ddot{\boldsymbol{q}} = \boldsymbol{y}$$

where $\boldsymbol{y}$ represents a new input vector whose expression is to be determined yet; the resulting block scheme is shown in Fig. 8.21. The nonlinear control law in (8.57) is termed *inverse dynamics control* since it is based on the computation of manipulator inverse dynamics. The system under control (8.57) is *linear* and *decoupled* with respect to the new input $\boldsymbol{y}$. In other words, the component y_i influences, with a double integrator relationship, only the joint variable q_i, independently of the motion of the other joints.

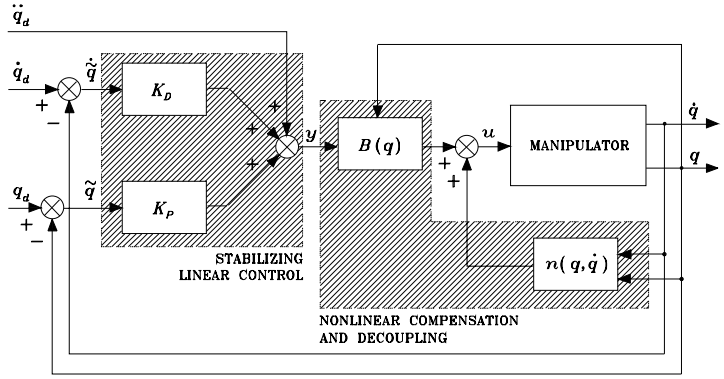

Fig. 8.22. Block scheme of joint space inverse dynamics control

In view of the choice (8.57), the manipulator control problem is reduced to that of finding a stabilizing control law $\boldsymbol{y}$. To this end, the choice

$$\boldsymbol{y} = -\boldsymbol{K}_P\boldsymbol{q} - \boldsymbol{K}_D\dot{\boldsymbol{q}} + \boldsymbol{r} \tag{8.58}$$

leads to the system of second-order equations

$$\ddot{\boldsymbol{q}} + \boldsymbol{K}_D\dot{\boldsymbol{q}} + \boldsymbol{K}_P\boldsymbol{q} = \boldsymbol{r} \tag{8.59}$$

which, under the assumption of positive definite matrices $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$, is asymptotically stable. Choosing $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$ as *diagonal* matrices of the type

$$\boldsymbol{K}_P = \text{diag}\{\omega_{n1}^2, \ldots, \omega_{nn}^2\} \qquad \boldsymbol{K}_D = \text{diag}\{2\zeta_1\omega_{n1}, \ldots, 2\zeta_n\omega_{nn}\},$$

gives a decoupled system. The reference component r_i influences only the joint variable q_i with a second-order input/output relationship characterized by a natural frequency ω_{ni} and a damping ratio ζ_i.

Given any desired trajectory $\boldsymbol{q}_d(t)$, tracking of this trajectory for the output $\boldsymbol{q}(t)$ is ensured by choosing

$$\boldsymbol{r} = \ddot{\boldsymbol{q}}_d + \boldsymbol{K}_D\dot{\boldsymbol{q}}_d + \boldsymbol{K}_P\boldsymbol{q}_d. \tag{8.60}$$

In fact, substituting (8.60) into (8.59) gives the homogeneous second-order differential equation

$$\ddot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_D\dot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_P\tilde{\boldsymbol{q}} = \boldsymbol{0} \tag{8.61}$$

expressing the dynamics of position error (8.46) while tracking the given trajectory. Such error occurs only if $\tilde{\boldsymbol{q}}(0)$ and/or $\dot{\tilde{\boldsymbol{q}}}(0)$ are different from zero and converges to zero with a speed depending on the matrices $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$ chosen.

The resulting block scheme is illustrated in Fig. 8.22, in which two feedback loops are represented; an inner loop based on the manipulator dynamic model, and an outer loop operating on the tracking error. The function of the *inner loop* is to obtain a *linear and decoupled input/output relationship*, whereas the *outer loop* is required to *stabilize the overall system*. The controller design for the outer loop is simplified since it operates on a linear and time-invariant system. Notice that the implementation of this control scheme requires computation of the inertia matrix $\boldsymbol{B}(\boldsymbol{q})$ and of the vector of Coriolis, centrifugal, gravitational, and damping terms $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in (8.56). Unlike computed torque control, these terms must be computed *on-line* since control is now based on nonlinear feedback of the current system state, and thus it is not possible to precompute the terms off line as for the previous technique.

The above technique of nonlinear compensation and decoupling is very attractive from a control viewpoint since the nonlinear and coupled manipulator dynamics is replaced with n linear and decoupled second-order subsystems. Nonetheless, this technique is based on the assumption of perfect cancellation of dynamic terms, and then it is quite natural to raise questions about sensitivity and robustness problems due to unavoidably imperfect compensation.

Implementation of inverse dynamics control laws indeed requires that parameters of the system dynamic model are accurately known and the complete equations of motion are computed in real time. These conditions are difficult to verify in practice. On one hand, the model is usually known with a certain degree of uncertainty due to imperfect knowledge of manipulator mechanical parameters, existence of unmodelled dynamics, and model dependence on end-effector payloads not exactly known and thus not perfectly compensated. On the other hand, inverse dynamics computation is to be performed at sampling times of the order of a millisecond so as to ensure that the assumption of operating in the continuous time domain is realistic. This may pose severe constraints on the hardware/software architecture of the control system. In such cases, it may be advisable to lighten the computation of inverse dynamics and compute only the dominant terms.

On the basis of the above remarks, from an implementation viewpoint, *compensation* may be *imperfect* both for model uncertainty and for the approximations made in on-line computation of inverse dynamics. In the following, two control techniques are presented which are aimed at counteracting the effects of imperfect compensation. The first consists of the introduction of an additional term to an inverse dynamics controller which provides *robustness* to the control system by counteracting the effects of the approximations made in on-line computation of inverse dynamics. The second *adapts* the parameters of the model used for inverse dynamics computation to those of the true manipulator dynamic model.

8.5.3 Robust Control

In the case of *imperfect compensation*, it is reasonable to assume in (8.55) a control vector expressed by

$$\boldsymbol{u} = \widehat{\boldsymbol{B}}(\boldsymbol{q})\boldsymbol{y} + \widehat{\boldsymbol{n}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{8.62}$$

where $\widehat{\boldsymbol{B}}$ and $\widehat{\boldsymbol{n}}$ represent the adopted computational model in terms of estimates of the terms in the dynamic model. The error on the estimates, i.e., the *uncertainty*, is represented by

$$\widetilde{\boldsymbol{B}} = \widehat{\boldsymbol{B}} - \boldsymbol{B} \qquad \widetilde{\boldsymbol{n}} = \widehat{\boldsymbol{n}} - \boldsymbol{n} \tag{8.63}$$

and is due to imperfect model compensation as well as to intentional simplification in inverse dynamics computation. Notice that by setting $\widehat{\boldsymbol{B}} = \bar{\boldsymbol{B}}$ (where $\bar{\boldsymbol{B}}$ is the diagonal matrix of average inertia at the joint axes) and $\widehat{\boldsymbol{n}} = \boldsymbol{0}$, the above decentralized control scheme is recovered where the control action $\boldsymbol{y}$ can be of the general PID type computed on the error.

Using (8.62) as a nonlinear control law gives

$$\boldsymbol{B}\ddot{\boldsymbol{q}} + \boldsymbol{n} = \widehat{\boldsymbol{B}}\boldsymbol{y} + \widehat{\boldsymbol{n}} \tag{8.64}$$

where functional dependence has been omitted. Since the inertia matrix $\boldsymbol{B}$ is invertible, it is

$$\ddot{\boldsymbol{q}} = \boldsymbol{y} + (\boldsymbol{B}^{-1}\widehat{\boldsymbol{B}} - \boldsymbol{I})\boldsymbol{y} + \boldsymbol{B}^{-1}\widetilde{\boldsymbol{n}} = \boldsymbol{y} - \boldsymbol{\eta} \tag{8.65}$$

where

$$\boldsymbol{\eta} = (\boldsymbol{I} - \boldsymbol{B}^{-1}\widehat{\boldsymbol{B}})\boldsymbol{y} - \boldsymbol{B}^{-1}\widetilde{\boldsymbol{n}}. \tag{8.66}$$

Taking as above

$$\boldsymbol{y} = \ddot{\boldsymbol{q}}_d + \boldsymbol{K}_D(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}}) + \boldsymbol{K}_P(\boldsymbol{q}_d - \boldsymbol{q}),$$

leads to

$$\ddot{\widetilde{\boldsymbol{q}}} + \boldsymbol{K}_D\dot{\widetilde{\boldsymbol{q}}} + \boldsymbol{K}_P\widetilde{\boldsymbol{q}} = \boldsymbol{\eta}. \tag{8.67}$$

The system described by (8.67) is still nonlinear and coupled, since $\boldsymbol{\eta}$ is a nonlinear function of $\widetilde{\boldsymbol{q}}$ and $\dot{\widetilde{\boldsymbol{q}}}$; error convergence to zero is not ensured by the term on the left-hand side only.

To find control laws ensuring error convergence to zero while tracking a trajectory even in the face of uncertainties, a linear PD control is no longer sufficient. To this end, the Lyapunov direct method can be utilized again for the design of an outer feedback loop on the error which should be *robust* to the uncertainty $\boldsymbol{\eta}$.

Let the desired trajectory $\boldsymbol{q}_d(t)$ be assigned in the joint space and let $\widetilde{\boldsymbol{q}} = \boldsymbol{q}_d - \boldsymbol{q}$ be the position error. Its first time-derivative is $\dot{\widetilde{\boldsymbol{q}}} = \dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}}$, while its second time-derivative in view of (8.65) is

$$\ddot{\widetilde{\boldsymbol{q}}} = \ddot{\boldsymbol{q}}_d - \boldsymbol{y} + \boldsymbol{\eta}. \tag{8.68}$$

By taking

$$\boldsymbol{\xi} = \begin{bmatrix} \widetilde{\boldsymbol{q}} \\ \dot{\widetilde{\boldsymbol{q}}} \end{bmatrix}, \tag{8.69}$$

as the system state, the following first-order differential matrix equation is obtained:

$$\dot{\boldsymbol{\xi}} = \boldsymbol{H}\boldsymbol{\xi} + \boldsymbol{D}(\ddot{\boldsymbol{q}}_d - \boldsymbol{y} + \boldsymbol{\eta}), \tag{8.70}$$

where $\boldsymbol{H}$ and $\boldsymbol{D}$ are block matrices of dimensions $(2n \times 2n)$ and $(2n \times n)$, respectively:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{O} & \boldsymbol{I} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \qquad \boldsymbol{D} = \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{I} \end{bmatrix}. \tag{8.71}$$

Then, the problem of tracking a given trajectory can be regarded as the problem of finding a control law $\boldsymbol{y}$ which stabilizes the nonlinear time-varying error system (8.70).

Control design is based on the assumption that, even though the uncertainty $\boldsymbol{\eta}$ is unknown, an estimate on its range of variation is available. The sought control law $\boldsymbol{y}$ should guarantee asymptotic stability of (8.70) for any $\boldsymbol{\eta}$ varying in the above range. By recalling that $\boldsymbol{\eta}$ in (8.66) is a function of $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}_d$, the following assumptions are made:

$$\sup_{t\geq 0} \|\ddot{\boldsymbol{q}}_d\| < Q_M < \infty \quad \forall \ddot{\boldsymbol{q}}_d \tag{8.72}$$

$$\|\boldsymbol{I} - \boldsymbol{B}^{-1}(\boldsymbol{q})\widehat{\boldsymbol{B}}(\boldsymbol{q})\| \leq \alpha \leq 1 \; \forall \boldsymbol{q} \tag{8.73}$$

$$\|\widetilde{\boldsymbol{n}}\| \leq \Phi < \infty \qquad \forall \boldsymbol{q}, \dot{\boldsymbol{q}}. \tag{8.74}$$

Assumption (8.72) is practically satisfied since any planned trajectory cannot require infinite accelerations.

Regarding assumption (8.73), since $\boldsymbol{B}$ is a positive definite matrix with upper and lower limited norms, the following inequality holds:

$$0 < B_m \leq \|\boldsymbol{B}^{-1}(\boldsymbol{q})\| \leq B_M < \infty \qquad \forall \boldsymbol{q}, \tag{8.75}$$

and then a choice for $\widehat{\boldsymbol{B}}$ always exists which satisfies (8.73). In fact, by setting

$$\widehat{\boldsymbol{B}} = \frac{2}{B_M + B_m}\boldsymbol{I},$$

from (8.73) it is

$$\|\boldsymbol{B}^{-1}\widehat{\boldsymbol{B}} - \boldsymbol{I}\| \leq \frac{B_M - B_m}{B_M + B_m} = \alpha < 1. \tag{8.76}$$

If $\widehat{\boldsymbol{B}}$ is a more accurate estimate of the inertia matrix, the inequality is satisfied with values of α that can be made arbitrarily small (in the limit, it is $\widehat{\boldsymbol{B}} = \boldsymbol{B}$ and $\alpha = 0$).

Finally, concerning assumption (8.74), observe that $\widetilde{\boldsymbol{n}}$ is a function of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$. For revolute joints a periodical dependence on $\boldsymbol{q}$ is obtained, while for prismatic joints a linear dependence is obtained, but the joint ranges are limited and then the above contribution is also limited. On the other hand, regarding the dependence on $\dot{\boldsymbol{q}}$, unbounded velocities for an unstable system may arise in the limit, but in reality saturations exist on the maximum velocities of the motors. In summary, assumption (8.74) can be realistically satisfied, too.

With reference to (8.65), choose now

$$\boldsymbol{y} = \ddot{\boldsymbol{q}}_d + \boldsymbol{K}_D\dot{\widetilde{\boldsymbol{q}}} + \boldsymbol{K}_P\widetilde{\boldsymbol{q}} + \boldsymbol{w} \tag{8.77}$$

where the PD term ensures stabilization of the error dynamic system matrix, $\ddot{\boldsymbol{q}}_d$ provides a feedforward term, and the term $\boldsymbol{w}$ is to be chosen to guarantee robustness to the effects of uncertainty described by $\boldsymbol{\eta}$ in (8.66).

Using (8.77) and setting $\boldsymbol{K} = [\,\boldsymbol{K}_P \quad \boldsymbol{K}_D\,]$ yields

$$\dot{\boldsymbol{\xi}} = \widetilde{\boldsymbol{H}}\boldsymbol{\xi} + \boldsymbol{D}(\boldsymbol{\eta} - \boldsymbol{w}), \tag{8.78}$$

where

$$\widetilde{\boldsymbol{H}} = (\boldsymbol{H} - \boldsymbol{D}\boldsymbol{K}) = \begin{bmatrix} \boldsymbol{O} & \boldsymbol{I} \\ -\boldsymbol{K}_P & -\boldsymbol{K}_D \end{bmatrix}$$

is a matrix whose eigenvalues all have negative real parts — $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$ being positive definite — which allows the desired error system dynamics to be prescribed. In fact, by choosing $\boldsymbol{K}_P = \mathrm{diag}\{\omega_{n1}^2, \ldots, \omega_{nn}^2\}$ and $\boldsymbol{K}_D = \mathrm{diag}\{2\zeta_1\omega_{n1}, \ldots, 2\zeta_n\omega_{nn}\}$, n decoupled equations are obtained as regards the linear part. If the uncertainty term vanishes, it is obviously $\boldsymbol{w} = \boldsymbol{0}$ and the above result with an exact inverse dynamics controller is recovered ($\widehat{\boldsymbol{B}} = \boldsymbol{B}$ and $\widehat{\boldsymbol{n}} = \boldsymbol{n}$).

To determine $\boldsymbol{w}$, consider the following positive definite quadratic form as Lyapunov function candidate:

$$V(\boldsymbol{\xi}) = \boldsymbol{\xi}^T\boldsymbol{Q}\boldsymbol{\xi} > 0 \qquad \forall \boldsymbol{\xi} \neq \boldsymbol{0}, \tag{8.79}$$

where $\boldsymbol{Q}$ is a $(2n \times 2n)$ positive definite matrix. The derivative of V along the trajectories of the error system (8.78) is

$$\begin{aligned} \dot{V} &= \dot{\boldsymbol{\xi}}^T\boldsymbol{Q}\boldsymbol{\xi} + \boldsymbol{\xi}^T\boldsymbol{Q}\dot{\boldsymbol{\xi}} \\ &= \boldsymbol{\xi}^T(\widetilde{\boldsymbol{H}}^T\boldsymbol{Q} + \boldsymbol{Q}\widetilde{\boldsymbol{H}})\boldsymbol{\xi} + 2\boldsymbol{\xi}^T\boldsymbol{Q}\boldsymbol{D}(\boldsymbol{\eta} - \boldsymbol{w}). \end{aligned} \tag{8.80}$$

Since $\widetilde{\boldsymbol{H}}$ has eigenvalues with all negative real parts, it is well-known that for any symmetric positive definite matrix $\boldsymbol{P}$, the equation

$$\widetilde{\boldsymbol{H}}^T\boldsymbol{Q} + \boldsymbol{Q}\widetilde{\boldsymbol{H}} = -\boldsymbol{P} \tag{8.81}$$

gives a unique solution $\boldsymbol{Q}$ which is symmetric positive definite as well. In view of this, (8.80) becomes

$$\dot{V} = -\boldsymbol{\xi}^T\boldsymbol{P}\boldsymbol{\xi} + 2\boldsymbol{\xi}^T\boldsymbol{Q}\boldsymbol{D}(\boldsymbol{\eta} - \boldsymbol{w}). \tag{8.82}$$

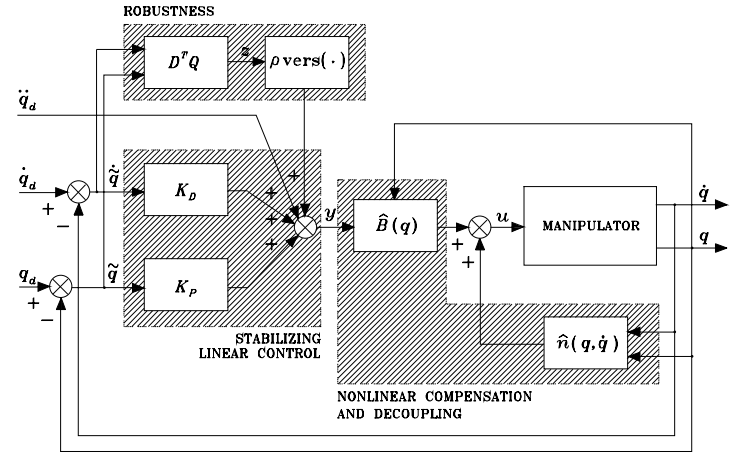

Fig. 8.23. Block scheme of joint space robust control

The first term on the right-hand side of (8.82) is negative definite and then the solutions converge if $\boldsymbol{\xi} \in \mathcal{N}(\boldsymbol{D}^T\boldsymbol{Q})$. If instead $\boldsymbol{\xi} \notin \mathcal{N}(\boldsymbol{D}^T\boldsymbol{Q})$, the control $\boldsymbol{w}$ must be chosen so as to render the second term in (8.82) less than or equal to zero. By setting $\boldsymbol{z} = \boldsymbol{D}^T\boldsymbol{Q}\boldsymbol{\xi}$, the second term in (8.82) can be rewritten as $\boldsymbol{z}^T(\boldsymbol{\eta} - \boldsymbol{w})$. Adopting the control law

$$\boldsymbol{w} = \frac{\rho}{\|\boldsymbol{z}\|}\boldsymbol{z} \qquad \rho > 0 \tag{8.83}$$

gives[8]

$$\begin{aligned} \boldsymbol{z}^T(\boldsymbol{\eta} - \boldsymbol{w}) &= \boldsymbol{z}^T\boldsymbol{\eta} - \frac{\rho}{\|\boldsymbol{z}\|}\boldsymbol{z}^T\boldsymbol{z} \\ &\leq \|\boldsymbol{z}\|\|\boldsymbol{\eta}\| - \rho\|\boldsymbol{z}\| \\ &= \|\boldsymbol{z}\|(\|\boldsymbol{\eta}\| - \rho). \end{aligned} \tag{8.84}$$

Then, if ρ is chosen so that

$$\rho \geq \|\boldsymbol{\eta}\| \qquad \forall \boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}_d, \tag{8.85}$$

the control (8.83) ensures that $\dot{V}$ is less than zero along all error system trajectories.

In order to satisfy (8.85), notice that, in view of the definition of $\boldsymbol{\eta}$ in (8.66) and of assumptions (8.72)–(8.74), and being $\|\boldsymbol{w}\| = \rho$, it is

$$\|\boldsymbol{\eta}\| \leq \|\boldsymbol{I} - \boldsymbol{B}^{-1}\widehat{\boldsymbol{B}}\|\big(\|\ddot{\boldsymbol{q}}_d\| + \|\boldsymbol{K}\|\,\|\boldsymbol{\xi}\| + \|\boldsymbol{w}\|\big) + \|\boldsymbol{B}^{-1}\|\,\|\widetilde{\boldsymbol{n}}\|$$

[8] Notice that it is necessary to divide $\boldsymbol{z}$ by the norm of $\boldsymbol{z}$ so as to obtain a linear dependence on $\boldsymbol{z}$ of the term containing the control $\boldsymbol{z}^T\boldsymbol{w}$, and thus to effectively counteract, for $\boldsymbol{z} \to \boldsymbol{0}$, the term containing the uncertainty $\boldsymbol{z}^T\boldsymbol{\eta}$ which is linear in $\boldsymbol{z}$.

$$\leq \alpha Q_M + \alpha \|\boldsymbol{K}\| \, \|\boldsymbol{\xi}\| + \alpha\rho + B_M\Phi. \tag{8.86}$$

Therefore, setting

$$\rho \geq \frac{1}{1-\alpha}(\alpha Q_M + \alpha\|\boldsymbol{K}\|\|\boldsymbol{\xi}\| + B_M\Phi) \tag{8.87}$$

gives

$$\dot{V} = -\boldsymbol{\xi}^T\boldsymbol{P}\boldsymbol{\xi} + 2\boldsymbol{z}^T\left(\boldsymbol{\eta} - \frac{\rho}{\|\boldsymbol{z}\|}\boldsymbol{z}\right) < 0 \qquad \forall \boldsymbol{\xi} \neq \boldsymbol{0}. \tag{8.88}$$

The resulting block scheme is illustrated in Fig. 8.23.

To summarize, the presented approach has lead to finding a *control* law which is formed by three different contributions:

- The term $\widehat{\boldsymbol{B}}\boldsymbol{y}+\widehat{\boldsymbol{n}}$ ensures an *approximate compensation of nonlinear effects and joint decoupling*.
- The term $\ddot{\boldsymbol{q}}_d + \boldsymbol{K}_D\tilde{\dot{\boldsymbol{q}}} + \boldsymbol{K}_P\tilde{\boldsymbol{q}}$ introduces a *linear feedforward action* ($\ddot{\boldsymbol{q}}_d + \boldsymbol{K}_D\dot{\boldsymbol{q}}_d + \boldsymbol{K}_P\boldsymbol{q}_d$) and *linear feedback action* ($-\boldsymbol{K}_D\dot{\boldsymbol{q}} - \boldsymbol{K}_P\boldsymbol{q}$) *which stabilizes the error system dynamics*.
- The term $\boldsymbol{w} = (\rho/\|\boldsymbol{z}\|)\boldsymbol{z}$ represents the *robust contribution that counteracts the indeterminacy* $\widetilde{\boldsymbol{B}}$ and $\widetilde{\boldsymbol{n}}$ in computing the nonlinear terms that depend on the manipulator state; the greater the uncertainty, the greater the positive scalar ρ. The resulting control law is of the *unit vector* type, since it is described by a vector of magnitude ρ aligned with the unit vector of $\boldsymbol{z} = \boldsymbol{D}^T\boldsymbol{Q}\boldsymbol{\xi}$, $\forall\boldsymbol{\xi}$.

All the resulting trajectories under the above robust control reach the subspace $\boldsymbol{z} = \boldsymbol{D}^T\boldsymbol{Q}\boldsymbol{\xi} = \boldsymbol{0}$ that depends on the matrix $\boldsymbol{Q}$ in the Lyapunov function V. On this *attractive* subspace, termed *sliding subspace*, the control $\boldsymbol{w}$ is ideally commuted at an infinite frequency and all error components tend to zero with a transient depending on the matrices $\boldsymbol{Q}$, $\boldsymbol{K}_P$, $\boldsymbol{K}_D$. A characterization of an error trajectory in the two-dimensional case is given in Fig. 8.24. Notice that in the case $\boldsymbol{\xi}(0) \neq \boldsymbol{0}$, with $\boldsymbol{\xi}(0) \notin \mathcal{N}(\boldsymbol{D}^T\boldsymbol{Q})$, the trajectory is attracted on the sliding hyperplane (a line) $\boldsymbol{z} = \boldsymbol{0}$ and tends towards the origin of the error state space with a time evolution governed by ρ.

In reality, the physical limits on the elements employed in the controller impose a control signal that commutes at a finite frequency, and the trajectories oscillate around the sliding subspace with a magnitude as low as the frequency is high.

Elimination of these high-frequency components (*chattering*) can be achieved by adopting a robust control law which, even if it does not guarantee error convergence to zero, ensures *bounded-norm* errors. A control law of this type is

$$\boldsymbol{w} = \begin{cases} \dfrac{\rho}{\|\boldsymbol{z}\|}\boldsymbol{z} & \text{per } \|\boldsymbol{z}\| \geq \epsilon \\ \dfrac{\rho}{\epsilon}\boldsymbol{z} & \text{per } \|\boldsymbol{z}\| < \epsilon. \end{cases} \tag{8.89}$$

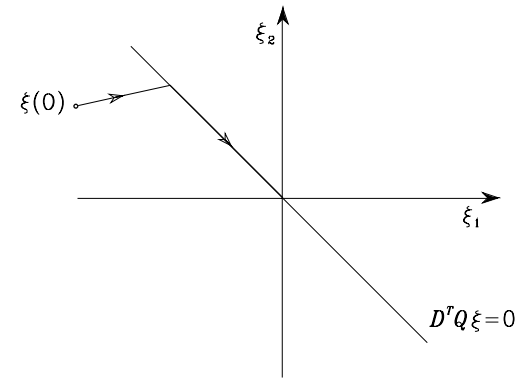

Fig. 8.24. Error trajectory with robust control

In order to provide an intuitive interpretation of this law, notice that (8.89) gives a null control input when the error is in the null space of matrix $\boldsymbol{D}^T\boldsymbol{Q}$. On the other hand, (8.83) has an equivalent gain tending to infinity when $\boldsymbol{z}$ tends to the null vector, thus generating a control input of limited magnitude. Since these inputs commute at an infinite frequency, they force the error system dynamics to stay on the sliding subspace. With reference to the above example, control law (8.89) gives rise to a hyperplane $\boldsymbol{z} = \boldsymbol{0}$ which is no longer attractive, and the error is allowed to vary within a boundary layer whose thickness depends on ϵ (Fig. 8.25).

The introduction of a contribution based on the computation of a suitable linear combination of the generalized error confers robustness to a control scheme based on nonlinear compensation. Even if the manipulator is accurately modeled, indeed, an exact nonlinear compensation may be computationally demanding, and thus it may require either a sophisticated hardware architecture or an increase of the sampling time needed to compute the control law. The solution then becomes weak from an engineering viewpoint, due either to infeasible costs of the control architecture, or to poor performance at decreased sampling rates. Therefore, considering a partial knowledge of the manipulator dynamic model with an accurate, pondered estimate of uncertainty may suggest robust control solutions of the kind presented above. It is understood that an estimate of the uncertainty should be found so as to impose control inputs which the mechanical structure can bear.

8.5.4 Adaptive Control

The computational model employed for computing inverse dynamics typically has the same structure as that of the true manipulator dynamic model, but parameter estimate uncertainty does exist. In this case, it is possible to devise solutions that allow an *on-line adaptation of the computational model to the*

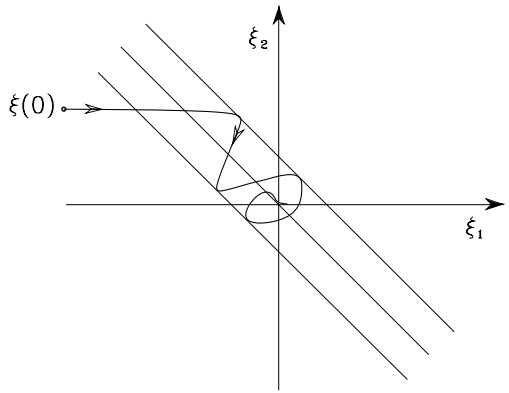

Fig. 8.25. Error trajectory with robust control and chattering elimination

dynamic model, thus performing a control scheme of the inverse dynamics type.

The possibility of finding adaptive control laws is ensured by the property of *linearity in the parameters* of the dynamic model of a manipulator. In fact, it is always possible to express the nonlinear equations of motion in a linear form with respect to a suitable set of constant dynamic parameters as in (7.81). The equation in (8.7) can then be written as

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\boldsymbol{\pi} = \boldsymbol{u}, \tag{8.90}$$

where $\boldsymbol{\pi}$ is a $(p \times 1)$ vector of constant parameters and $\boldsymbol{Y}$ is an $(n \times p)$ matrix which is a function of joint positions, velocities and accelerations. This property of linearity in the dynamic parameters is fundamental for deriving adaptive control laws, among which the technique illustrated below is one of the simplest.

At first, a control scheme which can be derived through a combined computed torque/inverse dynamics approach is illustrated. The computational model is assumed to coincide with the dynamic model.

Consider the control law

$$\boldsymbol{u} = \boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}}_r + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}_r + \boldsymbol{F}\dot{\boldsymbol{q}}_r + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{K}_D\boldsymbol{\sigma}, \tag{8.91}$$

with $\boldsymbol{K}_D$ a positive definite matrix. The choice

$$\dot{\boldsymbol{q}}_r = \dot{\boldsymbol{q}}_d + \boldsymbol{\Lambda}\tilde{\boldsymbol{q}} \qquad \ddot{\boldsymbol{q}}_r = \ddot{\boldsymbol{q}}_d + \boldsymbol{\Lambda}\dot{\tilde{\boldsymbol{q}}}, \tag{8.92}$$

with $\boldsymbol{\Lambda}$ a positive definite (usually diagonal) matrix, allows the nonlinear compensation and decoupling terms to be expressed as a function of the desired velocity and acceleration, corrected by the current state ($\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$) of the manipulator. In fact, notice that the term $\dot{\boldsymbol{q}}_r = \dot{\boldsymbol{q}}_d + \boldsymbol{\Lambda}\tilde{\boldsymbol{q}}$ weighs the contribution

that depends on velocity, not only on the basis of the desired velocity but also on the basis of the position tracking error. A similar argument also holds for the acceleration contribution, where a term depending on the velocity tracking error is considered besides the desired acceleration.

The term $\boldsymbol{K}_D\boldsymbol{\sigma}$ is equivalent to a PD action on the error if $\boldsymbol{\sigma}$ is taken as

$$\boldsymbol{\sigma} = \dot{\boldsymbol{q}}_r - \dot{\boldsymbol{q}} = \dot{\tilde{\boldsymbol{q}}} + \boldsymbol{\Lambda}\tilde{\boldsymbol{q}}. \tag{8.93}$$

Substituting (8.91) into (8.90) and accounting for (8.93) yields

$$\boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{\sigma}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{\sigma} + \boldsymbol{F}\boldsymbol{\sigma} + \boldsymbol{K}_D\boldsymbol{\sigma} = \boldsymbol{0}. \tag{8.94}$$

Consider the Lyapunov function candidate

$$V(\boldsymbol{\sigma},\tilde{\boldsymbol{q}}) = \frac{1}{2}\boldsymbol{\sigma}^T\boldsymbol{B}(\boldsymbol{q})\boldsymbol{\sigma} + \frac{1}{2}\tilde{\boldsymbol{q}}^T\boldsymbol{M}\tilde{\boldsymbol{q}} > 0 \qquad \forall \boldsymbol{\sigma},\tilde{\boldsymbol{q}} \neq \boldsymbol{0}, \tag{8.95}$$

where $\boldsymbol{M}$ is an $(n \times n)$ symmetric positive definite matrix; the introduction of the second term in (8.95) is necessary to obtain a Lyapunov function of the entire system state which vanishes for $\tilde{\boldsymbol{q}} = \boldsymbol{0}$ and $\dot{\tilde{\boldsymbol{q}}} = \boldsymbol{0}$. The time derivative of V along the trajectories of system (8.94) is

$$\begin{aligned}\dot{V} &= \boldsymbol{\sigma}^T\boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{\sigma}} + \frac{1}{2}\boldsymbol{\sigma}^T\dot{\boldsymbol{B}}(\boldsymbol{q})\boldsymbol{\sigma} + \tilde{\boldsymbol{q}}^T\boldsymbol{M}\dot{\tilde{\boldsymbol{q}}} \\ &= -\boldsymbol{\sigma}^T(\boldsymbol{F} + \boldsymbol{K}_D)\boldsymbol{\sigma} + \tilde{\boldsymbol{q}}^T\boldsymbol{M}\dot{\tilde{\boldsymbol{q}}},\end{aligned} \tag{8.96}$$

where the skew-symmetry propertyv of the matrix $\boldsymbol{N} = \dot{\boldsymbol{B}} - 2\boldsymbol{C}$ has been exploited. In view of the expression of $\boldsymbol{\sigma}$ in (8.93), with diagonal $\boldsymbol{\Lambda}$ and $\boldsymbol{K}_D$, it is convenient to choose $\boldsymbol{M} = 2\boldsymbol{\Lambda}\boldsymbol{K}_D$; this leads to

$$\dot{V} = -\boldsymbol{\sigma}^T\boldsymbol{F}\boldsymbol{\sigma} - \dot{\tilde{\boldsymbol{q}}}^T\boldsymbol{K}_D\dot{\tilde{\boldsymbol{q}}} - \tilde{\boldsymbol{q}}^T\boldsymbol{\Lambda}\boldsymbol{K}_D\boldsymbol{\Lambda}\tilde{\boldsymbol{q}}. \tag{8.97}$$

This expression shows that the time derivative is negative definite since it vanishes only if $\tilde{\boldsymbol{q}} \equiv \boldsymbol{0}$ and $\dot{\tilde{\boldsymbol{q}}} \equiv \boldsymbol{0}$; thus, it follows that the origin of the state space $[\,\tilde{\boldsymbol{q}}^T \quad \boldsymbol{\sigma}^T\,]^T = \boldsymbol{0}$ is *globally asymptotically stable*. It is worth noticing that, unlike the robust control case, the error trajectory tends to the subspace $\boldsymbol{\sigma} = \boldsymbol{0}$ without the need of a high-frequency control.

On the basis of this notable result, the *control* law can be made *adaptive* with respect to the vector of parameters $\boldsymbol{\pi}$.

Suppose that the computational model has the same structure as that of the manipulator dynamic model, but its parameters are not known exactly. The control law (8.91) is then modified into

$$\begin{aligned}\boldsymbol{u} &= \widehat{\boldsymbol{B}}(\boldsymbol{q})\ddot{\boldsymbol{q}}_r + \widehat{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}_r + \widehat{\boldsymbol{F}}\dot{\boldsymbol{q}}_r + \widehat{\boldsymbol{g}} + \boldsymbol{K}_D\boldsymbol{\sigma} \\ &= \boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\ddot{\boldsymbol{q}}_r)\widehat{\boldsymbol{\pi}} + \boldsymbol{K}_D\boldsymbol{\sigma},\end{aligned} \tag{8.98}$$

where $\widehat{\boldsymbol{\pi}}$ represents the available estimate on the parameters and, accordingly, $\widehat{\boldsymbol{B}}$, $\widehat{\boldsymbol{C}}$, $\widehat{\boldsymbol{F}}$, $\widehat{\boldsymbol{g}}$ denote the estimated terms in the dynamic model. Substituting control (8.98) into (8.90) gives

$$\begin{aligned}\boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{\sigma}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{\sigma} + \boldsymbol{F}\boldsymbol{\sigma} + \boldsymbol{K}_D\boldsymbol{\sigma} &= -\widetilde{\boldsymbol{B}}(\boldsymbol{q})\ddot{\boldsymbol{q}}_r - \widetilde{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}_r - \widetilde{\boldsymbol{F}}\dot{\boldsymbol{q}}_r - \widetilde{\boldsymbol{g}}(\boldsymbol{q}) \\ &= -\boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\ddot{\boldsymbol{q}}_r)\widetilde{\boldsymbol{\pi}},\end{aligned} \tag{8.99}$$

where the property of linearity in the error parameter vector

$$\widetilde{\boldsymbol{\pi}} = \widehat{\boldsymbol{\pi}} - \boldsymbol{\pi} \tag{8.100}$$

has been conveniently exploited. In view of (8.63), the modelling error is characterized by

$$\widetilde{\boldsymbol{B}} = \widehat{\boldsymbol{B}} - \boldsymbol{B} \qquad \widetilde{\boldsymbol{C}} = \widehat{\boldsymbol{C}} - \boldsymbol{C} \qquad \widetilde{\boldsymbol{F}} = \widehat{\boldsymbol{F}} - \boldsymbol{F} \qquad \widetilde{\boldsymbol{g}} = \widehat{\boldsymbol{g}} - \boldsymbol{g}. \tag{8.101}$$

It is worth remarking that, in view of position (8.92), the matrix $\boldsymbol{Y}$ does not depend on the actual joint accelerations but only on their desired values; this avoids problems due to direct measurement of acceleration.

At this point, modify the Lyapunov function candidate in (8.95) into the form

$$V(\boldsymbol{\sigma},\widetilde{\boldsymbol{q}},\widetilde{\boldsymbol{\pi}}) = \frac{1}{2}\boldsymbol{\sigma}^T\boldsymbol{B}(\boldsymbol{q})\boldsymbol{\sigma} + \widetilde{\boldsymbol{q}}^T\boldsymbol{\Lambda}\boldsymbol{K}_D\widetilde{\boldsymbol{q}} + \frac{1}{2}\widetilde{\boldsymbol{\pi}}^T\boldsymbol{K}_\pi\widetilde{\boldsymbol{\pi}} > 0 \qquad \forall\boldsymbol{\sigma},\widetilde{\boldsymbol{q}},\widetilde{\boldsymbol{\pi}} \neq \boldsymbol{0}, \tag{8.102}$$

which features an additional term accounting for the parameter error (8.100), with $\boldsymbol{K}_\pi$ symmetric positive definite. The time derivative of V along the trajectories of system (8.99) is

$$\dot{V} = -\boldsymbol{\sigma}^T\boldsymbol{F}\boldsymbol{\sigma} - \dot{\widetilde{\boldsymbol{q}}}^T\boldsymbol{K}_D\dot{\widetilde{\boldsymbol{q}}} - \widetilde{\boldsymbol{q}}^T\boldsymbol{\Lambda}\boldsymbol{K}_D\boldsymbol{\Lambda}\widetilde{\boldsymbol{q}} + \widetilde{\boldsymbol{\pi}}^T\big(\boldsymbol{K}_\pi\dot{\widetilde{\boldsymbol{\pi}}} - \boldsymbol{Y}^T(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\ddot{\boldsymbol{q}}_r)\boldsymbol{\sigma}\big). \tag{8.103}$$

If the estimate of the parameter vector is updated as in the adaptive law

$$\dot{\widehat{\boldsymbol{\pi}}} = \boldsymbol{K}_\pi^{-1}\boldsymbol{Y}^T(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\ddot{\boldsymbol{q}}_r)\boldsymbol{\sigma}, \tag{8.104}$$

the expression in (8.103) becomes

$$\dot{V} = -\boldsymbol{\sigma}^T\boldsymbol{F}\boldsymbol{\sigma} - \dot{\widetilde{\boldsymbol{q}}}^T\boldsymbol{K}_D\dot{\widetilde{\boldsymbol{q}}} - \widetilde{\boldsymbol{q}}^T\boldsymbol{\Lambda}\boldsymbol{K}_D\boldsymbol{\Lambda}\widetilde{\boldsymbol{q}}$$

since $\dot{\widehat{\boldsymbol{\pi}}} = \dot{\widetilde{\boldsymbol{\pi}}} - \boldsymbol{\pi}$ is constant.

By an argument similar to above, it is not difficult to show that the trajectories of the manipulator described by the model

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u},$$

under the control law

$$\boldsymbol{u} = \boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\ddot{\boldsymbol{q}}_r)\widehat{\boldsymbol{\pi}} + \boldsymbol{K}_D(\dot{\widetilde{\boldsymbol{q}}} + \boldsymbol{\Lambda}\widetilde{\boldsymbol{q}})$$

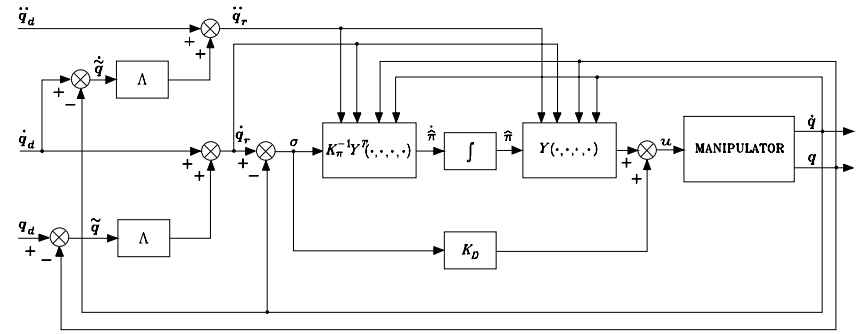

Fig. 8.26. Block scheme of joint space adaptive control

and the parameter adaptive law

$$\dot{\widehat{\boldsymbol{\pi}}} = \boldsymbol{K}_\pi^{-1}\boldsymbol{Y}^T(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\ddot{\boldsymbol{q}}_r)(\dot{\widetilde{\boldsymbol{q}}} + \boldsymbol{\Lambda}\widetilde{\boldsymbol{q}}),$$

globally asymptotically converge to $\boldsymbol{\sigma} = \boldsymbol{0}$ and $\widetilde{\boldsymbol{q}} = \boldsymbol{0}$, which implies *convergence to zero of* $\widetilde{\boldsymbol{q}}$, $\dot{\widetilde{\boldsymbol{q}}}$, and *boundedness of* $\widehat{\boldsymbol{\pi}}$. The equation in (8.99) shows that asymptotically it is

$$\boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\ddot{\boldsymbol{q}}_r)(\widehat{\boldsymbol{\pi}} - \boldsymbol{\pi}) = \boldsymbol{0}. \tag{8.105}$$

This equation does not imply that $\widehat{\boldsymbol{\pi}}$ tends to $\boldsymbol{\pi}$; indeed, convergence of parameters to their true values depends on the structure of the matrix $\boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\ddot{\boldsymbol{q}}_r)$ and then on the desired and actual trajectories. Nonetheless, the followed approach is aimed at solving a *direct* adaptive control problem, i.e., finding a control law that ensures limited tracking errors, and not at determining the actual parameters of the system (as in an indirect adaptive control problem). The resulting block scheme is illustrated in Fig. 8.26. To summarize, the above control law is formed by three different contributions:

- The term $\boldsymbol{Y}\widehat{\boldsymbol{\pi}}$ describes a control action of inverse dynamics type which ensures an *approximate compensation of nonlinear effects and joint decoupling*.
- The term $\boldsymbol{K}_D\boldsymbol{\sigma}$ introduces a *stabilizing linear control action of PD type on the tracking error*.
- The vector of parameter estimates $\widehat{\boldsymbol{\pi}}$ is updated by an *adaptive law of gradient type* so as to ensure asymptotic compensation of the terms in the manipulator dynamic model; the matrix $\boldsymbol{K}_\pi$ determines the convergence rate of parameters to their asymptotic values.

Notice that, with $\boldsymbol{\sigma} \approx \boldsymbol{0}$, the control law (8.98) is equivalent to a pure inverse dynamics compensation of the computed torque type on the basis of

desired velocities and accelerations; this is made possible by the fact that $\boldsymbol{Y}\hat{\boldsymbol{\pi}} \approx \boldsymbol{Y}\boldsymbol{\pi}$.

The control law with parameter adaptation requires the availability of a complete computational model and it does not feature any action aimed at reducing the effects of external disturbances. Therefore, a performance degradation is expected whenever unmodelled dynamic effects, e.g., when a reduced computational model is used, or external disturbances occur. In both cases, the effects induced on the output variables are attributed by the controller to parameter estimate mismatching; as a consequence, the control law attempts to counteract those effects by acting on quantities that did not provoke them originally.

On the other hand, robust control techniques provide a natural rejection to external disturbances, although they are sensitive to unmodelled dynamics; this rejection is provided by a high-frequency commuted control action that constrains the error trajectories to stay on the sliding subspace. The resulting inputs to the mechanical structure may be unacceptable. This inconvenience is in general not observed with the adoption of adaptive control techniques whose action has a naturally smooth time behaviour.

8.6 Operational Space Control

In all the above control schemes, it was always assumed that the desired trajectory is available in terms of the time sequence of the values of joint position, velocity and acceleration. Accordingly, the error for the control schemes was expressed in the joint space.

As often pointed out, motion specifications are usually assigned in the operational space, and then an inverse kinematics algorithm has to be utilized to transform operational space references into the corresponding joint space references. The process of kinematic inversion has an increasing computational load when, besides inversion of direct kinematics, inversion of first-order and second-order differential kinematics is also required to transform the desired time history of end-effector position, velocity and acceleration into the corresponding quantities at the joint level. It is for this reason that current industrial robot control systems compute the joint positions through kinematics inversion, and then perform a numerical differentiation to compute velocities and accelerations.

A different approach consists of considering control schemes developed directly in the operational space. If the motion is specified in terms of operational space variables, the measured joint space variables can be transformed into the corresponding operational space variables through direct kinematics relations. Comparing the desired input with the reconstructed variables allows the design of feedback control loops where trajectory inversion is replaced with a suitable coordinate transformation embedded in the feedback loop.

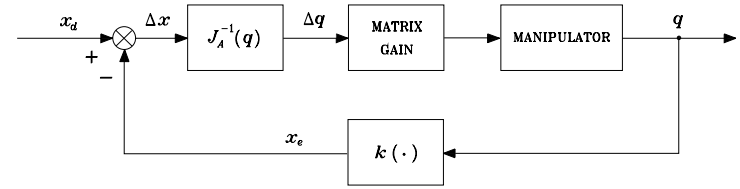

Fig. 8.27. Block scheme of Jacobian inverse control

All operational space control schemes present considerable computational requirements, in view of the necessity to perform a number of computations in the feedback loop which are somewhat representative of inverse kinematics functions. With reference to a numerical implementation, the presence of a computationally demanding load requires sampling times that may lead to degrading the performance of the overall control system.

In the face of the above limitations, it is worth presenting *operational space control* schemes, whose utilization becomes necessary when the problem of controlling interaction between the manipulator and the environment is of concern. In fact, joint space control schemes suffice only for motion control in the free space. When the manipulator's end-effector is constrained by the environment, e.g., in the case of end-effector in contact with an elastic environment, it is necessary to control both positions and contact forces and it is convenient to refer to operational space control schemes. Hence, below some solutions are presented; these are worked out for motion control, but they constitute the premise for the force/position control strategies that will be illustrated in the next chapter.

8.6.1 General Schemes

As pointed out above, operational space control schemes are based on a direct comparison of the inputs, specifying operational space trajectories, with the measurements of the corresponding manipulator outputs. It follows that the control system should incorporate some actions that allow the transformation from the operational space, in which the error is specified, to the joint space, in which control generalized forces are developed.

A possible control scheme that can be devised is the so-called *Jacobian inverse control* (Fig. 8.27). In this scheme, the end-effector pose in the operational space $\boldsymbol{x}_e$ is compared with the corresponding desired quantity $\boldsymbol{x}_d$, and then an operational space deviation $\Delta\boldsymbol{x}$ can be computed. Assumed that this deviation is sufficiently small for a good control system, $\Delta\boldsymbol{x}$ can be transformed into a corresponding joint space deviation $\Delta\boldsymbol{q}$ through the inverse of the manipulator Jacobian. Then, the control input generalized forces can be computed on the basis of this deviation through a suitable feedback matrix gain. The result is a presumable reduction of $\Delta\boldsymbol{q}$ and in turn of $\Delta\boldsymbol{x}$. In other words, the Jacobian inverse control leads to an overall system that intuitively

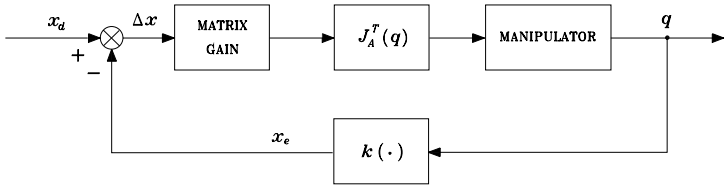

Fig. 8.28. Block scheme of Jacobian transpose control

behaves like a mechanical system with a generalized n-dimensional spring in the joint space, whose constant stiffness is determined by the feedback matrix gain. The role of such system is to take the deviation $\Delta\boldsymbol{q}$ to zero. If the matrix gain is diagonal, the generalized spring corresponds to n independent elastic elements, one for each joint.

A conceptually analogous scheme is the so-called *Jacobian transpose control* (Fig. 8.28). In this case, the operational space error is treated first through a matrix gain. The output of this block can then be considered as the elastic force generated by a generalized spring whose function in the operational space is that to reduce or to cancel the position deviation $\Delta\boldsymbol{x}$. In other words, the resulting force drives the end-effector along a direction so as to reduce $\Delta\boldsymbol{x}$. This operational space force has then to be transformed into the joint space generalized forces, through the transpose of the Jacobian, so as to realize the described behaviour.

Both Jacobian inverse and transpose control schemes have been derived in an intuitive fashion. Hence, there is no guarantee that such schemes are effective in terms of stability and trajectory tracking accuracy. These problems can be faced by presenting two mathematical solutions below, which will be shown to be substantially equivalent to the above schemes.

8.6.2 PD Control with Gravity Compensation

By analogy with joint space stability analysis, given a *constant* end-effector pose $\boldsymbol{x}_d$, it is desired to find the control structure so that the operational space error

$$\widetilde{\boldsymbol{x}} = \boldsymbol{x}_d - \boldsymbol{x}_e \tag{8.106}$$

tends asymptotically to zero. Choose the following positive definite quadratic form as a Lyapunov function candidate:

$$V(\dot{\boldsymbol{q}}, \widetilde{\boldsymbol{x}}) = \frac{1}{2}\dot{\boldsymbol{q}}^T\boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}} + \frac{1}{2}\widetilde{\boldsymbol{x}}^T\boldsymbol{K}_P\widetilde{\boldsymbol{x}} > 0 \qquad \forall \dot{\boldsymbol{q}}, \widetilde{\boldsymbol{x}} \neq \boldsymbol{0}, \tag{8.107}$$

with $\boldsymbol{K}_P$ a symmetric positive definite matrix. Differentiating (8.107) with respect to time gives

$$\dot{V} = \dot{\boldsymbol{q}}^T\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \frac{1}{2}\dot{\boldsymbol{q}}^T\dot{\boldsymbol{B}}(\boldsymbol{q})\dot{\boldsymbol{q}} + \dot{\widetilde{\boldsymbol{x}}}^T\boldsymbol{K}_P\widetilde{\boldsymbol{x}}.$$

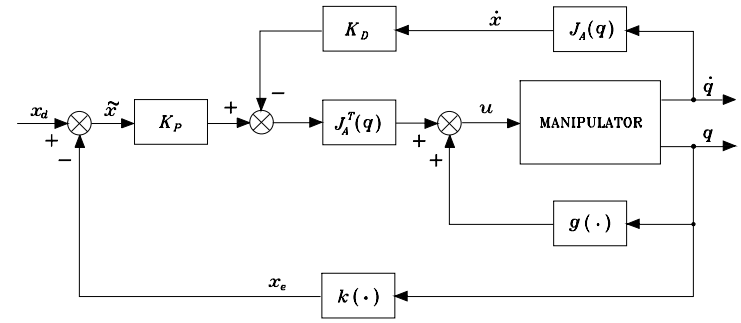

Fig. 8.29. Block scheme of operational space PD control with gravity compensation

Since $\dot{\boldsymbol{x}}_d = \boldsymbol{0}$, in view of (3.62) it is

$$\dot{\widetilde{\boldsymbol{x}}} = -\boldsymbol{J}_A(\boldsymbol{q})\dot{\boldsymbol{q}}$$

and then

$$\dot{V} = \dot{\boldsymbol{q}}^T\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \frac{1}{2}\dot{\boldsymbol{q}}^T\dot{\boldsymbol{B}}(\boldsymbol{q})\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}^T\boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{K}_P\widetilde{\boldsymbol{x}}. \tag{8.108}$$

By recalling the expression of the joint space manipulator dynamic model in (8.7) and the property in (7.49), the expression in (8.108) becomes

$$\dot{V} = -\dot{\boldsymbol{q}}^T\boldsymbol{F}\dot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T\big(\boldsymbol{u} - \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{K}_P\widetilde{\boldsymbol{x}}\big). \tag{8.109}$$

This equation suggests the structure of the controller; in fact, by choosing the control law

$$\boldsymbol{u} = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{K}_P\widetilde{\boldsymbol{x}} - \boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{K}_D\boldsymbol{J}_A(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{8.110}$$

with $\boldsymbol{K}_D$ positive definite, (8.109) becomes

$$\dot{V} = -\dot{\boldsymbol{q}}^T\boldsymbol{F}\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}^T\boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{K}_D\boldsymbol{J}_A(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{8.111}$$

As can be seen from Fig. 8.29, the resulting block scheme reveals an analogy with the scheme of Fig. 8.28. Control law (8.110) performs a *nonlinear compensating action of joint space gravitational forces* and an *operational space linear PD control action*. The last term has been introduced to enhance system damping; in particular, if measurement of $\dot{\boldsymbol{x}}$ is deduced from that of $\dot{\boldsymbol{q}}$, one can simply choose the derivative term as $-\boldsymbol{K}_D\dot{\boldsymbol{q}}$.

The expression in (8.111) shows that, for any system trajectory, the Lyapunov function decreases as long as $\dot{\boldsymbol{q}} \neq \boldsymbol{0}$. The system then reaches an *equilibrium posture*. By a stability argument similar to that in the joint space (see (8.52)–(8.54)) this posture is determined by

$$\boldsymbol{J}_A^T(\boldsymbol{q})\boldsymbol{K}_P\widetilde{\boldsymbol{x}} = \boldsymbol{0}. \tag{8.112}$$

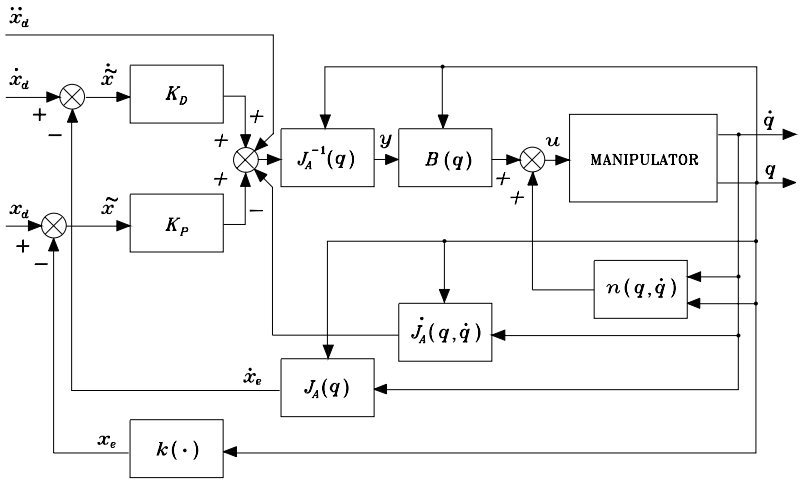

Fig. 8.30. Block scheme of operational space inverse dynamics control

From (8.112) it can be recognized that, under the assumption of *full-rank* Jacobian, it is

$$\widetilde{\boldsymbol{x}} = \boldsymbol{x}_d - \boldsymbol{x}_e = \boldsymbol{0},$$

i.e., the sought result.

If measurements of $\boldsymbol{x}_e$ and $\dot{\boldsymbol{x}}_e$ are made directly in the operational space, $\boldsymbol{k}(\boldsymbol{q})$ and $\boldsymbol{J}_A(\boldsymbol{q})$ in the scheme of Fig. 8.45 are just indicative of direct kinematics functions; it is, however, necessary to measure $\boldsymbol{q}$ to update both $\boldsymbol{J}_A^T(\boldsymbol{q})$ and $\boldsymbol{g}(\boldsymbol{q})$ on-line. If measurements of operational space quantities are indirect, the controller has to compute the direct kinematics functions, too.

8.6.3 Inverse Dynamics Control

Consider now the problem of tracking an operational space trajectory. Recall the manipulator dynamic model in the form (8.55)

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{u},$$

where $\boldsymbol{n}$ is given by (8.56). As in (8.57), the choice of the *inverse dynamics linearizing control*

$$\boldsymbol{u} = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{y} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})$$

leads to the system of double integrators

$$\ddot{\boldsymbol{q}} = \boldsymbol{y}. \tag{8.113}$$

The new control input $\boldsymbol{y}$ is to be designed so as to yield tracking of a trajectory specified by $\boldsymbol{x}_d(t)$. To this end, the second-order differential equation in the form (3.98)

$$\ddot{\boldsymbol{x}}_e = \boldsymbol{J}_A(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_A(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$$

suggests, for a nonredundant manipulator, the choice of the control law — formally analogous to (3.102) —

$$\boldsymbol{y} = \boldsymbol{J}_A^{-1}(\boldsymbol{q})\big(\ddot{\boldsymbol{x}}_d + \boldsymbol{K}_D\dot{\widetilde{\boldsymbol{x}}} + \boldsymbol{K}_P\widetilde{\boldsymbol{x}} - \dot{\boldsymbol{J}}_A(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}\big) \tag{8.114}$$

with $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$ positive definite (diagonal) matrices. In fact, substituting (8.114) into (8.113) gives

$$\ddot{\widetilde{\boldsymbol{x}}} + \boldsymbol{K}_D\dot{\widetilde{\boldsymbol{x}}} + \boldsymbol{K}_P\widetilde{\boldsymbol{x}} = \boldsymbol{0} \tag{8.115}$$

which describes the operational space error dynamics, with $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$ determining the error convergence rate to zero. The resulting inverse dynamics control scheme is reported in Fig. 8.30, which confirms the anticipated analogy with the scheme of Fig. 8.27. Again in this case, besides $\boldsymbol{x}_e$ and $\dot{\boldsymbol{x}}_e$, $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ are also to be measured. If measurements of $\boldsymbol{x}_e$ and $\dot{\boldsymbol{x}}_e$ are indirect, the controller must compute the direct kinematics functions $\boldsymbol{k}(\boldsymbol{q})$ and $\boldsymbol{J}_A(\boldsymbol{q})$ on-line.

A critical analysis of the schemes in Figs. 8.29, 8.30 reveals that the design of an operational space controller always requires computation of manipulator Jacobian. As a consequence, controlling a manipulator in the operational space is in general more complex than controlling it in the joint space. In fact, the presence of *singularities* and/or *redundancy* influences the Jacobian, and the induced effects are somewhat difficult to handle with an operational space controller. For instance, if a singularity occurs for the scheme of Fig. 8.29 and the error enters the null space of the Jacobian, the manipulator gets stuck at a different configuration from the desired one. This problem is even more critical for the scheme of Fig. 8.30 which would require the computation of a DLS inverse of the Jacobian. Yet, for a redundant manipulator, a joint space control scheme is naturally transparent to this situation, since redundancy has already been solved by inverse kinematics, whereas an operational space control scheme should incorporate a redundancy handling technique inside the feedback loop.

As a final remark, the above operational space control schemes have been derived with reference to a minimal description of orientation in terms of Euler angles. It is understood that, similar to what is presented in Sect. 3.7.3 for inverse kinematics algorithms, it is possible to adopt different definitions of orientation error, e.g., based on the angle and axis or the unit quaternion. The advantage is the use of the geometric Jacobian in lieu of the analytical Jacobian. The price to pay, however, is a more complex analysis of the stability and convergence characteristics of the closed-loop system. Even the inverse dynamics control scheme will not lead to a homogeneous error equation, and a Lyapunov argument should be invoked to ascertain its stability.

8.7 Comparison Among Various Control Schemes

In order to make a comparison between the various control schemes presented, consider the two-link planar arm with the same data of Example 7.2:

$$a_1 = a_2 = 1\,\text{m} \quad \ell_1 = \ell_2 = 0.5\,\text{m} \quad m_{\ell_1} = m_{\ell_2} = 50\,\text{kg} \quad I_{\ell_1} = I_{\ell_2} = 10\,\text{kg}\cdot\text{m}^2$$

$$k_{r1} = k_{r2} = 100 \qquad m_{m1} = m_{m2} = 5\,\text{kg} \qquad I_{m1} = I_{m2} = 0.01\,\text{kg}\cdot\text{m}^2.$$

The arm is assumed to be driven by two equal actuators with the following data:

$$F_{m1} = F_{m2} = 0.01\,\text{N}\cdot\text{m}\cdot\text{s/rad} \qquad R_{a1} = R_{a2} = 10\,\text{ohm}$$

$$k_{t1} = k_{t2} = 2\,\text{N}\cdot\text{m/A} \qquad k_{v1} = k_{v2} = 2\,\text{V}\cdot\text{s/rad};$$

it can be verified that $F_{m_i} \ll k_{vi}k_{ti}/R_{ai}$ for $i = 1, 2$.

The desired tip trajectories have a typical trapezoidal velocity profile, and thus it is anticipated that sharp torque variations will be induced. The tip path is a motion of 1.6 m along the horizontal axis, as in the path of Example 7.2. In the first case (*fast* trajectory), the acceleration time is 0.6 s and the maximum velocity is 1 m/s. In the second case (*slow* trajectory), the acceleration time is 0.6 s and the maximum velocity is 0.25 m/s. The motion of the controlled arm was simulated on a computer, by adopting a discrete-time implementation of the controller with a sampling time of 1 ms.

The following control schemes in the joint space and in the operational space have been utilized; an (analytic) inverse kinematics solution has been implemented to generate the reference inputs to the joint space control schemes:

A. Independent joint control with position and velocity feedback (Fig. 5.11) with the following data for each joint servo:

$$K_P = 5 \qquad K_V = 10 \qquad k_{TP} = k_{TV} = 1,$$

corresponding to $\omega_n = 5\,\text{rad/s}$ and $\zeta = 0.5$.

B. Independent joint control with position, velocity and acceleration feedback (Fig. 8.9) with the following data for each joint servo:

$$K_P = 5 \qquad K_V = 10 \qquad K_A = 2 \qquad k_{TP} = k_{TV} = k_{TA} = 1,$$

corresponding to $\omega_n = 5\,\text{rad/s}$, $\zeta = 0.5$, $X_R = 100$. To reconstruct acceleration, a first-order filter has been utilized (Fig. 8.11) characterized by $\omega_{3f} = 100\,\text{rad/s}$.

C. As in scheme **A** with the addition of a decentralized feedforward action (Fig. 8.13).

D. As in scheme **B** with the addition of a decentralized feedforward action (Fig. 8.14).

E. Joint space computed torque control (Fig. 8.19) with feedforward compensation of the diagonal terms of the inertia matrix and of gravitational terms, and decentralized feedback controllers as in scheme **A**.

F. Joint space PD control with gravity compensation (Fig. 8.20), modified by the addition of a feedforward velocity term $\boldsymbol{K}_D\dot{\boldsymbol{q}}_d$, with the following data:

$$\boldsymbol{K}_P = 3750\boldsymbol{I}_2 \qquad \boldsymbol{K}_D = 750\boldsymbol{I}_2.$$

G. Joint space inverse dynamics control (Fig. 8.22) with the following data:

$$\boldsymbol{K}_P = 25\boldsymbol{I}_2 \qquad \boldsymbol{K}_D = 5\boldsymbol{I}_2.$$

H. Joint space robust control (Fig. 8.23), under the assumption of constant inertia ($\widehat{\boldsymbol{B}} = \bar{\boldsymbol{B}}$) and compensation of friction and gravity ($\widehat{\boldsymbol{n}} = \boldsymbol{F}_v\dot{\boldsymbol{q}} + \boldsymbol{g}$), with the following data:

$$\boldsymbol{K}_P = 25\boldsymbol{I}_2 \qquad \boldsymbol{K}_D = 5\boldsymbol{I}_2 \qquad \boldsymbol{P} = \boldsymbol{I}_2 \qquad \rho = 70 \qquad \epsilon = 0.004.$$

I. As in case **H** with $\epsilon = 0.01$.

J. Joint space adaptive control (Fig. 8.26) with a parameterization of the arm dynamic model (7.82) as in (7.83), (7.84). The initial estimate of the vector $\widehat{\boldsymbol{\pi}}$ is computed on the basis of the nominal parameters. The arm is supposed to carry a load which causes the following variations on the second link parameters:

$$\Delta m_2 = 10\,\text{kg} \qquad \Delta m_2\ell_{C2} = 11\,\text{kg}\cdot\text{m} \qquad \Delta\widehat{I}_2 = 12.12\,\text{kg}\cdot\text{m}^2.$$

This information is obviously utilized only to update the simulated arm model. Further, the following data are set:

$$\boldsymbol{\Lambda} = 5\boldsymbol{I}_2 \qquad \boldsymbol{K}_D = 750\boldsymbol{I}_2 \qquad \boldsymbol{K}_\pi = 0.01\boldsymbol{I}_8.$$

K. Operational space PD control with gravity compensation (Fig. 8.29), modified by the addition of a feedforward velocity term $\boldsymbol{K}_D\dot{\boldsymbol{x}}_d$, with the following data:

$$\boldsymbol{K}_P = 16250\boldsymbol{I}_2 \qquad \boldsymbol{K}_D = 3250\boldsymbol{I}_2.$$

L. Operational space inverse dynamics control (Fig. 8.30) with the following data:

$$\boldsymbol{K}_P = 25\boldsymbol{I}_2 \qquad \boldsymbol{K}_D = 5\boldsymbol{I}_2.$$

It is worth remarking that the adopted model of the dynamic system of arm with drives is that described by (8.7). In the decentralized control schemes **A**–**E**, the joints have been voltage-controlled as in the block scheme of Fig. 8.3, with unit amplifier gains ($\boldsymbol{G}_v = \boldsymbol{I}$). On the other hand, in the centralized control schemes **F**–**L**, the joints have been current-controlled as in the block scheme of Fig. 8.4, with unit amplifier gains ($\boldsymbol{G}_i = \boldsymbol{I}$).

Regarding the parameters of the various controllers, these have been chosen in such a way as to allow a significant comparison of the performance of each scheme in response to congruent control actions. In particular, it can be observed that:

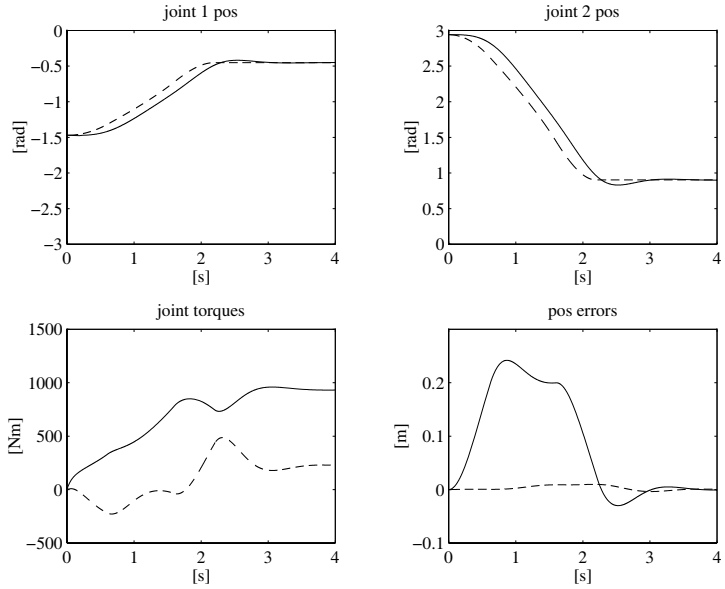

Fig. 8.31. Time history of the joint positions and torques and of the tip position errors for the *fast* trajectory with control scheme **A**

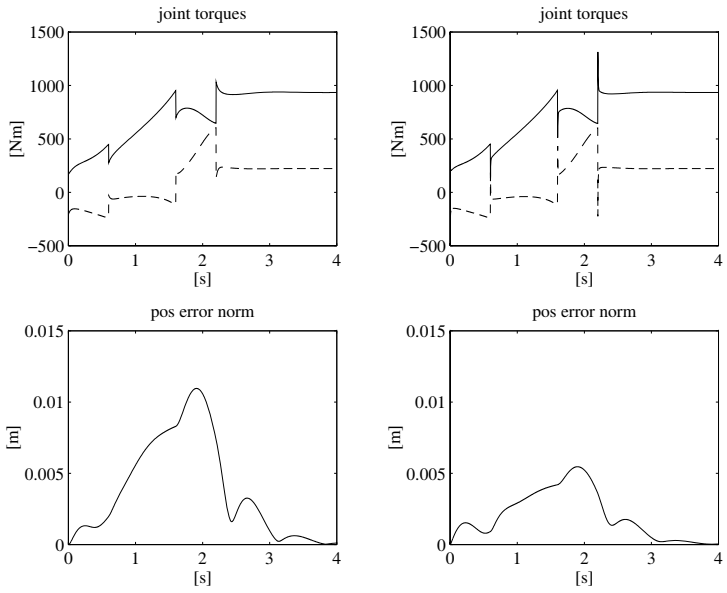

Fig. 8.32. Time history of the joint torques and of the norm of tip position error for the *fast* trajectory; *left*: with control scheme **C**, *right*: with control scheme **D**

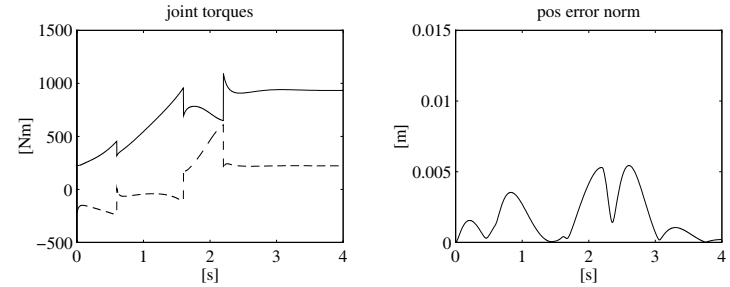

Fig. 8.33. Time history of the joint torques and of the norm of tip position error for the *fast* trajectory with control scheme **E**

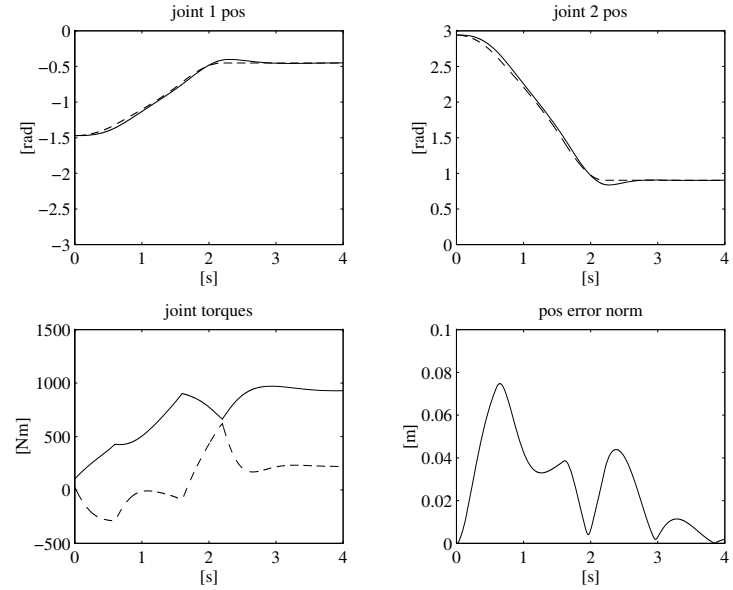

Fig. 8.34. Time history of the joint positions and torques and of the norm of tip position error for the *fast* trajectory with control scheme **F**

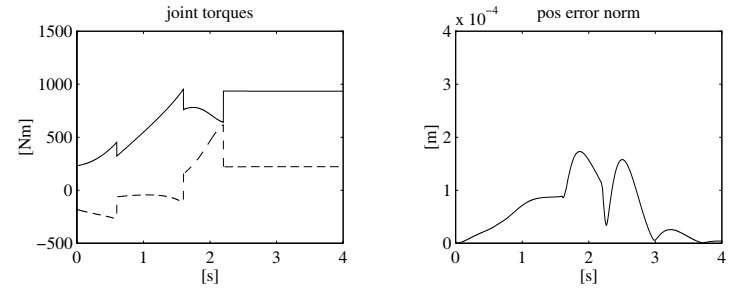

Fig. 8.35. Time history of the joint torques and of the norm of tip position error for the *fast* trajectory with control scheme **G**

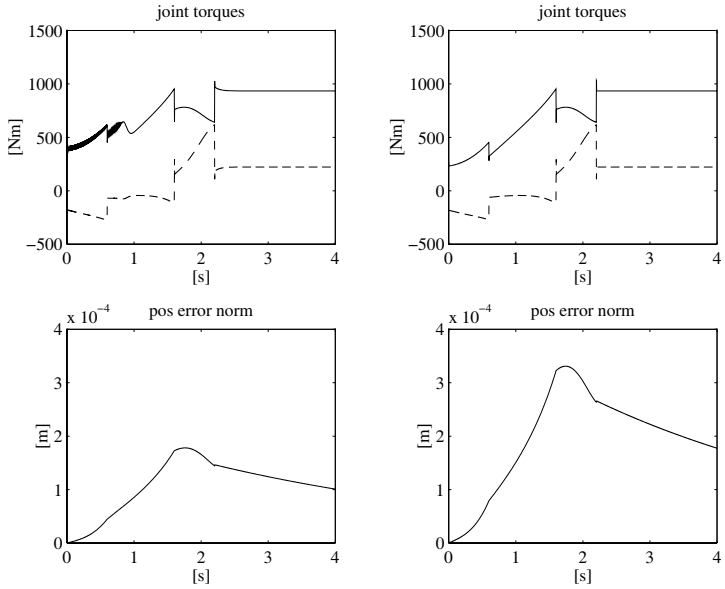

Fig. 8.36. Time history of the joint torques and of the norm of tip position error for the *fast* trajectory; *left*: with control scheme **H**, *right*: with control scheme **I**

- The dynamic behaviour of the joints is the same for schemes **A**–**E**.
- The gains of the PD actions in schemes **G**, **H**, **I** and **L** have been chosen so as to obtain the same natural frequency and damping ratios as those of schemes **A**–**E**.

The results obtained with the various control schemes are illustrated in Figs. 8.31–8.39 for the *fast* trajectory and in Figs. 8.40–8.48 for the *slow* trajectory, respectively. In the case of two quantities represented in the same plot notice that:

- For the joint trajectories, the dashed line indicates the reference trajectory obtained from the tip trajectory via inverse kinematics, while the solid line indicates the actual trajectory followed by the arm.
- For the joint torques, the solid line refers to Joint 1 while the dashed line refers to Joint 2.
- For the tip position error, the solid line indicates the error component along the horizontal axis while the dashed line indicates the error component along the vertical axis.

Finally, the representation scales have been made as uniform as possible in order to allow a more direct comparison of the results.

Regarding performance of the various control schemes for the *fast* trajectory, the obtained results lead to the following considerations.

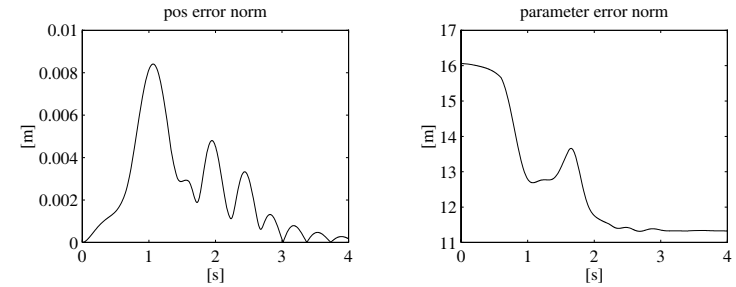

Fig. 8.37. Time history of the norm of tip position error and of the norm of parameter error vector for the *fast* trajectory with control scheme **J**

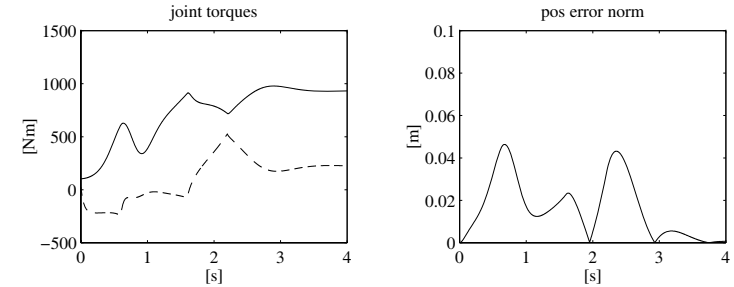

Fig. 8.38. Time history of the joint torques and of the norm of tip position error for the *fast* trajectory with control scheme **K**

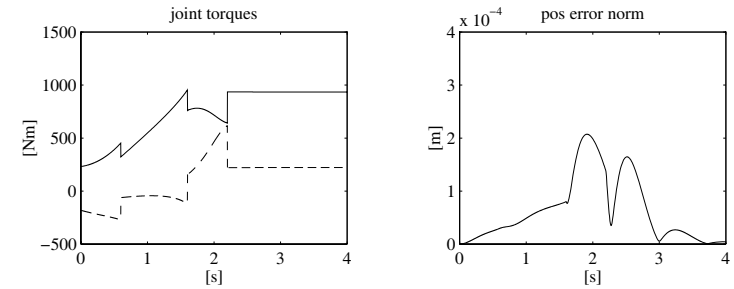

Fig. 8.39. Time history of the joint torques and of the norm of tip position error for the *fast* trajectory with control scheme **L**

Deviation of the actual joint trajectories from the desired ones shows that tracking performance of scheme **A** is quite poor (Fig. 8.31). It should be noticed, however, that the largest contribution to the error is caused by a time lag of the actual trajectory behind the desired one, while the distance

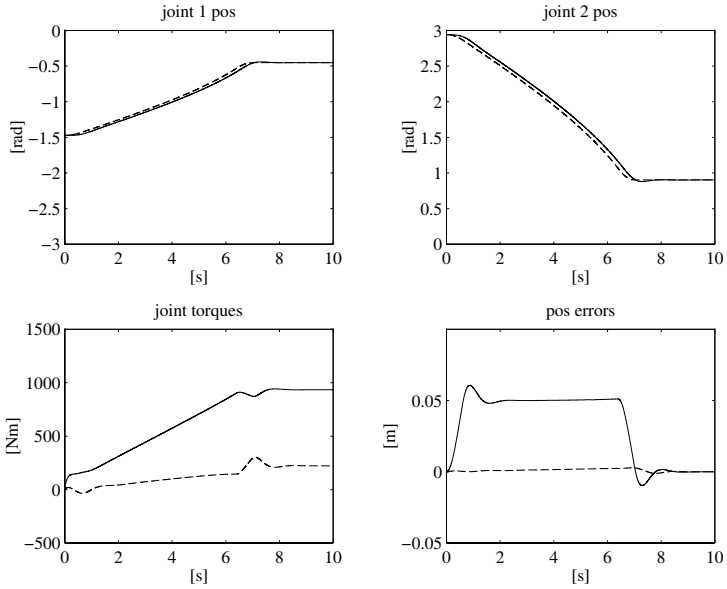

Fig. 8.40. Time history of the joint positions and torques and of the tip position errors for the *slow* trajectory with control scheme **A**

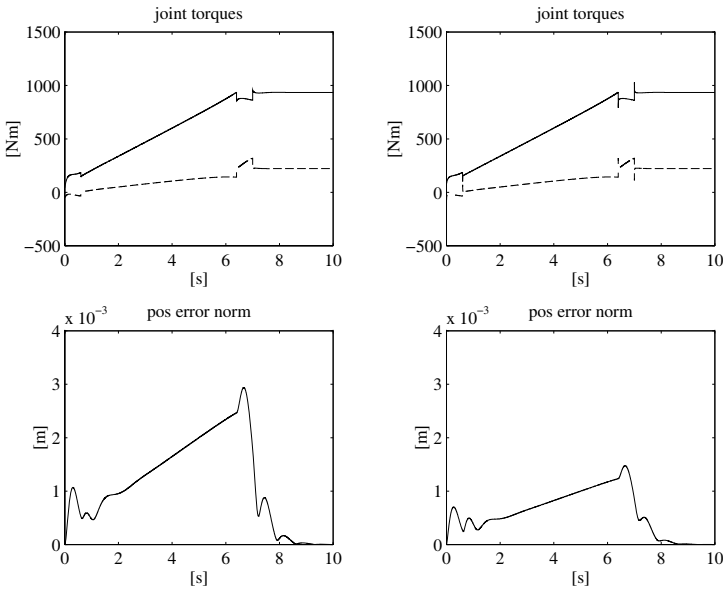

Fig. 8.41. Time history of the joint torques and of the norm of tip position error for the *slow* trajectory; *left*: with control scheme **C**, *right*: with control scheme **D**

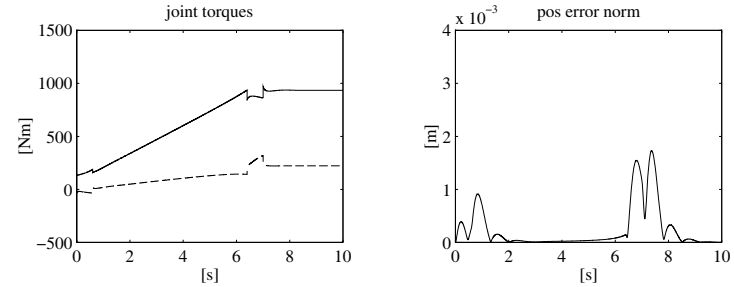

Fig. 8.42. Time history of the joint torques and of the norm of tip position error for the *slow* trajectory with control scheme **E**

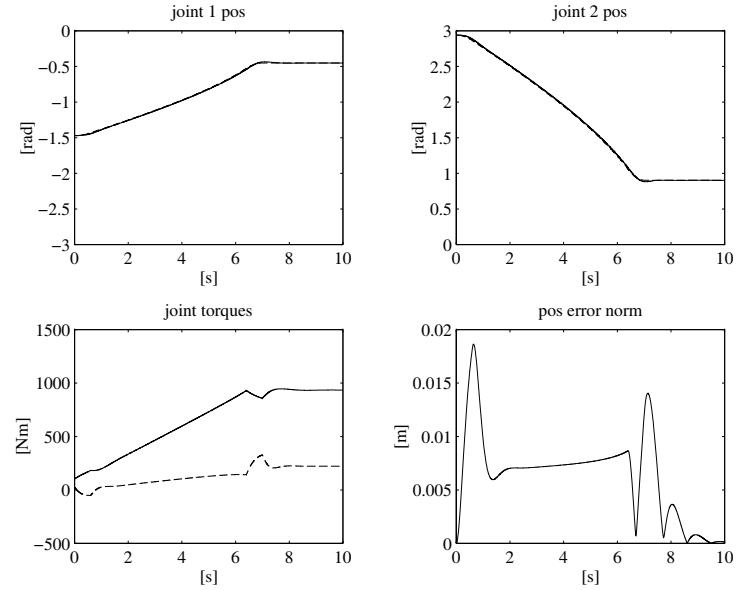

Fig. 8.43. Time history of the joint positions and torques and of the norm of tip position error for the *slow* trajectory with control scheme **F**

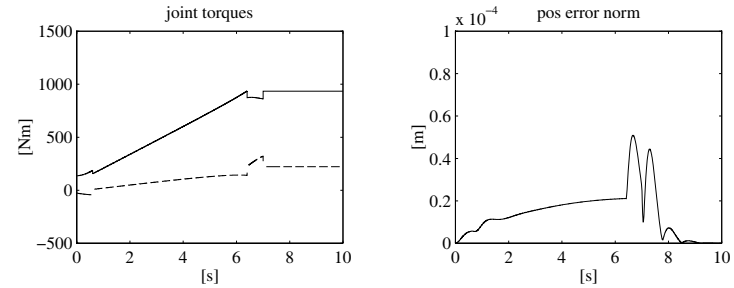

Fig. 8.44. Time history of the joint torques and of the norm of tip position error for the *slow* trajectory with control scheme **G**

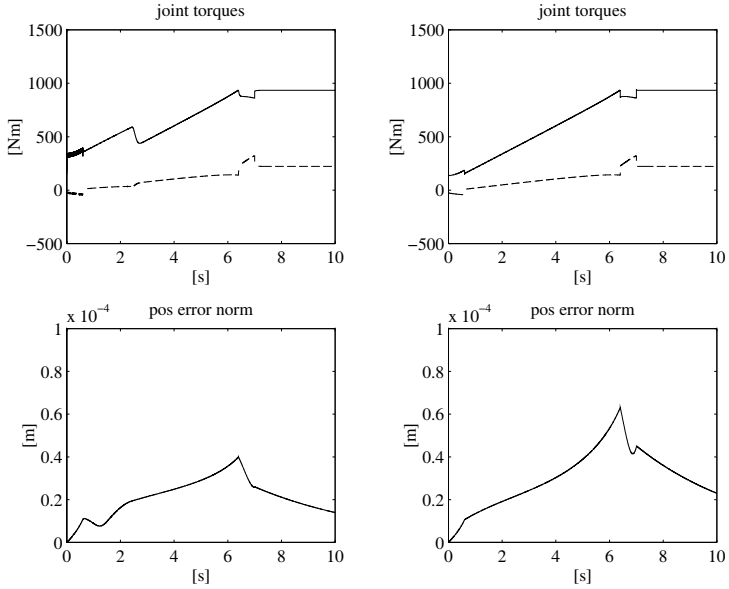

Fig. 8.45. Time history of the joint torques and of the norm of tip position error for the *slow* trajectory; *left*: with control scheme **H**, *right*: with control scheme **I**

of the tip from the geometric path is quite contained. Similar results were obtained with scheme **B**, and then they have not been reported.

With schemes **C** and **D**, an appreciable tracking accuracy improvement is observed (Fig. 8.32), with better performance for the second scheme, thanks to the outer acceleration feedback loop that allows a disturbance rejection factor twice as much as for the first scheme. Notice that the feedforward action yields a set of torques which are closer to the nominal ones required to execute the desired trajectory; the torque time history has a discontinuity in correspondence of the acceleration and deceleration fronts.

The tracking error is further decreased with scheme **E** (Fig. 8.33), by virtue of the additional nonlinear feedforward compensation.

Scheme **F** guarantees stable convergence to the final arm posture with a tracking performance which is better than that of schemes **A** and **B**, thanks to the presence of a velocity feedforward action, but worse than that of schemes **C**–**E**, in view of lack of an acceleration feedforward action (Fig. 8.34).

As would be logical to expect, the best results are observed with scheme **G** for which the tracking error is practically zero, and it is mainly due to numerical discretization of the controller (Fig. 8.35).

It is then worth comparing the performance of schemes **H** and **I** (Fig. 8.36). In fact, the choice of a small threshold value for ϵ (scheme **H**) induces high-

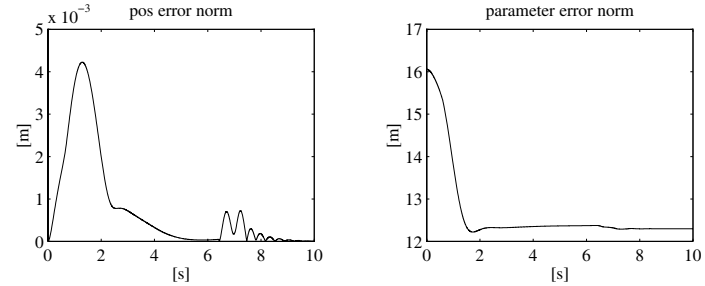

Fig. 8.46. Time history of the norm of tip position error and of the norm of parameter error vector for the *slow* trajectory with control scheme **J**

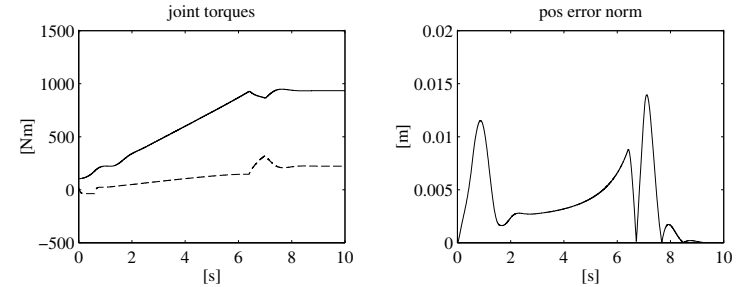

Fig. 8.47. Time history of the joint torques and of the norm of tip position error for the *slow* trajectory with control scheme **K**

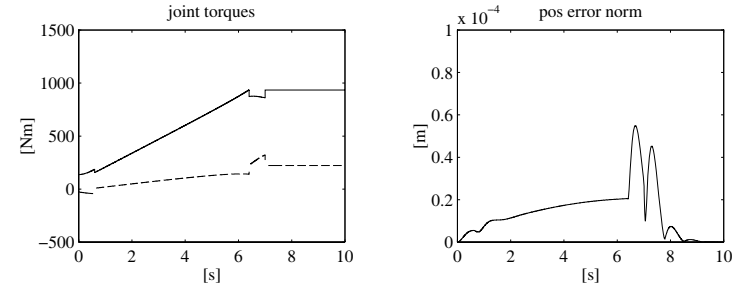

Fig. 8.48. Time history of the joint torques and of the norm of tip position error for the *slow* trajectory with control scheme **L**

frequency components in Joint 1 torque (see the thick portions of the torque plot) at the advantage of a very limited tracking error. As the threshold value is increased (scheme **I**), the torque assumes a smoother behaviour at the expense of a doubled norm of tracking error, though.

For scheme **J**, a lower tracking error than that of scheme **F** is observed, thanks to the effectiveness of the adaptive action on the parameters of the dynamic model. Nonetheless, the parameters do not converge to their nominal values, as confirmed by the time history of the norm of the parameter error vector that reaches a non-null steady-state value (Fig. 8.37).

Finally, the performance of schemes **K** and **L** is substantially comparable to that of corresponding schemes **F** and **G** (Figs. 8.38 and 8.39).

Performance of the various control schemes for the *slow* trajectory is globally better than that for the *fast* trajectory. Such improvement is particularly evident for the decentralized control schemes (Figs. 8.40–8.42), whereas the tracking error reduction for the centralized control schemes is less dramatic (Figs. 8.43–8.48), in view of the small order of magnitude of the errors already obtained for the *fast* trajectory. In any case, as regards performance of each single scheme, it is possible to make a number of remarks analogous to those previously made.

Bibliography

The independent joint control is analyzed in classical texts [180, 120, 200], and scientific articles [19, 127, 141, 101, 39]. Stability of PD control with gravity compensation is proved in [7], on the basis of the notable properties of the dynamic model in [226].

Computed torque control and inverse dynamics control were developed at the beginning of the 1970s. One of the first experimental works is [149]. Other articles on the topic are [83, 4, 117, 121, 126, 227, 29].

The main approaches of robust control are inspired to the work [50]. Among them it is worth citing [212, 84, 130, 219, 205, 216]. Robust controllers based on the high gain concept are presented in [192, 222]. A survey on robust control is [1].

One of the first approaches to adaptive control, based on the assumption of decoupled joint dynamics, is presented in [67]. The first works on adaptive control accounting for the manipultor nonlinear dynamics are [15, 167, 100], yet they exploit the notable properties of the dynamic model only to some extent. The adaptive version of inverse dynamics control is analyzed in [52, 157]. The approach based on the energy properties of the dynamic model has been proposed in [214] and further analyzed in [218]. An interesting tutorial on adaptive control is [175].

Operational space control has been proposed in [114], on the basis of the resolved acceleration control concept [143]. Inverse dynamics control schemes in the operational space are given in [30]. For the extension to redundant manipulators see [102].

Problems

8.1. With reference to the block scheme with position feedback in Fig. 5.10, find the transfer functions of the forward path, the return path, and the closed-loop system.

8.2. With reference to the block scheme with position and velocity feedback in Fig. 5.11, find the transfer functions of the forward path, the return path, and the closed-loop system.

8.3. With reference to the block scheme with position, velocity and acceleration feedback in Fig. 8.9, find the transfer functions of the forward path, the return path, and the closed-loop system.

8.4. For a single joint drive system with the data: $I = 6\,\text{kg}\cdot\text{m}^2$, $R_a = 0.3\,\text{ohm}$, $k_t = 0.5\,\text{N}\cdot\text{m/A}$, $k_v = 0.5\,\text{V}\cdot\text{s/rad}$, $F_m = 0.001\,\text{N}\cdot\text{m}\cdot\text{s/rad}$, find the parameters of the controller with position feedback (unit transducer constant) that yield a closed-loop response with damping ratio $\zeta \geq 0.4$. Discuss disturbance rejection properties.

8.5. For the drive system of Problem 8.4, find the parameters of the controller with position and velocity feedback (unit transducer constants) that yield a closed-loop response with damping ratio $\zeta \geq 0.4$ and natural frequency $\omega_n = 20\,\text{rad/s}$. Discuss disturbance rejection properties.

8.6. For the drive system of Problem 8.4, find the parameters of the controller with position, velocity and acceleration feedback (unit transducer constants) that yield a closed-loop response with damping ratio $\zeta \geq 0.4$, natural frequency $\omega_n = 20\,\text{rad/s}$ and disturbance rejection factor $X_R = 400$. Also, design a first-order filter that allows acceleration measurement reconstruction.

8.7. Verify that the control schemes in Figs. 8.12, 8.13, 8.14 correspond to realizing (8.42), (8.43), (8.44), respectively.

8.8. Verify that the standard regulation schemes in Figs. 8.15, 8.16, 8.17 are equivalent to the schemes in Figs. 8.12, 8.13, 8.14, respectively.

8.9. Prove inequality (8.76).

8.10. For the two-link planar arm with the same data as in Sect. 8.7, design a joint control of PD type with gravity compensation. By means of a computer simulation, verify stability for the following postures $\boldsymbol{q} = [\,\pi/4 \quad -\pi/2\,]^T$ and $\boldsymbol{q} = [\,-\pi \quad -3\pi/4\,]^T$, respectively. Implement the control in discrete-time with a sampling time of 1 ms.

8.11. For the two-link planar arm with the same data as in Sect. 8.7, under the assumption of a concentrated tip payload of mass $m_L = 10\,\text{kg}$, design an independent joint control with feedforward computed torque. Perform a

computer simulation of the motion of the controlled arm along the joint space rectilinear path from $\boldsymbol{q}_i = [\,0 \quad \pi/4\,]^T$ to $\boldsymbol{q}_f = [\,\pi/2 \quad \pi/2\,]^T$ with a trapezoidal velocity profile and a trajectory duration $t_f = 1\,\text{s}$. Implement the control in discrete-time with a sampling time of 1 ms.

8.12. For the two-link planar arm of Problem 8.11, design an inverse dynamics joint control. Perform a computer simulation of the motion of the controlled arm along the trajectory specified in Problem 8.11. Implement the control in discrete-time with a sampling time of 1 ms.

8.13. For the two-link planar arm of Problem 8.11, design a robust joint control. Perform a computer simulation of the motion of the controlled arm along the trajectory specified in Problem 8.11. Implement the control in discrete-time with a sampling time of 1 ms.

8.14. For the two-link planar arm of Problem 8.11, design an adaptive joint control, on the basis of a suitable parameterization of the arm dynamic model. Perform a computer simulation of the motion of the controlled arm along the trajectory specified in Problem 8.11. Implement the control in discrete-time with a sampling time of 1 ms.

8.15. For the two-link planar of Problem 8.11, design a PD control in the operational space with gravity compensation. By means of a computer simulation, verify stability for the following postures $\boldsymbol{p} = [\,0.5 \quad 0.5\,]^T$ and $\boldsymbol{p} = [\,0.6 \quad -0.2\,]^T$, respectively. Implement the control in discrete-time with a sampling time of 1 ms.

8.16. For the two-link planar arm of Problem 8.11, design an inverse dynamics control in the operational space. Perform a computer simulation of the motion of the controlled arm along the operational space rectlinear path from $\boldsymbol{p}(0) = [\,0.7 \quad 0.2\,]^T$ to $\boldsymbol{p}(1) = [\,0.1 \quad -0.6\,]^T$ with a trapezoidal velocity profile and a trajectory duration $t_f = 1\,\text{s}$. Implement the control in discrete-time with a sampling time of 1 ms.

A

Linear Algebra

Since modelling and control of robot manipulators requires an extensive use of *matrices* and *vectors* as well as of matrix and vector *operations*, the goal of this appendix is to provide a brush-up of *linear algebra*.

A.1 Definitions

A *matrix* of dimensions $(m \times n)$, with m and n positive integers, is an array of elements a_{ij} arranged into m *rows* and n *columns*:

$$\boldsymbol{A} = [a_{ij}]_{\substack{i=1,\dots,m \\ j=1,\dots,n}} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}. \tag{A.1}$$

If $m = n$, the matrix is said to be *square*; if $m < n$, the matrix has more columns than rows; if $m > n$ the matrix has more rows than columns. Further, if $n = 1$, the notation (A.1) is used to represent a (column) vector $\boldsymbol{a}$ of dimensions $(m \times 1)$;[1] the elements a_i are said to be vector components.

A square matrix $\boldsymbol{A}$ of dimensions $(n \times n)$ is said to be *upper triangular* if $a_{ij} = 0$ for $i > j$:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ 0 & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{nn} \end{bmatrix};$$

the matrix is said to be *lower triangular* if $a_{ij} = 0$ for $i < j$.

[1] According to standard mathematical notation, small boldface is used to denote vectors while capital boldface is used to denote matrices. Scalars are denoted by roman characters.

An $(n \times n)$ square matrix $\boldsymbol{A}$ is said to be *diagonal* if $a_{ij} = 0$ for $i \neq j$, i.e.,

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{nn} \end{bmatrix} = \text{diag}\{a_{11}, a_{22}, \dots, a_{nn}\}.$$

If an $(n \times n)$ diagonal matrix has all unit elements on the diagonal ($a_{ii} = 1$), the matrix is said to be *identity* and is denoted by $\boldsymbol{I}_n$.[2] A matrix is said to be *null* if all its elements are null and is denoted by $\boldsymbol{O}$. The null column vector is denoted by $\boldsymbol{0}$.

The *transpose* $\boldsymbol{A}^T$ of a matrix $\boldsymbol{A}$ of dimensions $(m \times n)$ is the matrix of dimensions $(n \times m)$ which is obtained from the original matrix by interchanging its rows and columns:

$$\boldsymbol{A}^T = \begin{bmatrix} a_{11} & a_{21} & \dots & a_{m1} \\ a_{12} & a_{22} & \dots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{mn} \end{bmatrix}. \tag{A.2}$$

The transpose of a column vector $\boldsymbol{a}$ is the row vector $\boldsymbol{a}^T$.

An $(n \times n)$ square matrix $\boldsymbol{A}$ is said to be *symmetric* if $\boldsymbol{A}^T = \boldsymbol{A}$, and thus $a_{ij} = a_{ji}$:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{12} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{nn} \end{bmatrix}.$$

An $(n \times n)$ square matrix $\boldsymbol{A}$ is said to be *skew-symmetric* if $\boldsymbol{A}^T = -\boldsymbol{A}$, and thus $a_{ij} = -a_{ji}$ for $i \neq j$ and $a_{ii} = 0$, leading to

$$\boldsymbol{A} = \begin{bmatrix} 0 & a_{12} & \dots & a_{1n} \\ -a_{12} & 0 & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{1n} & -a_{2n} & \dots & 0 \end{bmatrix}.$$

A *partitioned* matrix is a matrix whose elements are matrices (*blocks*) of proper dimensions:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \dots & \boldsymbol{A}_{1n} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \dots & \boldsymbol{A}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}_{m1} & \boldsymbol{A}_{m2} & \dots & \boldsymbol{A}_{mn} \end{bmatrix}.$$

[2] Subscript n is usually omitted if the dimensions are clear from the context.

A partitioned matrix may be block-triangular or block-diagonal. Special partitions of a matrix are that by columns

$$\boldsymbol{A} = [\, \boldsymbol{a}_1 \quad \boldsymbol{a}_2 \quad \dots \quad \boldsymbol{a}_n \,]$$

and that by rows

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_1^T \\ \boldsymbol{a}_2^T \\ \vdots \\ \boldsymbol{a}_m^T \end{bmatrix}.$$

Given a square matrix $\boldsymbol{A}$ of dimensions $(n \times n)$, the *algebraic complement* $\boldsymbol{A}_{(ij)}$ of element a_{ij} is the matrix of dimensions $((n-1) \times (n-1))$ which is obtained by eliminating row i and column j of matrix $\boldsymbol{A}$.

A.2 Matrix Operations

The *trace* of an $(n \times n)$ square matrix $\boldsymbol{A}$ is the sum of the elements on the diagonal:

$$\text{Tr}(\boldsymbol{A}) = \sum_{i=1}^{n} a_{ii}. \tag{A.3}$$

Two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ of the same dimensions $(m \times n)$ are equal if $a_{ij} = b_{ij}$. If $\boldsymbol{A}$ and $\boldsymbol{B}$ are two matrices of the same dimensions, their *sum* is the matrix

$$\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{B} \tag{A.4}$$

whose elements are given by $c_{ij} = a_{ij} + b_{ij}$. The following properties hold:

$$\begin{aligned} &\boldsymbol{A} + \boldsymbol{O} = \boldsymbol{A} \\ &\boldsymbol{A} + \boldsymbol{B} = \boldsymbol{B} + \boldsymbol{A} \\ &(\boldsymbol{A} + \boldsymbol{B}) + \boldsymbol{C} = \boldsymbol{A} + (\boldsymbol{B} + \boldsymbol{C}). \end{aligned}$$

Notice that two matrices of the same dimensions and partitioned in the same way can be summed formally by operating on the blocks in the same position and treating them like elements.

The *product of a scalar α by an $(m \times n)$ matrix* $\boldsymbol{A}$ is the matrix $\alpha\boldsymbol{A}$ whose elements are given by αa_{ij}. If $\boldsymbol{A}$ is an $(n \times n)$ diagonal matrix with all equal elements on the diagonal ($a_{ii} = a$), it follows that $\boldsymbol{A} = a\boldsymbol{I}_n$.

If $\boldsymbol{A}$ is a square matrix, one may write

$$\boldsymbol{A} = \boldsymbol{A}_s + \boldsymbol{A}_a \tag{A.5}$$

where

$$\boldsymbol{A}_s = \frac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^T) \tag{A.6}$$

is a symmetric matrix representing the *symmetric* part of $\boldsymbol{A}$, and

$$\boldsymbol{A}_a = \frac{1}{2}(\boldsymbol{A} - \boldsymbol{A}^T) \tag{A.7}$$

is a skew-symmetric matrix representing the *skew-symmetric* part of $\boldsymbol{A}$.

The row-by-column *product* of a matrix $\boldsymbol{A}$ of dimensions $(m \times p)$ by a matrix $\boldsymbol{B}$ of dimensions $(p \times n)$ is the matrix of dimensions $(m \times n)$

$$\boldsymbol{C} = \boldsymbol{AB} \tag{A.8}$$

whose elements are given by $c_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj}$. The following properties hold:

$$\begin{aligned}
&\boldsymbol{A} = \boldsymbol{A}\boldsymbol{I}_p = \boldsymbol{I}_m \boldsymbol{A} \\
&\boldsymbol{A}(\boldsymbol{BC}) = (\boldsymbol{AB})\boldsymbol{C} \\
&\boldsymbol{A}(\boldsymbol{B} + \boldsymbol{C}) = \boldsymbol{AB} + \boldsymbol{AC} \\
&(\boldsymbol{A} + \boldsymbol{B})\boldsymbol{C} = \boldsymbol{AC} + \boldsymbol{BC} \\
&(\boldsymbol{AB})^T = \boldsymbol{B}^T \boldsymbol{A}^T.
\end{aligned}$$

Notice that, in general, $\boldsymbol{AB} \neq \boldsymbol{BA}$, and $\boldsymbol{AB} = \boldsymbol{O}$ does not imply that $\boldsymbol{A} = \boldsymbol{O}$ or $\boldsymbol{B} = \boldsymbol{O}$; further, notice that $\boldsymbol{AC} = \boldsymbol{BC}$ does not imply that $\boldsymbol{A} = \boldsymbol{B}$.

If an $(m \times p)$ matrix $\boldsymbol{A}$ and a $(p \times n)$ matrix $\boldsymbol{B}$ are partitioned in such a way that the number of blocks for each row of $\boldsymbol{A}$ is equal to the number of blocks for each column of $\boldsymbol{B}$, and the blocks $\boldsymbol{A}_{ik}$ and $\boldsymbol{B}_{kj}$ have dimensions compatible with product, the matrix product $\boldsymbol{AB}$ can be formally obtained by operating by rows and columns on the blocks of proper position and treating them like elements.

For an $(n \times n)$ *square* matrix $\boldsymbol{A}$, the *determinant* of $\boldsymbol{A}$ is the scalar given by the following expression, which holds $\forall i = 1, \ldots, n$:

$$\det(\boldsymbol{A}) = \sum_{j=1}^{n} a_{ij} (-1)^{i+j} \det\big(\boldsymbol{A}_{(ij)}\big). \tag{A.9}$$

The determinant can be computed according to any row i as in (A.9); the same result is obtained by computing it according to any column j. If $n = 1$, then $\det(a_{11}) = a_{11}$. The following property holds:

$$\det(\boldsymbol{A}) = \det(\boldsymbol{A}^T).$$

Moreover, interchanging two generic columns p and q of a matrix $\boldsymbol{A}$ yields

$$\det\big([\, \boldsymbol{a}_1 \ldots \boldsymbol{a}_p \ldots \boldsymbol{a}_q \ldots \boldsymbol{a}_n \,]\big) = -\det\big([\, \boldsymbol{a}_1 \ldots \boldsymbol{a}_q \ldots \boldsymbol{a}_p \ldots \boldsymbol{a}_n \,]\big).$$

As a consequence, if a matrix has two equal columns (rows), then its determinant is null. Also, it is $\det(\alpha \boldsymbol{A}) = \alpha^n \det(\boldsymbol{A})$.

Given an $(m \times n)$ matrix $\boldsymbol{A}$, the determinant of the square block obtained by selecting an equal number k of rows and columns is said to be k-order *minor* of matrix $\boldsymbol{A}$. The minors obtained by taking the *first* k rows and columns of $\boldsymbol{A}$ are said to be *principal* minors.

If $\boldsymbol{A}$ and $\boldsymbol{B}$ are square matrices, then

$$\det(\boldsymbol{AB}) = \det(\boldsymbol{A}) \det(\boldsymbol{B}). \tag{A.10}$$

If $\boldsymbol{A}$ is an $(n \times n)$ triangular matrix (in particular diagonal), then

$$\det(\boldsymbol{A}) = \prod_{i=1}^{n} a_{ii}.$$

More generally, if $\boldsymbol{A}$ is block-triangular with m blocks $\boldsymbol{A}_{ii}$ on the diagonal, then

$$\det(\boldsymbol{A}) = \prod_{i=1}^{m} \det(\boldsymbol{A}_{ii}).$$

A square matrix $\boldsymbol{A}$ is said to be *singular* when $\det(\boldsymbol{A}) = 0$.

The *rank* $\varrho(\boldsymbol{A})$ of a matrix $\boldsymbol{A}$ of dimensions $(m \times n)$ is the maximum integer r so that at least a non-null minor of order r exists. The following properties hold:

$$\begin{aligned}
&\varrho(\boldsymbol{A}) \leq \min\{m, n\} \\
&\varrho(\boldsymbol{A}) = \varrho(\boldsymbol{A}^T) \\
&\varrho(\boldsymbol{A}^T \boldsymbol{A}) = \varrho(\boldsymbol{A}) \\
&\varrho(\boldsymbol{AB}) \leq \min\{\varrho(\boldsymbol{A}), \varrho(\boldsymbol{B})\}.
\end{aligned}$$

A matrix so that $\varrho(\boldsymbol{A}) = \min\{m, n\}$ is said to be *full-rank*.

The *adjoint* of a square matrix $\boldsymbol{A}$ is the matrix

$$\operatorname{Adj} \boldsymbol{A} = [(-1)^{i+j} \det(\boldsymbol{A}_{(ij)})]^T_{\substack{i=1,\ldots,n \\ j=1,\ldots,n}}. \tag{A.11}$$

An $(n \times n)$ square matrix $\boldsymbol{A}$ is said to be *invertible* if a matrix $\boldsymbol{A}^{-1}$ exists, termed *inverse* of $\boldsymbol{A}$, so that

$$\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I}_n.$$

Since $\varrho(\boldsymbol{I}_n) = n$, an $(n \times n)$ square matrix $\boldsymbol{A}$ is invertible if and only if $\varrho(\boldsymbol{A}) = n$, i.e., $\det(\boldsymbol{A}) \neq 0$ (nonsingular matrix). The inverse of $\boldsymbol{A}$ can be computed as

$$\boldsymbol{A}^{-1} = \frac{1}{\det(\boldsymbol{A})} \operatorname{Adj} \boldsymbol{A}. \tag{A.12}$$

The following properties hold:

$$\begin{aligned}
&(\boldsymbol{A}^{-1})^{-1} = \boldsymbol{A} \\
&(\boldsymbol{A}^T)^{-1} = (\boldsymbol{A}^{-1})^T.
\end{aligned}$$

If the inverse of a square matrix is equal to its transpose

$$\boldsymbol{A}^T = \boldsymbol{A}^{-1} \tag{A.13}$$

then the matrix is said to be *orthogonal*; in this case it is

$$\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}. \tag{A.14}$$

A square matrix $\boldsymbol{A}$ is said *idempotent* if

$$\boldsymbol{A}\boldsymbol{A} = \boldsymbol{A}. \tag{A.15}$$

If $\boldsymbol{A}$ and $\boldsymbol{B}$ are invertible square matrices of the same dimensions, then

$$(\boldsymbol{A}\boldsymbol{B})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1}. \tag{A.16}$$

Given n square matrices $\boldsymbol{A}_{ii}$ all invertible, the following expression holds:

$$\big(\mathrm{diag}\{\boldsymbol{A}_{11},\ldots,\boldsymbol{A}_{nn}\}\big)^{-1} = \mathrm{diag}\{\boldsymbol{A}_{11}^{-1},\ldots,\boldsymbol{A}_{nn}^{-1}\}.$$

where $\mathrm{diag}\{\boldsymbol{A}_{11},\ldots,\boldsymbol{A}_{nn}\}$ denotes the block-diagonal matrix.

If $\boldsymbol{A}$ and $\boldsymbol{C}$ are invertible square matrices of proper dimensions, the following expression holds:

$$(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{C}\boldsymbol{D})^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{B}+\boldsymbol{C}^{-1})^{-1}\boldsymbol{D}\boldsymbol{A}^{-1},$$

where the matrix $\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{B}+\boldsymbol{C}^{-1}$ must be invertible.

If a block-partitioned matrix is invertible, then its inverse is given by the general expression

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{D} \\ \boldsymbol{C} & \boldsymbol{B} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{A}^{-1}+\boldsymbol{E}\boldsymbol{\Delta}^{-1}\boldsymbol{F} & -\boldsymbol{E}\boldsymbol{\Delta}^{-1} \\ -\boldsymbol{\Delta}^{-1}\boldsymbol{F} & \boldsymbol{\Delta}^{-1} \end{bmatrix} \tag{A.17}$$

where $\boldsymbol{\Delta} = \boldsymbol{B}-\boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{D}$, $\boldsymbol{E} = \boldsymbol{A}^{-1}\boldsymbol{D}$ and $\boldsymbol{F} = \boldsymbol{C}\boldsymbol{A}^{-1}$, under the assumption that the inverses of matrices $\boldsymbol{A}$ and $\boldsymbol{\Delta}$ exist. In the case of a block-triangular matrix, invertibility of the matrix requires invertibility of the blocks on the diagonal. The following expressions hold:

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{O} \\ \boldsymbol{C} & \boldsymbol{B} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{A}^{-1} & \boldsymbol{O} \\ -\boldsymbol{B}^{-1}\boldsymbol{C}\boldsymbol{A}^{-1} & \boldsymbol{B}^{-1} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{D} \\ \boldsymbol{O} & \boldsymbol{B} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{A}^{-1} & -\boldsymbol{A}^{-1}\boldsymbol{D}\boldsymbol{B}^{-1} \\ \boldsymbol{O} & \boldsymbol{B}^{-1} \end{bmatrix}.$$

The *derivative* of an $(m \times n)$ matrix $\boldsymbol{A}(t)$, whose elements $a_{ij}(t)$ are differentiable functions, is the matrix

$$\dot{\boldsymbol{A}}(t) = \frac{d}{dt}\boldsymbol{A}(t) = \left[\frac{d}{dt}a_{ij}(t)\right]_{\substack{i=1,\ldots,m \\ j=1,\ldots,n}}. \tag{A.18}$$

If an $(n \times n)$ square matrix $\boldsymbol{A}(t)$ is so that $\varrho(\boldsymbol{A}(t)) = n\ \forall t$ and its elements $a_{ij}(t)$ are differentiable functions, then the derivative of the *inverse* of $\boldsymbol{A}(t)$ is given by

$$\frac{d}{dt}\boldsymbol{A}^{-1}(t) = -\boldsymbol{A}^{-1}(t)\dot{\boldsymbol{A}}(t)\boldsymbol{A}^{-1}(t). \tag{A.19}$$

Given a scalar function $f(\boldsymbol{x})$, endowed with partial derivatives with respect to the elements x_i of the $(n \times 1)$ vector $\boldsymbol{x}$, the *gradient* of function f with respect to vector $\boldsymbol{x}$ is the $(n \times 1)$ column vector

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \left(\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right)^T = \begin{bmatrix} \dfrac{\partial f(\boldsymbol{x})}{\partial x_1} & \dfrac{\partial f(\boldsymbol{x})}{\partial x_2} & \ldots & \dfrac{\partial f(\boldsymbol{x})}{\partial x_n} \end{bmatrix}^T. \tag{A.20}$$

Further, if $\boldsymbol{x}(t)$ is a differentiable function with respect to t, then

$$\dot{f}(\boldsymbol{x}) = \frac{d}{dt}f(\boldsymbol{x}(t)) = \frac{\partial f}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} = \nabla_{\boldsymbol{x}}^T f(\boldsymbol{x})\dot{\boldsymbol{x}}. \tag{A.21}$$

Given a vector function $\boldsymbol{g}(\boldsymbol{x})$ of dimensions $(m \times 1)$, whose elements g_i are differentiable with respect to the vector $\boldsymbol{x}$ of dimensions $(n \times 1)$, the Jacobian matrix (or simply *Jacobian*) of the function is defined as the $(m \times n)$ matrix

$$\boldsymbol{J}_g(\boldsymbol{x}) = \frac{\partial \boldsymbol{g}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \begin{bmatrix} \dfrac{\partial g_1(\boldsymbol{x})}{\partial \boldsymbol{x}} \\ \dfrac{\partial g_2(\boldsymbol{x})}{\partial \boldsymbol{x}} \\ \vdots \\ \dfrac{\partial g_m(\boldsymbol{x})}{\partial \boldsymbol{x}} \end{bmatrix}. \tag{A.22}$$

If $\boldsymbol{x}(t)$ is a differentiable function with respect to t, then

$$\dot{\boldsymbol{g}}(\boldsymbol{x}) = \frac{d}{dt}\boldsymbol{g}(\boldsymbol{x}(t)) = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} = \boldsymbol{J}_g(\boldsymbol{x})\dot{\boldsymbol{x}}. \tag{A.23}$$

A.3 Vector Operations

Given n vectors $\boldsymbol{x}_i$ of dimensions $(m \times 1)$, they are said to be *linearly independent* if the expression

$$k_1\boldsymbol{x}_1 + k_2\boldsymbol{x}_2 + \ldots + k_n\boldsymbol{x}_n = \boldsymbol{0}$$

holds true only when all the constants k_i vanish. A necessary and sufficient condition for the vectors $\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots, \boldsymbol{x}_n$ to be linearly independent is that the matrix

$$\boldsymbol{A} = [\,\boldsymbol{x}_1 \quad \boldsymbol{x}_2 \quad \ldots \quad \boldsymbol{x}_n\,]$$

has rank n; this implies that a necessary condition for linear independence is that $n \leq m$. If instead $\varrho(\boldsymbol{A}) = r < n$, then only r vectors are linearly independent and the remaining $n - r$ vectors can be expressed as a linear combination of the previous ones.

A system of vectors $\mathcal{X}$ is a *vector space* on the field of real numbers $\mathbb{R}$ if the operations of *sum of two vectors* of $\mathcal{X}$ and *product of a scalar by a vector* of $\mathcal{X}$ have values in $\mathcal{X}$ and the following properties hold:

$$\begin{aligned}
&\boldsymbol{x} + \boldsymbol{y} = \boldsymbol{y} + \boldsymbol{x} \qquad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{X} \\
&(\boldsymbol{x} + \boldsymbol{y}) + \boldsymbol{z} = \boldsymbol{x} + (\boldsymbol{y} + \boldsymbol{z}) \qquad \forall \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathcal{X} \\
&\exists \boldsymbol{0} \in \mathcal{X} : \boldsymbol{x} + \boldsymbol{0} = \boldsymbol{x} \qquad \forall \boldsymbol{x} \in \mathcal{X} \\
&\forall \boldsymbol{x} \in \mathcal{X}, \;\; \exists (-\boldsymbol{x}) \in \mathcal{X} : \boldsymbol{x} + (-\boldsymbol{x}) = \boldsymbol{0} \\
&1\boldsymbol{x} = \boldsymbol{x} \qquad \forall \boldsymbol{x} \in \mathcal{X} \\
&\alpha(\beta \boldsymbol{x}) = (\alpha\beta)\boldsymbol{x} \qquad \forall \alpha, \beta \in \mathbb{R} \quad \forall \boldsymbol{x} \in \mathcal{X} \\
&(\alpha + \beta)\boldsymbol{x} = \alpha\boldsymbol{x} + \beta\boldsymbol{x} \qquad \forall \alpha, \beta \in \mathbb{R} \quad \forall \boldsymbol{x} \in \mathcal{X} \\
&\alpha(\boldsymbol{x} + \boldsymbol{y}) = \alpha\boldsymbol{x} + \alpha\boldsymbol{y} \qquad \forall \alpha \in \mathbb{R} \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}.
\end{aligned}$$

The *dimension* of the space $\dim(\mathcal{X})$ is the maximum number of linearly independent vectors $\boldsymbol{x}$ in the space. A set $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ of linearly independent vectors is a *basis* of vector space $\mathcal{X}$, and each vector $\boldsymbol{y}$ in the space can be uniquely expressed as a linear combination of vectors from the basis

$$\boldsymbol{y} = c_1\boldsymbol{x}_1 + c_2\boldsymbol{x}_2 + \ldots + c_n\boldsymbol{x}_n, \tag{A.24}$$

where the constants $c_1, c_2, \ldots, c_n$ are said to be the *components* of the vector $\boldsymbol{y}$ in the basis $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$.

A subset $\mathcal{Y}$ of a vector space $\mathcal{X}$ is a *subspace* $\mathcal{Y} \subseteq \mathcal{X}$ if it is a vector space with the operations of vector sum and product of a scalar by a vector, i.e.,

$$\alpha\boldsymbol{x} + \beta\boldsymbol{y} \in \mathcal{Y} \qquad \forall \alpha, \beta \in \mathbb{R} \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{Y}.$$

According to a geometric interpretation, a subspace is a *hyperplane* passing by the origin (null element) of $\mathcal{X}$.

The *scalar product* $< \boldsymbol{x}, \boldsymbol{y} >$ of two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ of dimensions $(m \times 1)$ is the scalar that is obtained by summing the products of the respective components in a given basis

$$< \boldsymbol{x}, \boldsymbol{y} >= x_1y_1 + x_2y_2 + \ldots + x_my_m = \boldsymbol{x}^T\boldsymbol{y} = \boldsymbol{y}^T\boldsymbol{x}. \tag{A.25}$$

Two vectors are said to be *orthogonal* when their scalar product is null:

$$\boldsymbol{x}^T\boldsymbol{y} = 0. \tag{A.26}$$

The *norm* of a vector can be defined as

$$\|\boldsymbol{x}\| = \sqrt{\boldsymbol{x}^T\boldsymbol{x}}. \tag{A.27}$$

It is possible to show that both the *triangle inequality*

$$\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\| \tag{A.28}$$

and the *Schwarz inequality*

$$|\boldsymbol{x}^T\boldsymbol{y}| \leq \|\boldsymbol{x}\| \, \|\boldsymbol{y}\|. \tag{A.29}$$

hold. A *unit vector* $\hat{\boldsymbol{x}}$ is a vector whose *norm* is unity, i.e., $\hat{\boldsymbol{x}}^T\hat{\boldsymbol{x}} = 1$. Given a vector $\boldsymbol{x}$, its unit vector is obtained by dividing each component by its norm:

$$\widehat{\boldsymbol{x}} = \frac{1}{\|\boldsymbol{x}\|}\boldsymbol{x}. \tag{A.30}$$

A typical example of vector space is the *Euclidean space* whose dimension is 3; in this case a basis is constituted by the unit vectors of a coordinate frame.

The *vector product* of two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in the Euclidean space is the vector

$$\boldsymbol{x} \times \boldsymbol{y} = \begin{bmatrix} x_2y_3 - x_3y_2 \\ x_3y_1 - x_1y_3 \\ x_1y_2 - x_2y_1 \end{bmatrix}. \tag{A.31}$$

The following properties hold:

$$\begin{aligned}
&\boldsymbol{x} \times \boldsymbol{x} = \boldsymbol{0} \\
&\boldsymbol{x} \times \boldsymbol{y} = -\boldsymbol{y} \times \boldsymbol{x} \\
&\boldsymbol{x} \times (\boldsymbol{y} + \boldsymbol{z}) = \boldsymbol{x} \times \boldsymbol{y} + \boldsymbol{x} \times \boldsymbol{z}.
\end{aligned}$$

The vector product of two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ can be expressed also as the product of a matrix operator $\boldsymbol{S}(\boldsymbol{x})$ by the vector $\boldsymbol{y}$. In fact, by introducing the *skew-symmetric* matrix

$$\boldsymbol{S}(\boldsymbol{x}) = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \tag{A.32}$$

obtained with the components of vector $\boldsymbol{x}$, the vector product $\boldsymbol{x} \times \boldsymbol{y}$ is given by

$$\boldsymbol{x} \times \boldsymbol{y} = \boldsymbol{S}(\boldsymbol{x})\boldsymbol{y} = -\boldsymbol{S}(\boldsymbol{y})\boldsymbol{x} \tag{A.33}$$

as can be easily verified. Moreover, the following properties hold:

$$\begin{aligned}
&\boldsymbol{S}(\boldsymbol{x})\boldsymbol{x} = \boldsymbol{S}^T(\boldsymbol{x})\boldsymbol{x} = \boldsymbol{0} \\
&\boldsymbol{S}(\alpha\boldsymbol{x} + \beta\boldsymbol{y}) = \alpha\boldsymbol{S}(\boldsymbol{x}) + \beta\boldsymbol{S}(\boldsymbol{y}).
\end{aligned}$$

Given three vectors $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ in the Euclidean space, the following expressions hold for the *scalar triple products*:

$$\boldsymbol{x}^T(\boldsymbol{y} \times \boldsymbol{z}) = \boldsymbol{y}^T(\boldsymbol{z} \times \boldsymbol{x}) = \boldsymbol{z}^T(\boldsymbol{x} \times \boldsymbol{y}). \tag{A.34}$$

If any two vectors of three are equal, then the scalar triple product is null; e.g.,

$$\boldsymbol{x}^T(\boldsymbol{x} \times \boldsymbol{y}) = \boldsymbol{0}.$$

A.4 Linear Transformation

Consider a vector space $\mathcal{X}$ of dimension n and a vector space $\mathcal{Y}$ of dimension m with $m \leq n$. The *linear transformation* (or linear map) between the vectors $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{y} \in \mathcal{Y}$ can be defined as

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \tag{A.35}$$

in terms of the matrix $\boldsymbol{A}$ of dimensions $(m \times n)$. The *range space* (or simply range) of the transformation is the subspace

$$\mathcal{R}(\boldsymbol{A}) = \{\boldsymbol{y} : \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x},\ \boldsymbol{x} \in \mathcal{X}\} \subseteq \mathcal{Y}, \tag{A.36}$$

which is the subspace generated by the linearly independent columns of matrix $\boldsymbol{A}$ taken as a basis of $\mathcal{Y}$. It is easy to recognize that

$$\varrho(\boldsymbol{A}) = \dim(\mathcal{R}(\boldsymbol{A})). \tag{A.37}$$

On the other hand, the *null space* (or simply null) of the transformation is the subspace

$$\mathcal{N}(\boldsymbol{A}) = \{\boldsymbol{x} : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0},\ \boldsymbol{x} \in \mathcal{X}\} \subseteq \mathcal{X}. \tag{A.38}$$

Given a matrix $\boldsymbol{A}$ of dimensions $(m \times n)$, the notable result holds:

$$\varrho(\boldsymbol{A}) + \dim(\mathcal{N}(\boldsymbol{A})) = n. \tag{A.39}$$

Therefore, if $\varrho(\boldsymbol{A}) = r \leq \min\{m, n\}$, then $\dim(\mathcal{R}(\boldsymbol{A})) = r$ and $\dim(\mathcal{N}(\boldsymbol{A})) = n - r$. It follows that if $m < n$, then $\mathcal{N}(\boldsymbol{A}) \neq \emptyset$ independently of the rank of $\boldsymbol{A}$; if $m = n$, then $\mathcal{N}(\boldsymbol{A}) \neq \emptyset$ only in the case of $\varrho(\boldsymbol{A}) = r < m$.

If $\boldsymbol{x} \in \mathcal{N}(\boldsymbol{A})$ and $\boldsymbol{y} \in \mathcal{R}(\boldsymbol{A}^T)$, then $\boldsymbol{y}^T\boldsymbol{x} = 0$, i.e., the vectors in the null space of $\boldsymbol{A}$ are orthogonal to each vector in the range space of the transpose of $\boldsymbol{A}$. It can be shown that the set of vectors orthogonal to each vector of the range space of $\boldsymbol{A}^T$ coincides with the null space of $\boldsymbol{A}$, whereas the set of vectors orthogonal to each vector in the null space of $\boldsymbol{A}^T$ coincides with the range space of $\boldsymbol{A}$. In symbols:

$$\mathcal{N}(\boldsymbol{A}) \equiv \mathcal{R}^{\perp}(\boldsymbol{A}^T) \qquad \mathcal{R}(\boldsymbol{A}) \equiv \mathcal{N}^{\perp}(\boldsymbol{A}^T) \tag{A.40}$$

where $\perp$ denotes the *orthogonal complement* of a subspace.

If the matrix $\boldsymbol{A}$ in (A.35) is square and idempotent, the matrix represents the *projection* of space $\mathcal{X}$ into a subspace.

A linear transformation allows the definition of the *norm* of a matrix $\boldsymbol{A}$ induced by the norm defined for a vector $\boldsymbol{x}$ as follows. In view of the property

$$\|\boldsymbol{A}\boldsymbol{x}\| \leq \|\boldsymbol{A}\|\ \|\boldsymbol{x}\|, \tag{A.41}$$

the norm of $\boldsymbol{A}$ can be defined as

$$\|\boldsymbol{A}\| = \sup_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{\|\boldsymbol{A}\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \tag{A.42}$$

which can also be computed as

$$\max_{\|\boldsymbol{x}\|=1} \|\boldsymbol{A}\boldsymbol{x}\|.$$

A direct consequence of (A.41) is the property

$$\|\boldsymbol{A}\boldsymbol{B}\| \leq \|\boldsymbol{A}\|\ \|\boldsymbol{B}\|. \tag{A.43}$$

A different norm of a matrix is the *Frobenius norm* defined as

$$\|\boldsymbol{A}\|_F = \Big(\mathrm{Tr}(\boldsymbol{A}^T\boldsymbol{A})\Big)^{1/2} \tag{A.44}$$

A.5 Eigenvalues and Eigenvectors

Consider the linear transformation on a vector $\boldsymbol{u}$ established by an $(n \times n)$ square matrix $\boldsymbol{A}$. If the vector resulting from the transformation has the same direction of $\boldsymbol{u}$ (with $\boldsymbol{u} \neq \boldsymbol{0}$), then

$$\boldsymbol{A}\boldsymbol{u} = \lambda\boldsymbol{u}. \tag{A.45}$$

The equation in (A.45) can be rewritten in matrix form as

$$(\lambda\boldsymbol{I} - \boldsymbol{A})\boldsymbol{u} = \boldsymbol{0}. \tag{A.46}$$

For the homogeneous system of equations in (A.46) to have a solution different from the trivial one $\boldsymbol{u} = \boldsymbol{0}$, it must be

$$\det(\lambda\boldsymbol{I} - \boldsymbol{A}) = 0 \tag{A.47}$$

which is termed a *characteristic equation*. Its solutions $\lambda_1, \ldots, \lambda_n$ are the *eigenvalues* of matrix $\boldsymbol{A}$; they coincide with the eigenvalues of matrix $\boldsymbol{A}^T$. On the assumption of distinct eigenvalues, the n vectors $\boldsymbol{u}_i$ satisfying the equation

$$(\lambda_i\boldsymbol{I} - \boldsymbol{A})\boldsymbol{u}_i = \boldsymbol{0} \qquad i = 1, \ldots, n \tag{A.48}$$

are said to be the *eigenvectors* associated with the eigenvalues λ_i.

The matrix $\boldsymbol{U}$ formed by the column vectors $\boldsymbol{u}_i$ is invertible and constitutes a basis in the space of dimension n. Further, the *similarity transformation* established by $\boldsymbol{U}$

$$\boldsymbol{\Lambda} = \boldsymbol{U}^{-1}\boldsymbol{A}\boldsymbol{U} \tag{A.49}$$

is so that $\boldsymbol{\Lambda} = \mathrm{diag}\{\lambda_1, \ldots, \lambda_n\}$. It follows that $\det(\boldsymbol{A}) = \prod_{i=1}^{n} \lambda_i$.

If the matrix $\boldsymbol{A}$ is *symmetric*, its eigenvalues are real and $\boldsymbol{\Lambda}$ can be written as

$$\boldsymbol{\Lambda} = \boldsymbol{U}^T\boldsymbol{A}\boldsymbol{U}; \tag{A.50}$$

hence, the eigenvector matrix $\boldsymbol{U}$ is orthogonal.

A.6 Bilinear Forms and Quadratic Forms

A *bilinear form* in the variables x_i and y_j is the scalar

$$B = \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}x_i y_j$$

which can be written in matrix form

$$B(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{y} = \boldsymbol{y}^T\boldsymbol{A}^T\boldsymbol{x} \tag{A.51}$$

where $\boldsymbol{x} = [\,x_1 \quad x_2 \quad \dots \quad x_m\,]^T$, $\boldsymbol{y} = [\,y_1 \quad y_2 \quad \dots \quad y_n\,]^T$, and $\boldsymbol{A}$ is the $(m \times n)$ matrix of the coefficients a_{ij} representing the core of the form.

A special case of bilinear form is the *quadratic form*

$$Q(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \tag{A.52}$$

where $\boldsymbol{A}$ is an $(n \times n)$ square matrix. Hence, for computation of (A.52), the matrix $\boldsymbol{A}$ can be replaced with its symmetric part $\boldsymbol{A}_s$ given by (A.6). It follows that if $\boldsymbol{A}$ is a *skew-symmetric* matrix, then

$$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0} \qquad \forall \boldsymbol{x}.$$

The quadratic form (A.52) is said to be *positive definite* if

$$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} > 0 \quad \forall \boldsymbol{x} \neq \boldsymbol{0} \qquad \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = 0 \quad \boldsymbol{x} = \boldsymbol{0}. \tag{A.53}$$

The matrix $\boldsymbol{A}$ core of the form is also said to be *positive definite*. Analogously, a quadratic form is said to be *negative definite* if it can be written as $-Q(\boldsymbol{x}) = -\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}$ where $Q(\boldsymbol{x})$ is positive definite.

A necessary condition for a square matrix to be positive definite is that its elements on the diagonal are strictly positive. Further, in view of (A.50), the eigenvalues of a positive definite matrix are all positive. If the eigenvalues are not known, a necessary and sufficient condition for a symmetric matrix to be positive definite is that its principal minors are strictly positive (*Sylvester criterion*). It follows that a positive definite matrix is full-rank and thus it is always invertible.

A symmetric positive definite matrix $\boldsymbol{A}$ can always be decomposed as

$$\boldsymbol{A} = \boldsymbol{U}^T\boldsymbol{\Lambda}\boldsymbol{U} \tag{A.54}$$

where $\boldsymbol{U}$ is an orthogonal matrix of eigenvectors ($\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$) and $\boldsymbol{\Lambda}$ is the diagonal matrix of the eigenvalues of $\boldsymbol{A}$.

Let $\lambda_{\min}(\boldsymbol{A})$ and $\lambda_{\max}(\boldsymbol{A})$ respectively denote the smallest and largest eigenvalues of a positive definite matrix $\boldsymbol{A}$ ($\lambda_{\min}, \lambda_{\max} > 0$). Then, the quadratic form in (A.52) satisfies the following inequality:

$$\lambda_{\min}(\boldsymbol{A})\|\boldsymbol{x}\|^2 \leq \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \leq \lambda_{\max}(\boldsymbol{A})\|\boldsymbol{x}\|^2. \tag{A.55}$$

An $(n \times n)$ square matrix $\boldsymbol{A}$ is said to be *positive semi-definite* if

$$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \geq 0 \qquad \forall \boldsymbol{x}. \tag{A.56}$$

This definition implies that $\varrho(\boldsymbol{A}) = r < n$, and thus r eigenvalues of $\boldsymbol{A}$ are positive and $n - r$ are null. Therefore, a positive semi-definite matrix $\boldsymbol{A}$ has a null space of finite dimension, and specifically the form vanishes when $\boldsymbol{x} \in \mathcal{N}(\boldsymbol{A})$. A typical example of a positive semi-definite matrix is the matrix $\boldsymbol{A} = \boldsymbol{H}^T\boldsymbol{H}$ where $\boldsymbol{H}$ is an $(m \times n)$ matrix with $m < n$. In an analogous way, a *negative semi-definite* matrix can be defined.

Given the *bilinear form* in (A.51), the *gradient* of the form with respect to $\boldsymbol{x}$ is given by

$$\nabla_{\boldsymbol{x}} B(\boldsymbol{x},\boldsymbol{y}) = \left(\frac{\partial B(\boldsymbol{x},\boldsymbol{y})}{\partial \boldsymbol{x}}\right)^T = \boldsymbol{A}\boldsymbol{y}, \tag{A.57}$$

whereas the gradient of B with respect to $\boldsymbol{y}$ is given by

$$\nabla_{\boldsymbol{y}} B(\boldsymbol{x},\boldsymbol{y}) = \left(\frac{\partial B(\boldsymbol{x},\boldsymbol{y})}{\partial \boldsymbol{y}}\right)^T = \boldsymbol{A}^T\boldsymbol{x}. \tag{A.58}$$

Given the *quadratic form* in (A.52) with $\boldsymbol{A}$ *symmetric*, the *gradient* of the form with respect to $\boldsymbol{x}$ is given by

$$\nabla_{\boldsymbol{x}} Q(\boldsymbol{x}) = \left(\frac{\partial Q(\boldsymbol{x})}{\partial \boldsymbol{x}}\right)^T = 2\boldsymbol{A}\boldsymbol{x}. \tag{A.59}$$

Further, if $\boldsymbol{x}$ and $\boldsymbol{A}$ are differentiable functions of t, then

$$\dot{Q}(x) = \frac{d}{dt}Q(\boldsymbol{x}(t)) = 2\boldsymbol{x}^T\boldsymbol{A}\dot{\boldsymbol{x}} + \boldsymbol{x}^T\dot{\boldsymbol{A}}\boldsymbol{x}; \tag{A.60}$$

if $\boldsymbol{A}$ is constant, then the second term obviously vanishes.

A.7 Pseudo-inverse

The inverse of a matrix can be defined only when the matrix is square and nonsingular. The inverse operation can be extended to the case of non-square matrices. Consider a matrix $\boldsymbol{A}$ of dimensions $(m \times n)$ with $\varrho(\boldsymbol{A}) = \min\{m, n\}$

If $m < n$, a *right inverse* of $\boldsymbol{A}$ can be defined as the matrix $\boldsymbol{A}_r$ of dimensions $(n \times m)$ so that

$$\boldsymbol{A}\boldsymbol{A}_r = \boldsymbol{I}_m.$$

If instead $m > n$, a *left inverse* of $\boldsymbol{A}$ can be defined as the matrix $\boldsymbol{A}_l$ of dimensions $(n \times m)$ so that

$$\boldsymbol{A}_l\boldsymbol{A} = \boldsymbol{I}_n.$$

If $\boldsymbol{A}$ has more columns than rows ($m < n$) and has rank m, a special right inverse is the matrix

$$\boldsymbol{A}_r^\dagger = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1} \tag{A.61}$$

which is termed *right pseudo-inverse*, since $\boldsymbol{A}\boldsymbol{A}_r^\dagger = \boldsymbol{I}_m$. If $\boldsymbol{W}_r$ is an $(n \times n)$ *positive definite* matrix, a *weighted* right pseudo-inverse is given by

$$\boldsymbol{A}_r^\dagger = \boldsymbol{W}_r^{-1}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{W}_r^{-1}\boldsymbol{A}^T)^{-1}. \tag{A.62}$$

If $\boldsymbol{A}$ has more rows than columns ($m > n$) and has rank n, a special left inverse is the matrix

$$\boldsymbol{A}_l^\dagger = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T \tag{A.63}$$

which is termed *left pseudo-inverse*, since $\boldsymbol{A}_l^\dagger\boldsymbol{A} = \boldsymbol{I}_n$.[3] If $\boldsymbol{W}_l$ is an $(m \times m)$ *positive definite* matrix, a *weighted* left pseudo-inverse is given by

$$\boldsymbol{A}_l^\dagger = (\boldsymbol{A}^T\boldsymbol{W}_l\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{W}_l. \tag{A.64}$$

The pseudo-inverse is very useful to invert a linear transformation $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ with $\boldsymbol{A}$ a full-rank matrix. If $\boldsymbol{A}$ is a square nonsingular matrix, then obviously $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{y}$ and then $\boldsymbol{A}_l^\dagger = \boldsymbol{A}_r^\dagger = \boldsymbol{A}^{-1}$.

If $\boldsymbol{A}$ has more columns than rows ($m < n$) and has rank m, then the solution $\boldsymbol{x}$ for a given $\boldsymbol{y}$ is not unique; it can be shown that the expression

$$\boldsymbol{x} = \boldsymbol{A}^\dagger\boldsymbol{y} + (\boldsymbol{I} - \boldsymbol{A}^\dagger\boldsymbol{A})\boldsymbol{k}, \tag{A.65}$$

with $\boldsymbol{k}$ an arbitrary $(n \times 1)$ vector and $\boldsymbol{A}^\dagger$ as in (A.61), is a solution to the system of linear equations established by (A.35). The term $\boldsymbol{A}^\dagger\boldsymbol{y} \in \mathcal{N}^\perp(\boldsymbol{A}) \equiv \mathcal{R}(\boldsymbol{A}^T)$ minimizes the norm of the solution $\|\boldsymbol{x}\|$. The term $(\boldsymbol{I} - \boldsymbol{A}^\dagger\boldsymbol{A})\boldsymbol{k}$ is the projection of $\boldsymbol{k}$ in $\mathcal{N}(\boldsymbol{A})$ and is termed *homogeneous solution*; as $\boldsymbol{k}$ varies, all the solutions to the homogeneous equation system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$ associated with (A.35) are generated.

On the other hand, if $\boldsymbol{A}$ has more rows than columns ($m > n$), the equation in (A.35) has no solution; it can be shown that an *approximate* solution is given by

$$\boldsymbol{x} = \boldsymbol{A}^\dagger\boldsymbol{y} \tag{A.66}$$

where $\boldsymbol{A}^\dagger$ as in (A.63) minimizes $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|$. If instead $\boldsymbol{y} \in \mathcal{R}(\boldsymbol{A})$, then (A.66) is a real solution.

Notice that the use of the weighted (left or right) pseudo-inverses in the solution to the linear equation systems leads to analogous results where the minimized norms are weighted according to the metrics defined by matrices $\boldsymbol{W}_r$ and $\boldsymbol{W}_l$, respectively.

The results of this section can be easily extended to the case of (square or nonsquare) matrices $\boldsymbol{A}$ not having full-rank. In particular, the expression (A.66) (with the pseudo-inverse computed by means of the singular value decomposition of $\boldsymbol{A}$) gives the minimum-norm vector among all those minimizing $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|$.

[3] Subscripts l and r are usually omitted whenever the use of a left or right pseudo-inverse is clear from the context.

A.8 Singular Value Decomposition

For a nonsquare matrix it is not possible to define eigenvalues. An extension of the eigenvalue concept can be obtained by singular values. Given a matrix $\boldsymbol{A}$ of dimensions $(m \times n)$, the matrix $\boldsymbol{A}^T\boldsymbol{A}$ has n nonnegative eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0$ (ordered from the largest to the smallest) which can be expressed in the form

$$\lambda_i = \sigma_i^2 \qquad \sigma_i \geq 0.$$

The scalars $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$ are said to be the *singular values* of matrix $\boldsymbol{A}$. The *singular value decomposition* (SVD) of matrix $\boldsymbol{A}$ is given by

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{A.67}$$

where $\boldsymbol{U}$ is an $(m \times m)$ orthogonal matrix

$$\boldsymbol{U} = [\,\boldsymbol{u}_1 \quad \boldsymbol{u}_2 \quad \ldots \quad \boldsymbol{u}_m\,], \tag{A.68}$$

$\boldsymbol{V}$ is an $(n \times n)$ orthogonal matrix

$$\boldsymbol{V} = [\,\boldsymbol{v}_1 \quad \boldsymbol{v}_2 \quad \ldots \quad \boldsymbol{v}_n\,] \tag{A.69}$$

and $\boldsymbol{\Sigma}$ is an $(m \times n)$ matrix

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{D} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \qquad \boldsymbol{D} = \mathrm{diag}\{\sigma_1, \sigma_2, \ldots, \sigma_r\} \tag{A.70}$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$. The number of non-null singular values is equal to the rank r of matrix $\boldsymbol{A}$.

The columns of $\boldsymbol{U}$ are the eigenvectors of the matrix $\boldsymbol{A}\boldsymbol{A}^T$, whereas the columns of $\boldsymbol{V}$ are the eigenvectors of the matrix $\boldsymbol{A}^T\boldsymbol{A}$. In view of the partitions of $\boldsymbol{U}$ and $\boldsymbol{V}$ in (A.68), (A.69), it is $\boldsymbol{A}\boldsymbol{v}_i = \sigma_i\boldsymbol{u}_i$, for $i = 1, \ldots, r$ and $\boldsymbol{A}\boldsymbol{v}_i = \boldsymbol{0}$, for $i = r+1, \ldots, n$.

Singular value decomposition is useful for analysis of the linear transformation $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ established in (A.35). According to a geometric interpretation, the matrix $\boldsymbol{A}$ transforms the unit sphere in $\mathbb{R}^n$ defined by $\|\boldsymbol{x}\| = 1$ into the set of vectors $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ which define an *ellipsoid* of dimension r in $\mathbb{R}^m$. The singular values are the lengths of the various axes of the ellipsoid. The *condition number* of the matrix

$$\kappa = \frac{\sigma_1}{\sigma_r}$$

is related to the eccentricity of the ellipsoid and provides a measure of ill-conditioning ($\kappa \gg 1$) for numerical solution of the system established by (A.35).

It is worth noticing that the numerical procedure of singular value decomposition is commonly adopted to compute the (right or left) pseudo-inverse $\boldsymbol{A}^\dagger$, even in the case of a matrix $\boldsymbol{A}$ not having full rank. In fact, from (A.67), (A.70) it is

$$\boldsymbol{A}^\dagger = \boldsymbol{V}\boldsymbol{\Sigma}^\dagger\boldsymbol{U}^T \tag{A.71}$$

with

$$\boldsymbol{\Sigma}^{\dagger} = \begin{bmatrix} \boldsymbol{D}^{\dagger} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \qquad \boldsymbol{D}^{\dagger} = \operatorname{diag}\left\{ \frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \ldots, \frac{1}{\sigma_r} \right\}. \tag{A.72}$$

Bibliography

A reference text on linear algebra is [169]. For matrix computation see [88]. The properties of pseudo-inverse matrices are discussed in [24].

B

Rigid-body Mechanics

The goal of this appendix is to recall some fundamental concepts of *rigid body mechanics* which are preliminary to the study of manipulator *kinematics*, *statics* and *dynamics*.

B.1 Kinematics

A *rigid body* is a system characterized by the constraint that the distance between any two points is always constant.

Consider a rigid body $\mathcal{B}$ moving with respect to an orthonormal reference frame O–xyz of unit vectors $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$, called *fixed frame*. The rigidity assumption allows the introduction of an orthonormal frame O'–$x'y'z'$ attached to the body, called *moving frame*, with respect to which the position of any point of $\mathcal{B}$ is independent of time. Let $\boldsymbol{x}'(t)$, $\boldsymbol{y}'(t)$, $\boldsymbol{z}'(t)$ be the unit vectors of the moving frame expressed in the fixed frame at time t.

The orientation of the moving frame O'–$x'y'z'$ at time t with respect to the fixed frame O–xyz can be expressed by means of the *orthogonal* (3×3) matrix

$$\boldsymbol{R}(t) = \begin{bmatrix} \boldsymbol{x}'^{T}(t)\boldsymbol{x} & \boldsymbol{y}'^{T}(t)\boldsymbol{x} & \boldsymbol{z}'^{T}(t)\boldsymbol{x} \\ \boldsymbol{x}'^{T}(t)\boldsymbol{y} & \boldsymbol{y}'^{T}(t)\boldsymbol{y} & \boldsymbol{z}'^{T}(t)\boldsymbol{y} \\ \boldsymbol{x}'^{T}(t)\boldsymbol{z} & \boldsymbol{y}'^{T}(t)\boldsymbol{z} & \boldsymbol{z}'^{T}(t)\boldsymbol{z} \end{bmatrix}, \tag{B.1}$$

which is termed *rotation matrix* defined in the orthonormal special group $SO(3)$ of the (3×3) matrices with orthonormal columns and determinant equal to 1. The columns of the matrix in (B.1) represent the components of the unit vectors of the moving frame when expressed in the fixed frame, whereas the rows represent the components of the unit vectors of the fixed frame when expressed in the moving frame.

Let $\boldsymbol{p}'$ be the *constant* position vector of a generic point P of $\mathcal{B}$ in the moving frame O'–$x'y'z'$. The motion of P with respect to the fixed frame O–xyz is described by the equation

$$\boldsymbol{p}(t) = \boldsymbol{p}_{O'}(t) + \boldsymbol{R}(t)\boldsymbol{p}', \tag{B.2}$$

where $\boldsymbol{p}_{O'}(t)$ is the position vector of origin O' of the moving frame with respect to the fixed frame.

Notice that a position vector is a *bound vector* since its line of application and point of application are both prescribed, in addition to its direction; the point of application typically coincides with the origin of a reference frame. Therefore, to transform a bound vector from a frame to another, both translation and rotation between the two frames must be taken into account.

If the positions of the points of $\mathcal{B}$ in the moving frame are known, it follows from (B.2) that the motion of each point of $\mathcal{B}$ with respect to the fixed frame is uniquely determined once the position of the origin and the orientation of the moving frame with respect to the fixed frame are specified in time. The origin of the moving frame is determined by *three* scalar functions of time. Since the orthonormality conditions impose six constraints on the nine elements of matrix $\boldsymbol{R}(t)$, the *orientation* of the moving frame depends only on *three* independent scalar functions, three being the minimum number of parameters to represent $SO(3)$.[1]

Therefore, a rigid body motion is described by arbitrarily specifying *six* scalar functions of time, which describe the body *pose* (position + orientation). The resulting rigid motions belong to the *special Euclidean group* $SE(3) = \mathbb{R}^3 \times SO(3)$.

The expression in (B.2) continues to hold if the position vector $\boldsymbol{p}_{O'}(t)$ of the origin of the moving frame is replaced with the position vector of any other point of $\mathcal{B}$, i.e.,

$$\boldsymbol{p}(t) = \boldsymbol{p}_Q(t) + \boldsymbol{R}(t)(\boldsymbol{p}' - \boldsymbol{p}'_Q) \tag{B.3}$$

where $\boldsymbol{p}_Q(t)$ and $\boldsymbol{p}'_Q$ are the position vectors of a point Q of $\mathcal{B}$ in the fixed and moving frames, respectively.

In the following, for simplicity of notation, the dependence on the time variable t will be dropped.

Differentiating (B.3) with respect to time gives the known velocity composition rule

$$\dot{\boldsymbol{p}} = \dot{\boldsymbol{p}}_Q + \boldsymbol{\omega} \times (\boldsymbol{p} - \boldsymbol{p}_Q), \tag{B.4}$$

where $\boldsymbol{\omega}$ is the *angular velocity* of rigid body $\mathcal{B}$. Notice that $\boldsymbol{\omega}$ is a *free vector* since its point of application is not prescribed. To transform a free vector from a frame to another, only rotation between the two frames must be taken into account.

By recalling the definition of the skew-symmetric operator $\boldsymbol{S}(\cdot)$ in (A.32), the expression in (B.4) can be rewritten as

$$\begin{aligned}\dot{\boldsymbol{p}} &= \dot{\boldsymbol{p}}_Q + \boldsymbol{S}(\boldsymbol{\omega})(\boldsymbol{p} - \boldsymbol{p}_Q) \\ &= \dot{\boldsymbol{p}}_Q + \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}(\boldsymbol{p}' - \boldsymbol{p}'_Q).\end{aligned}$$

[1] The minimum number of parameters represent a special orthonormal group $SO(m)$ is equal to $m(m-1)/2$.

Comparing this equation with the formal time derivative of (B.3) leads to the result

$$\dot{\boldsymbol{R}} = \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}. \tag{B.5}$$

In view of (B.4), the *elementary displacement* of a point P of the rigid body $\mathcal{B}$ in the time interval $(t, t+dt)$ is

$$\begin{aligned}d\boldsymbol{p} = \dot{\boldsymbol{p}}dt &= \big(\dot{\boldsymbol{p}}_Q + \boldsymbol{\omega} \times (\boldsymbol{p} - \boldsymbol{p}_Q)\big)dt \\ &= d\boldsymbol{p}_Q + \boldsymbol{\omega}dt \times (\boldsymbol{p} - \boldsymbol{p}_Q).\end{aligned} \tag{B.6}$$

Differentiating (B.4) with respect to time yields the following expression for acceleration:

$$\ddot{\boldsymbol{p}} = \ddot{\boldsymbol{p}}_Q + \dot{\boldsymbol{\omega}} \times (\boldsymbol{p} - \boldsymbol{p}_Q) + \boldsymbol{\omega} \times \big(\boldsymbol{\omega} \times (\boldsymbol{p} - \boldsymbol{p}_Q)\big). \tag{B.7}$$

B.2 Dynamics

Let ρdV be the mass of an elementary particle of a rigid body $\mathcal{B}$, where ρ denotes the density of the particle of volume dV. Also let $V_{\mathcal{B}}$ be the body volume and $m = \int_{V_{\mathcal{B}}} \rho dV$ its *total mass* assumed to be constant. If $\boldsymbol{p}$ denotes the position vector of the particle of mass ρdV in the frame O–xyz, the *centre of mass* of $\mathcal{B}$ is defined as the point C whose position vector is

$$\boldsymbol{p}_C = \frac{1}{m}\int_{V_{\mathcal{B}}} \boldsymbol{p}\rho dV. \tag{B.8}$$

In the case when $\mathcal{B}$ is the union of n distinct parts of mass $m_1, \ldots, m_n$ and centres of mass $\boldsymbol{p}_{C1} \ldots \boldsymbol{p}_{Cn}$, the centre of mass of $\mathcal{B}$ can be computed as

$$\boldsymbol{p}_C = \frac{1}{m}\sum_{i=1}^{n} m_i \boldsymbol{p}_{Ci}$$

with $m = \sum_{i=1}^{n} m_i$.

Let r be a line passing by O and $d(\boldsymbol{p})$ the distance from r of the particle of $\mathcal{B}$ of mass ρdV and position vector $\boldsymbol{p}$. The *moment of inertia* of body $\mathcal{B}$ with respect to line r is defined as the positive scalar

$$I_r = \int_{V_{\mathcal{B}}} d^2(\boldsymbol{p})\rho dV.$$

Let $\boldsymbol{r}$ denote the unit vector of line r; then, the moment of inertia of $\mathcal{B}$ with respect to line r can be expressed as

$$I_r = \boldsymbol{r}^T \left(\int_{V_{\mathcal{B}}} \boldsymbol{S}^T(\boldsymbol{p})\boldsymbol{S}(\boldsymbol{p})\rho dV\right)\boldsymbol{r} = \boldsymbol{r}^T \boldsymbol{I}_O \boldsymbol{r}, \tag{B.9}$$

where $\boldsymbol{S}(\cdot)$ is the skew-symmetric operator in (A.31), and the *symmetric*, *positive definite* matrix

$$\boldsymbol{I}_O = \begin{bmatrix} \int_{V_\mathcal{B}}(p_y^2+p_z^2)\rho dV & -\int_{V_\mathcal{B}} p_x p_y \rho dV & -\int_{V_\mathcal{B}} p_x p_z \rho dV \\ * & \int_{V_\mathcal{B}}(p_x^2+p_z^2)\rho dV & -\int_{V_\mathcal{B}} p_y p_z \rho dV \\ * & * & \int_{V_\mathcal{B}}(p_x^2+p_y^2)\rho dV \end{bmatrix}$$

$$= \begin{bmatrix} I_{Oxx} & -I_{Oxy} & -I_{Oxz} \\ * & I_{Oyy} & -I_{Oyz} \\ * & * & I_{Ozz} \end{bmatrix} \tag{B.10}$$

is termed *inertia tensor* of body $\mathcal{B}$ relative to pole O.[2] The (positive) elements $I_{Oxx}, I_{Oyy}, I_{Ozz}$ are the *inertia moments* with respect to three coordinate axes of the reference frame, whereas the elements I_{Oxy}, I_{Oxz}, I_{Oyz} (of any sign) are said to be *products of inertia*.

The expression of the inertia tensor of a rigid body $\mathcal{B}$ depends both on the pole and the reference frame. If orientation of the reference frame with origin at O is changed according to a rotation matrix $\boldsymbol{R}$, the inertia tensor $\boldsymbol{I}'_O$ in the new frame is related to $\boldsymbol{I}_O$ by the relationship

$$\boldsymbol{I}_O = \boldsymbol{R}\boldsymbol{I}'_O\boldsymbol{R}^T. \tag{B.11}$$

The way an inertia tensor is transformed when the pole is changed can be inferred by the following equation, also known as *Steiner theorem* or parallel axis theorem:

$$\boldsymbol{I}_O = \boldsymbol{I}_C + m\boldsymbol{S}^T(\boldsymbol{p}_C)\boldsymbol{S}(\boldsymbol{p}_C), \tag{B.12}$$

where $\boldsymbol{I}_C$ is the inertia tensor relative to the centre of mass of $\mathcal{B}$, when expressed in a frame parallel to the frame with origin at O and with origin at the centre of mass C.

Since the inertia tensor is a symmetric positive definite matrix, there always exists a reference frame in which the inertia tensor attains a diagonal form; such a frame is said to be a *principal frame* (relative to pole O) and its coordinate axes are said to be *principal axes*. In the case when pole O coincides with the centre of mass, the frame is said to be a *central frame* and its axes are said to be *central axes*.

Notice that if the rigid body is moving with respect to the reference frame with origin at O, then the elements of the inertia tensor $\boldsymbol{I}_O$ become a function of time. With respect to a pole and a reference frame attached to the body (moving frame), instead, the elements of the inertia tensor represent six structural constants of the body which are known once the pole and reference frame have been specified.

[2] The symbol '$*$' has been used to avoid rewriting the symmetric elements.

Let $\dot{\boldsymbol{p}}$ be the velocity of a particle of $\mathcal{B}$ of elementary mass ρdV in frame O–xyz. The *linear momentum* of body $\mathcal{B}$ is defined as the vector

$$\boldsymbol{l} = \int_{V_\mathcal{B}} \dot{\boldsymbol{p}}\rho dV = m\dot{\boldsymbol{p}}_C. \tag{B.13}$$

Let Ω be any point in space and $\boldsymbol{p}_\Omega$ its position vector in frame O–xyz; then, the *angular momentum* of body $\mathcal{B}$ relative to pole Ω is defined as the vector

$$\boldsymbol{k}_\Omega = \int_{V_\mathcal{B}} \dot{\boldsymbol{p}} \times (\boldsymbol{p}_\Omega - \boldsymbol{p})\rho dV.$$

The pole can be either fixed or moving with respect to the reference frame. The angular momentum of a rigid body has the following notable expression:

$$\boldsymbol{k}_\Omega = \boldsymbol{I}_C\boldsymbol{\omega} + m\dot{\boldsymbol{p}}_C \times (\boldsymbol{p}_\Omega - \boldsymbol{p}_C), \tag{B.14}$$

where $\boldsymbol{I}_C$ is the inertia tensor relative to the centre of mass, when expressed in a frame parallel to the reference frame with origin at the centre of mass.

The *forces* acting on a generic system of material particles can be distinguished into *internal* forces and *external* forces.

The internal forces, exerted by one part of the system on another, have null linear and angular momentum and thus they do not influence rigid body motion.

The external forces, exerted on the system by an agency outside the system, in the case of a rigid body $\mathcal{B}$ are distinguished into *active* forces and *reaction* forces.

The active forces can be either *concentrated* forces or *body* forces. The former are applied to specific points of $\mathcal{B}$, whereas the latter act on all elementary particles of the body. An example of body force is the *gravitational force* which, for any elementary particle of mass ρdV, is equal to $\boldsymbol{g}_0\rho dV$ where $\boldsymbol{g}_0$ is the gravity acceleration vector.

The reaction forces are those exerted because of surface contact between two or more bodies. Such forces can be distributed on the contact surfaces or they can be assumed to be concentrated.

For a rigid body $\mathcal{B}$ subject to gravitational force, as well as to active and or reaction forces $\boldsymbol{f}_1 \dots \boldsymbol{f}_n$ concentrated at points $\boldsymbol{p}_1 \dots \boldsymbol{p}_n$, the *resultant* of the external forces $\boldsymbol{f}$ and the *resultant moment* $\boldsymbol{\mu}_\Omega$ with respect to a pole Ω are respectively

$$\boldsymbol{f} = \int_{V_\mathcal{B}} \boldsymbol{g}_0\rho dV + \sum_{i=1}^{n} \boldsymbol{f}_i = m\boldsymbol{g}_0 + \sum_{i=1}^{n} \boldsymbol{f}_i \tag{B.15}$$

$$\boldsymbol{\mu}_\Omega = \int_{V_\mathcal{B}} \boldsymbol{g}_0 \times (\boldsymbol{p}_\Omega - \boldsymbol{p})\rho dV + \sum_{i=1}^{n} \boldsymbol{f}_i \times (\boldsymbol{p}_\Omega - \boldsymbol{p}_i)$$

$$= m\boldsymbol{g}_0 \times (\boldsymbol{p}_\Omega - \boldsymbol{p}_C) + \sum_{i=1}^{n} \boldsymbol{f}_i \times (\boldsymbol{p}_\Omega - \boldsymbol{p}_i). \tag{B.16}$$

In the case when $\boldsymbol{f}$ and $\boldsymbol{\mu}_\Omega$ are known and it is desired to compute the resultant moment with respect to a point Ω' other than Ω, the following relation holds:

$$\boldsymbol{\mu}_{\Omega'} = \boldsymbol{\mu}_\Omega + \boldsymbol{f} \times (\boldsymbol{p}_{\Omega'} - \boldsymbol{p}_\Omega). \tag{B.17}$$

Consider now a generic system of material particles subject to *external forces* of resultant $\boldsymbol{f}$ and resultant moment $\boldsymbol{\mu}_\Omega$. The motion of the system in a frame O–xyz is established by the following *fundamental principles of dynamics* (Newton laws of motion):

$$\boldsymbol{f} = \dot{\boldsymbol{l}} \tag{B.18}$$

$$\boldsymbol{\mu}_\Omega = \dot{\boldsymbol{k}}_\Omega \tag{B.19}$$

where Ω is a pole fixed or coincident with the centre of mass C of the system. These equations hold for any mechanical system and can be used even in the case of variable mass. For a system with constant mass, computing the time derivative of the momentum in (B.18) gives *Newton equations of motion* in the form

$$\boldsymbol{f} = m\ddot{\boldsymbol{p}}_C, \tag{B.20}$$

where the quantity on the right-hand side represents the *resultant of inertia forces*.

If, besides the assumption of constant mass, the assumption of rigid system holds too, the expression in (B.14) of the angular momentum with (B.19) yield *Euler equations of motion* in the form

$$\boldsymbol{\mu}_\Omega = \boldsymbol{I}_\Omega \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\boldsymbol{I}_\Omega \boldsymbol{\omega}), \tag{B.21}$$

where the quantity on the right-hand side represents the *resultant moment of inertia forces*.

For a system constituted by a set of rigid bodies, the external forces obviously do not include the reaction forces exerted between the bodies belonging to the same system.

B.3 Work and Energy

Given a force $\boldsymbol{f}_i$ applied at a point of position $\boldsymbol{p}_i$ with respect to frame O–xyz, the *elementary work* of the force $\boldsymbol{f}_i$ on the displacement $d\boldsymbol{p}_i = \dot{\boldsymbol{p}}_i dt$ is defined as the scalar

$$dW_i = \boldsymbol{f}_i^T d\boldsymbol{p}_i.$$

For a rigid body $\mathcal{B}$ subject to a system of forces of resultant $\boldsymbol{f}$ and resultant moment $\boldsymbol{\mu}_Q$ with respect to any point Q of $\mathcal{B}$, the elementary work on the rigid displacement (B.6) is given by

$$dW = (\boldsymbol{f}^T \dot{\boldsymbol{p}}_Q + \boldsymbol{\mu}_Q^T \boldsymbol{\omega}) dt = \boldsymbol{f}^T d\boldsymbol{p}_Q + \boldsymbol{\mu}_Q^T \boldsymbol{\omega} dt. \tag{B.22}$$

The *kinetic energy* of a body $\mathcal{B}$ is defined as the scalar quantity

$$\mathcal{T} = \frac{1}{2} \int_{V_\mathcal{B}} \dot{\boldsymbol{p}}^T \dot{\boldsymbol{p}} \rho dV$$

which, for a rigid body, takes on the notable expression

$$\mathcal{T} = \frac{1}{2} m \dot{\boldsymbol{p}}_C^T \dot{\boldsymbol{p}}_C + \frac{1}{2} \boldsymbol{\omega}^T \boldsymbol{I}_C \boldsymbol{\omega} \tag{B.23}$$

where $\boldsymbol{I}_C$ is the inertia tensor relative to the centre of mass expressed in a frame parallel to the reference frame with origin at the centre of mass.

A system of position forces, i.e., the forces depending only on the positions of the points of application, is said to be *conservative* if the work done by each force is independent of the trajectory described by the point of application of the force but it depends only on the initial and final positions of the point of application. In this case, the elementary work of the system of forces is equal to minus the total differential of a scalar function termed *potential energy*, i.e.,

$$dW = -d\mathcal{U}. \tag{B.24}$$

An example of a conservative system of forces on a rigid body is the gravitational force, with which is associated the potential energy

$$\mathcal{U} = -\int_{V_\mathcal{B}} \boldsymbol{g}_0^T \boldsymbol{p} \rho dV = -m \boldsymbol{g}_0^T \boldsymbol{p}_C. \tag{B.25}$$

B.4 Constrained Systems

Consider a system $\mathcal{B}_r$ of r rigid bodies and assume that all the elements of $\mathcal{B}_r$ can reach any position in space. In order to find uniquely the position of all the points of the system, it is necessary to assign a vector $\boldsymbol{x} = [\, x_1 \quad \dots \quad x_p \,]^T$ of $6r = p$ parameters, termed *configuration*. These parameters are termed *Lagrange* or *generalized coordinates* of the *unconstrained* system $\mathcal{B}_r$, and p determines the number of *degrees of freedom* (DOFs).

Any limitation on the mobility of the system $\mathcal{B}_r$ is termed *constraint*. A constraint acting on $\mathcal{B}_r$ is said to be *holonomic* if it is expressed by a system of equations

$$\boldsymbol{h}(\boldsymbol{x}, t) = \boldsymbol{0}, \tag{B.26}$$

where $\boldsymbol{h}$ is a vector of dimensions $(s \times 1)$, with $s < m$. On the other hand, a constraint in the form $\boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}, t) = \boldsymbol{0}$ which is nonintegrable is said to be *nonholonomic*. For simplicity, only equality (or *bilateral*) constraints are considered. If the equations in (B.26) do not explicitly depend on time, the constraint is said to be *scleronomic*.

On the assumption that $\boldsymbol{h}$ has continuous and continuously differentiable components, and its Jacobian $\partial \boldsymbol{h} / \partial \boldsymbol{x}$ has full rank, the equations in (B.26)

allow the elimination of s out of m coordinates of the system $\mathcal{B}_r$. With the remaining $n = m - s$ coordinates it is possible to determine uniquely the configurations of $\mathcal{B}_r$ satisfying the constraints (B.26). Such coordinates are the *Lagrange* or *generalized coordinates* and n is the number of *degrees of freedom* of the *unconstrained* system $\mathcal{B}_r$.[3]

The motion of a system $\mathcal{B}_r$ with n DOFs and holonomic equality constraints can be described by equations of the form

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{q}(t), t), \tag{B.27}$$

where $\boldsymbol{q}(t) = [\, q_1(t) \quad \dots \quad q_n(t) \,]^T$ is a vector of Lagrange coordinates.

The *elementary displacement* of system (B.27) relative to the interval $(t, t+dt)$ is defined as

$$d\boldsymbol{x} = \frac{\partial \boldsymbol{x}(\boldsymbol{q}, t)}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} dt + \frac{\partial \boldsymbol{x}(\boldsymbol{q}, t)}{\partial t} dt. \tag{B.28}$$

The *virtual displacement* of system (B.27) at time t, relative to an increment $\delta\boldsymbol{\lambda}$, is defined as the quantity

$$\delta\boldsymbol{x} = \frac{\partial \boldsymbol{x}(\boldsymbol{q}, t)}{\partial \boldsymbol{q}} \delta\boldsymbol{q}. \tag{B.29}$$

The difference between the elementary displacement and the virtual displacement is that the former is relative to an actual motion of the system in an interval $(t, t + dt)$ which is consistent with the constraints, while the latter is relative to an imaginary motion of the system when the constraints are made invariant and equal to those at time t.

For a system with time-invariant constraints, the equations of motion (B.27) become

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{q}(t)), \tag{B.30}$$

and then, by setting $\delta\boldsymbol{\lambda} = d\boldsymbol{\lambda} = \dot{\boldsymbol{\lambda}} dt$, the virtual displacements (B.29) coincide with the elementary displacements (B.28).

To the concept of virtual displacement can be associated that of *virtual work* of a system of forces, by considering a virtual displacement instead of an elementary displacement.

If external forces are distinguished into *active forces* and *reaction forces*, a direct consequence of the principles of dynamics (B.18), (B.19) applied to the system of rigid bodies $\mathcal{B}_r$ is that, for each virtual displacement, the following relation holds:

$$\delta W_m + \delta W_a + \delta W_h = 0, \tag{B.31}$$

where δW_m, δW_a, δW_h are the total virtual works done by the inertia, active, reaction forces, respectively.

[3] In general, the Lagrange coordinates of a constrained system have a local validity; in certain cases, such as the joint variables of a manipulator, they can have a global validity.

In the case of *frictionless* equality constraints, reaction forces are exerted orthogonally to the contact surfaces and the virtual work is always null. Hence, (B.31) reduces to

$$\delta W_m + \delta W_a = 0. \tag{B.32}$$

For a steady system, inertia forces are identically null. Then the condition for the equilibrium of system $\mathcal{B}_r$ is that the virtual work of the active forces is identically null on any virtual displacement, which gives the fundamental equation of *statics* of a constrained system

$$\delta W_a = 0 \tag{B.33}$$

known as *principle of virtual work*. Expressing (B.33) in terms of the increment $\delta\boldsymbol{\lambda}$ of generalized coordinates leads to

$$\delta W_a = \boldsymbol{\zeta}^T \delta\boldsymbol{q} = 0 \tag{B.34}$$

where $\boldsymbol{\zeta}$ denotes the $(n \times 1)$ vector of active *generalized* forces.

In the dynamic case, it is worth distinguishing active forces into *conservative* (that can be derived from a potential) and *nonconservative*. The virtual work of conservative forces is given by

$$\delta W_c = -\frac{\partial \mathcal{U}}{\partial \boldsymbol{q}} \delta\boldsymbol{q}, \tag{B.35}$$

where $\mathcal{U}(\boldsymbol{\lambda})$ is the total potential energy of the system. The work of nonconservative forces can be expressed in the form

$$\delta W_{nc} = \boldsymbol{\xi}^T \delta\boldsymbol{q}, \tag{B.36}$$

where $\boldsymbol{\xi}$ denotes the vector of nonconservative generalized forces. It follows that the vector of active generalized forces is

$$\boldsymbol{\zeta} = \boldsymbol{\xi} - \left(\frac{\partial \mathcal{U}}{\partial \boldsymbol{q}} \right)^T. \tag{B.37}$$

Moreover, the work of inertia forces can be computed from the total kinetic energy of system $\mathcal{T}$ as

$$\delta W_m = \left(\frac{\partial \mathcal{T}}{\partial \boldsymbol{q}} - \frac{d}{dt} \frac{\partial \mathcal{T}}{\partial \dot{\boldsymbol{q}}} \right) \delta\boldsymbol{q}. \tag{B.38}$$

Substituting (B.35), (B.36), (B.38) into (B.32) and observing that (B.32) holds true for any increment $\delta\boldsymbol{\lambda}$ leads to *Lagrange equations*

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{q}}} \right)^T - \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{q}} \right)^T = \boldsymbol{\xi}, \tag{B.39}$$

where

$$\mathcal{L} = \mathcal{T} - \mathcal{U} \tag{B.40}$$

is the *Lagrangian* function of the system. The equations in (B.39) completely describe the dynamic behaviour of an n-DOF system with holonomic equality constraints.

The sum of kinetic and potential energy of a system with time-invariant constraints is termed *Hamiltonian* function

$$\mathcal{H} = \mathcal{T} + \mathcal{U}. \tag{B.41}$$

Conservation of energy dictates that the time derivative of the Hamiltonian must balance the power generated by the nonconservative forces acting on the system, i.e.,

$$\frac{d\mathcal{H}}{dt} = \boldsymbol{\xi}^T \dot{\boldsymbol{q}}. \tag{B.42}$$

In view of (B.37), (B.41), the equation in (B.42) becomes

$$\frac{d\mathcal{T}}{dt} = \boldsymbol{\zeta}^T \dot{\boldsymbol{q}}. \tag{B.43}$$

Bibliography

The fundamental concepts of rigid-body mechanics and constrained systems can be found in classical texts such as [87, 154, 224]. An authoritative reference on rigid-body system dynamics is [187].

C

Feedback Control

As a premise to the study of manipulator decentralized control and centralized control, the fundamental principles of *feedback control* of *linear systems* are recalled, and an approach to the determination of control laws for *nonlinear systems* based on the use of *Lyapunov functions* is presented.

C.1 Control of Single-input/Single-output Linear Systems

According to classical *automatic control* theory of *linear time-invariant single-input/single-output systems*, in order to servo the output $y(t)$ of a system to a reference $r(t)$, it is worth adopting a *negative feedback control* structure. This structure indeed allows the use of approximate mathematical models to describe the input/output relationship of the system to control, since negative feedback has a potential for reducing the effects of system parameter variations and nonmeasurable disturbance inputs $d(t)$ on the output.

This structure can be represented in the *domain of complex variable* s as in the block scheme of Fig. C.1, where $G(s)$, $H(s)$ and $C(s)$ are the transfer functions of the system to control, the transducer and the controller, respectively. From this scheme it is easy to derive

$$Y(s) = W(s)R(s) + W_D(s)D(s), \tag{C.1}$$

where

$$W(s) = \frac{C(s)G(s)}{1 + C(s)G(s)H(s)} \tag{C.2}$$

is the *closed-loop input/output transfer function* and

$$W_D(s) = \frac{G(s)}{1 + C(s)G(s)H(s)} \tag{C.3}$$

is the *disturbance/output transfer function*.

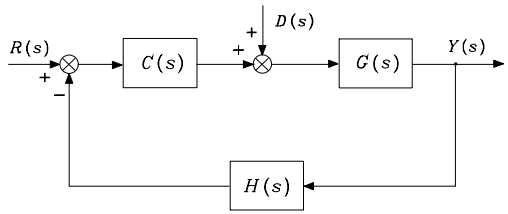

Fig. C.1. Feedback control structure

The goal of the controller design is to find a control structure $C(s)$ ensuring that the output variable $Y(s)$ tracks a reference input $R(s)$. Further, the controller should guarantee that the effects of the disturbance input $D(s)$ on the output variable are suitably reduced. The goal is then twofold, namely, *reference tracking* and *disturbance rejection*.

The basic problem for controller design consists of the determination of an action $C(s)$ which can make the system *asymptotically stable*. In the absence of positive or null real part pole/zero and zero/pole cancellation in the *open-loop* function $F(s) = C(s)G(s)H(s)$, a necessary and sufficient condition for asymptotic stability is that the *poles* of $W(s)$ and $W_D(s)$ have all *negative real parts*; such poles coincide with the zeros of the rational transfer function $1 + F(s)$. Testing for this condition can be performed by resorting to stability criteria, thus avoiding computation of the function zeros.

Routh criterion allows the determination of the sign of the real parts of the zeros of the function $1 + F(s)$ by constructing a table with the coefficients of the polynomial at the numerator of $1 + F(s)$ (*characteristic polynomial*).

Routh criterion is easy to apply for testing stability of a feedback system, but it does not provide a direct relationship between the open-loop function and stability of the closed-loop system. It is then worth resorting to *Nyquist criterion* which is based on the representation, in the complex plane, of the open-loop transfer function $F(s)$ evaluated in the *domain of real angular frequency* ($s = j\omega, -\infty < \omega < +\infty$).

Drawing of Nyquist plot and computation of the number of circles made by the vector representing the complex number $1 + F(j\omega)$ when ω continuously varies from $-\infty$ to $+\infty$ allows a test on whether or not the *closed-loop* system is asymptotically stable. It is also possible to determine the number of positive, null and negative real part roots of the characteristic polynomial, similarly to application of Routh criterion. Nonetheless, Nyquist criterion is based on the plot of the open-loop transfer function, and thus it allows the determination of a direct relationship between this function and closed-loop system stability. It is then possible from an examination of the Nyquist plot to draw suggestions on the controller structure $C(s)$ which ensures closed-loop system asymptotic stability.

If the closed-loop system is asymptotically stable, the *steady-state response* to a sinusoidal input $r(t)$, with $d(t) = 0$, is sinusoidal, too. In this case, the function $W(s)$, evaluated for $s = j\omega$, is termed *frequency response function*; the frequency response function of a feedback system can be assimilated to that of a low-pass filter with the possible occurrence of a *resonance peak* inside its *bandwidth*.

As regards the transducer, this should be chosen so that its bandwidth is much greater than the feedback system bandwidth, in order to ensure a nearly instantaneous response for any value of ω inside the bandwidth of $W(j\omega)$. Therefore, setting $H(j\omega) \approx H_0$ and assuming that the *loop gain* $|C(j\omega)G(j\omega)H_0| \gg 1$ in the same bandwidth, the expression in (C.1) for $s = j\omega$ can be approximated as

$$Y(j\omega) \approx \frac{R(j\omega)}{H_0} + \frac{D(j\omega)}{C(j\omega)H_0}.$$

Assuming $R(j\omega) = H_0 Y_d(j\omega)$ leads to

$$Y(j\omega) \approx Y_d(j\omega) + \frac{D(j\omega)}{C(j\omega)H_0}; \tag{C.4}$$

i.e., the output tracks the desired output $Y_d(j\omega)$ and the frequency components of the disturbance in the bandwidth of $W(j\omega)$ produce an effect on the output which can be reduced by increasing $|C(j\omega)H_0|$. Furthermore, if the disturbance input is a constant, the steady-state output is not influenced by the disturbance as long as $C(s)$ has at least a pole at the origin.

Therefore, a feedback control system is capable of establishing a proportional relationship between the desired output and the actual output, as evidenced by (C.4). This equation, however, requires that the frequency content of the input (desired output) be inside the frequency range for which the loop gain is much greater than unity.

The previous considerations show the advantage of including a *proportional action* and an *integral action* in the controller $C(s)$, leading to the transfer function

$$C(s) = K_I \frac{1 + sT_I}{s} \tag{C.5}$$

of a *proportional-integral controller* (PI); T_I is the time constant of the integral action and the quantity $K_I T_I$ is called proportional sensitivity.

The adoption of a PI controller is effective for low-frequency response of the system, but it may involve a reduction of *stability margins* and/or a reduction of closed-loop system bandwidth. To avoid these drawbacks, a *derivative action* can be added to the proportional and integral actions, leading to the transfer function

$$C(s) = K_I \frac{1 + sT_I + s^2 T_D T_I}{s} \tag{C.6}$$

of a *proportional-integral-derivative controller* (PID); T_D denotes the time constant of the derivative action. Notice that physical realizability of (C.6)

demands the introduction of a high-frequency pole which little influences the input/output relationship in the system bandwidth. The transfer function in (C.6) is characterized by the presence of two zeros which provide a stabilizing action and an enlargement of the closed-loop system bandwidth. Bandwidth enlargement implies shorter *response time* of the system, in terms of both variations of the reference signal and recovery action of the feedback system to output variations induced by the disturbance input.

The parameters of the adopted control structure should be chosen so as to satisfy requirements on the system behaviour at *steady state* and during the *transient*. Classical tools to determine such parameters are the *root locus* in the domain of the complex variable s or the *Nichols chart* in the domain of the real angular frequency ω. The two tools are conceptually equivalent. Their potential is different in that root locus allows a control law to be found which assigns the exact parameters of the closed-loop system time response, whereas Nichols chart allows a controller to be specified which confers good transient and steady-state behaviour to the system response.

A feedback system with strict requirements on the steady-state and transient behaviour, typically, has a response that can be assimilated to that of a *second-order system*. In fact, even for closed-loop functions of greater order, it is possible to identify a pair of complex conjugate poles whose real part absolute value is smaller than the real part absolute values of the other poles. Such a pair of poles is *dominant* in that its contribution to the transient response prevails over that of the other poles. It is then possible to approximate the input/output relationship with the transfer function

$$W(s) = \frac{k_W}{1 + \dfrac{2\zeta s}{\omega_n} + \dfrac{s^2}{\omega_n^2}} \tag{C.7}$$

which has to be realized by a proper choice of the controller. Regarding the values to assign to the parameters characterizing the transfer function in (C.7), the following remarks are in order. The constant k_W represents the input/output *steady-state gain*, which is equal to $1/H_0$ if $C(s)G(s)H_0$ has at least a pole at the origin. The *natural frequency* ω_n is the modulus of the complex conjugate poles, whose real part is given by $-\zeta\omega_n$ where ζ is the *damping ratio* of the pair of poles.

The influence of parameters ζ and ω_n on the closed-loop frequency response can be evaluated in terms of the resonance peak magnitude

$$M_r = \frac{1}{2\zeta\sqrt{1-\zeta^2}},$$

occurring at the resonant frequency

$$\omega_r = \omega_n\sqrt{1-2\zeta^2},$$

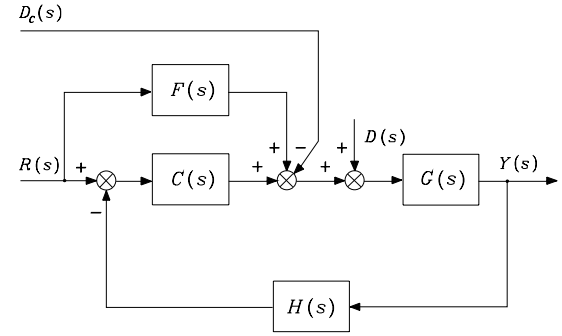

Fig. C.2. Feedback control structure with feedforward compensation

and of the 3 dB bandwidth

$$\omega_3 = \omega_n\sqrt{1-2\zeta^2+\sqrt{2-4\zeta^2+4\zeta^4}}.$$

A step input is typically used to characterize the transient response in the time domain. The influence of parameters ζ and ω_n on the *step response* can be evaluated in terms of the percentage of *overshoot*

$$s\% = 100\exp(-\pi\zeta/\sqrt{1-\zeta^2}),$$

of the *rise time*

$$t_r \approx \frac{1.8}{\omega_n}$$

and of the *settling time* within 1%

$$t_s = \frac{4.6}{\zeta\omega_n}.$$

The adoption of a *feedforward compensation* action represents a feasible solution both for tracking a time-varying reference input and for enhancing rejection of the effects of a disturbance on the output. Consider the general scheme in Fig. C.2. Let $R(s)$ denote a given input reference and $D_c(s)$ denote a computed estimate of the disturbance $D(s)$; the introduction of the feedforward action yields the input/output relationship

$$Y(s) = \left(\frac{C(s)G(s)}{1+C(s)G(s)H(s)} + \frac{F(s)G(s)}{1+C(s)G(s)H(s)}\right)R(s) + \frac{G(s)}{1+C(s)G(s)H(s)}\big(D(s)-D_c(s)\big). \tag{C.8}$$

By assuming that the desired output is related to the reference input by a constant factor K_d and regarding the transducer as an instantaneous system ($H(s) \approx H_0 = 1/K_d$) for the current operating conditions, the choice

$$F(s) = \frac{K_d}{G(s)} \tag{C.9}$$

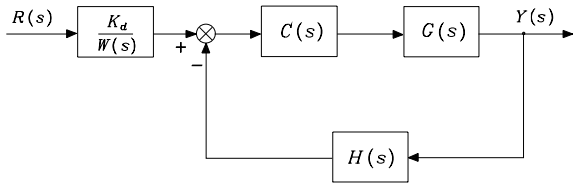

Fig. C.3. Feedback control structure with inverse model technique

yields the input/output relationship

$$Y(s) = Y_d(s) + \frac{G(s)}{1 + C(s)G(s)H_0}\big(D(s) - D_c(s)\big). \tag{C.10}$$

If $|C(j\omega)G(j\omega)H_0| \gg 1$, the effect of the disturbance on the output is further reduced by means of an accurate estimate of the disturbance.

Feedforward compensation technique may lead to a solution, termed *inverse model control*, illustrated in the scheme of Fig. C.3. It should be remarked, however, that such a solution is based on dynamics cancellation, and thus it can be employed only for a minimum-phase system, i.e., a system whose poles and zeros have all strictly negative real parts. Further, one should consider physical realizability issues as well as effects of parameter variations which prevent perfect cancellation.

C.2 Control of Nonlinear Mechanical Systems

If the system to control does not satisfy the linearity property, the control design problem becomes more complex. The fact that a *system* is qualified as *nonlinear*, whenever linearity does not hold, leads to understanding how it is not possible to resort to general techniques for control design, but it is necessary to face the problem for each class of nonlinear systems which can be defined through imposition of special properties.

On the above premise, the control design problem of nonlinear systems described by the dynamic model

$$\boldsymbol{H}(\boldsymbol{x})\ddot{\boldsymbol{x}} + \boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \boldsymbol{u} \tag{C.11}$$

is considered, where $[\,\boldsymbol{x}^T \quad \dot{\boldsymbol{x}}^T\,]^T$ denotes the $(2n \times 1)$ *state* vector of the system, $\boldsymbol{u}$ is the $(n \times 1)$ *input* vector, $\boldsymbol{H}(\boldsymbol{x})$ is an $(n \times n)$ *positive definite* (and thus invertible) matrix depending on $\boldsymbol{x}$, and $\boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ is an $(n \times 1)$ vector depending on state. Several *mechanical systems* can be reduced to this class, including manipulators with rigid links and joints.

The *control* law can be found through a nonlinear compensating action obtained by choosing the following *nonlinear state feedback* law (*inverse dynamics* control):

$$\boldsymbol{u} = \widehat{\boldsymbol{H}}(\boldsymbol{x})\boldsymbol{v} + \widehat{\boldsymbol{h}}(\boldsymbol{x}, \dot{\boldsymbol{x}}) \tag{C.12}$$

where $\widehat{\boldsymbol{H}}(\boldsymbol{x})$ and $\widehat{\boldsymbol{h}}(\boldsymbol{x})$ respectively denote the *estimates* of the terms $\boldsymbol{H}(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{x})$, computed on the basis of measures on the system state, and $\boldsymbol{v}$ is a new control input to be defined later. In general, it is

$$\widehat{\boldsymbol{H}}(\boldsymbol{x}) = \boldsymbol{H}(\boldsymbol{x}) + \Delta\boldsymbol{H}(\boldsymbol{x}) \tag{C.13}$$

$$\widehat{\boldsymbol{h}}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}) + \Delta\boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}) \tag{C.14}$$

because of the unavoidable modelling approximations or as a consequence of an intentional simplification in the compensating action. Substituting (C.12) into (C.11) and accounting for (C.13), (C.14) yields

$$\ddot{\boldsymbol{x}} = \boldsymbol{v} + \boldsymbol{z}(\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{v}) \tag{C.15}$$

where

$$\boldsymbol{z}(\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{v}) = \boldsymbol{H}^{-1}(\boldsymbol{x})\big(\Delta\boldsymbol{H}(\boldsymbol{x})\boldsymbol{v} + \Delta\boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}})\big).$$

If *tracking* of a trajectory $(\boldsymbol{x}_d(t), \dot{\boldsymbol{x}}_d(t), \ddot{\boldsymbol{x}}_d(t))$ is desired, the tracking error can be defined as

$$\boldsymbol{e} = \begin{bmatrix} \boldsymbol{x}_d - \boldsymbol{x} \\ \dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}} \end{bmatrix} \tag{C.16}$$

and it is necessary to derive the error dynamics equation to study convergence of the actual state to the desired one. To this end, the choice

$$\boldsymbol{v} = \ddot{\boldsymbol{x}}_d + \boldsymbol{w}(\boldsymbol{e}), \tag{C.17}$$

substituted into (C.15), leads to the error equation

$$\dot{\boldsymbol{e}} = \boldsymbol{F}\boldsymbol{e} - \boldsymbol{G}\boldsymbol{w}(\boldsymbol{e}) - \boldsymbol{G}\boldsymbol{z}(\boldsymbol{e}, \boldsymbol{x}_d, \dot{\boldsymbol{x}}_d, \ddot{\boldsymbol{x}}_d), \tag{C.18}$$

where the $(2n \times 2n)$ and $(2n \times n)$ matrices, respectively,

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{O} & \boldsymbol{I} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \qquad \boldsymbol{G} = \begin{bmatrix} \boldsymbol{O} \\ \boldsymbol{I} \end{bmatrix}$$

follow from the error definition in (C.16). Control law design consists of finding the error function $\boldsymbol{w}(\boldsymbol{e})$ which makes (C.18) *globally asymptotically stable*,[1] i.e.,

$$\lim_{t \to \infty} \boldsymbol{e}(t) = \boldsymbol{0}.$$

In the case of *perfect* nonlinear compensation ($\boldsymbol{z}(\cdot) = \boldsymbol{0}$), the simplest choice of the control action is the *linear* one

$$\begin{aligned} \boldsymbol{w}(\boldsymbol{e}) &= -\boldsymbol{K}_P(\boldsymbol{x}_d - \boldsymbol{x}) - \boldsymbol{K}_D(\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}) \\ &= [\,-\boldsymbol{K}_P \quad -\boldsymbol{K}_D\,]\,\boldsymbol{e}, \end{aligned} \tag{C.19}$$

[1] *Global* asymptotic stability is invoked to remark that the equilibrium state is asymptotically stable for any perturbation.

where asymptotic stability of the error equation is ensured by choosing *positive definite* matrices $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$. The error transient behaviour is determined by the eigenvalues of the matrix

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{O} & \boldsymbol{I} \\ -\boldsymbol{K}_P & -\boldsymbol{K}_D \end{bmatrix} \tag{C.20}$$

characterizing the error dynamics

$$\dot{\boldsymbol{e}} = \boldsymbol{A}\boldsymbol{e}. \tag{C.21}$$

If compensation is *imperfect*, then $\boldsymbol{z}(\cdot)$ cannot be neglected and the error equation in (C.18) takes on the general form

$$\dot{\boldsymbol{e}} = \boldsymbol{f}(\boldsymbol{e}). \tag{C.22}$$

It may be worth choosing the control law $\boldsymbol{w}(\boldsymbol{e})$ as the sum of a nonlinear term and a linear term of the kind in (C.19); in this case, the error equation can be written as

$$\dot{\boldsymbol{e}} = \boldsymbol{A}\boldsymbol{e} + \boldsymbol{k}(\boldsymbol{e}), \tag{C.23}$$

where $\boldsymbol{A}$ is given by (C.20) and $\boldsymbol{k}(\boldsymbol{e})$ is available to make the system globally asymptotically stable. The equations in (C.22), (C.23) express nonlinear differential equations of the error. To test for stability and obtain advise on the choice of suitable control actions, one may resort to *Lyapunov direct method* illustrated below.

C.3 Lyapunov Direct Method

The philosophy of the *Lyapunov direct method* is the same as that of most methods used in control engineering to study stability, namely, testing for stability without solving the differential equations describing the dynamic system.

This method can be presented in short on the basis of the following reasoning. If it is possible to associate an energy-based description with a (linear or nonlinear) autonomous dynamic system and, for each system state with the exception of the equilibrium state, the time rate of such energy is negative, then energy decreases along any system trajectory until it attains its minimum at the equilibrium state; this argument justifies an intuitive concept of stability.

With reference to (C.22), by setting $\boldsymbol{f}(\boldsymbol{0}) = \boldsymbol{0}$, the *equilibrium state* is $\boldsymbol{e} = \boldsymbol{0}$. A scalar function $V(\boldsymbol{e})$ of the system state, continuous together with its first derivative, is defined a *Lyapunov function* if the following properties hold:

$$V(\boldsymbol{e}) > 0 \qquad \forall \boldsymbol{e} \neq \boldsymbol{0}$$

$$\begin{aligned} &V(\boldsymbol{e}) = 0 && \boldsymbol{e} = \boldsymbol{0} \\ &\dot{V}(\boldsymbol{e}) < 0 && \forall \boldsymbol{e} \neq \boldsymbol{0} \\ &V(\boldsymbol{e}) \to \infty && \|\boldsymbol{e}\| \to \infty. \end{aligned}$$

The existence of such a function ensures *global asymptotic stability* of the equilibrium $\boldsymbol{e} = \boldsymbol{0}$. In practice, the equilibrium $\boldsymbol{e} = \boldsymbol{0}$ is globally asymptotically stable if a positive definite, radially unbounded function $\boldsymbol{V}(\boldsymbol{e})$ is found so that its time derivative along the system trajectories is negative definite.

If positive definiteness of $V(\boldsymbol{e})$ is realized by the adoption of a *quadratic form*, i.e.,

$$V(\boldsymbol{e}) = \boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} \tag{C.24}$$

with $\boldsymbol{Q}$ a symmetric positive definite matrix, then in view of (C.22) it follows

$$\dot{V}(\boldsymbol{e}) = 2\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{f}(\boldsymbol{e}). \tag{C.25}$$

If $\boldsymbol{f}(\boldsymbol{e})$ is so as to render the function $\dot{V}(\boldsymbol{e})$ negative definite, the function $V(\boldsymbol{e})$ is a *Lyapunov function*, since the choice (C.24) allows system global asymptotic stability to be proved. If $\dot{V}(\boldsymbol{e})$ in (C.25) is not negative definite for the given $V(\boldsymbol{e})$, nothing can be inferred on the stability of the system, since the Lyapunov method gives only a *sufficient* condition. In such cases one should resort to different choices of $V(\boldsymbol{e})$ in order to find, if possible, a negative definite $\dot{V}(\boldsymbol{e})$.

In the case when the property of negative definiteness does not hold, but $\dot{V}(\boldsymbol{e})$ is only *negative semi-definite*

$$\dot{V}(\boldsymbol{e}) \leq 0,$$

global asymptotic stability of the equilibrium state is ensured if the only system trajectory for which $\dot{V}(\boldsymbol{e})$ is *identically* null ($\dot{V}(\boldsymbol{e}) \equiv 0$) is the equilibrium trajectory $\boldsymbol{e} \equiv \boldsymbol{0}$ (a consequence of *La Salle theorem*).

Finally, consider the stability problem of the nonlinear system in the form (C.23); under the assumption that $\boldsymbol{k}(\boldsymbol{0}) = \boldsymbol{0}$, it is easy to verify that $\boldsymbol{e} = \boldsymbol{0}$ is an equilibrium state for the system. The choice of a Lyapunov function candidate as in (C.24) leads to the following expression for its derivative:

$$\dot{V}(\boldsymbol{e}) = \boldsymbol{e}^T(\boldsymbol{A}^T\boldsymbol{Q} + \boldsymbol{Q}\boldsymbol{A})\boldsymbol{e} + 2\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{k}(\boldsymbol{e}). \tag{C.26}$$

By setting

$$\boldsymbol{A}^T\boldsymbol{Q} + \boldsymbol{Q}\boldsymbol{A} = -\boldsymbol{P}, \tag{C.27}$$

the expression in (C.26) becomes

$$\dot{V}(\boldsymbol{e}) = -\boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e} + 2\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{k}(\boldsymbol{e}). \tag{C.28}$$

The matrix equation in (C.27) is said to be a *Lyapunov equation*; for any choice of a symmetric positive definite matrix $\boldsymbol{P}$, the solution matrix $\boldsymbol{Q}$ exists

and is symmetric positive definite if and only if the eigenvalues of $\boldsymbol{A}$ have all negative real parts. Since matrix $\boldsymbol{A}$ in (C.20) verifies such condition, it is always possible to assign a positive definite matrix $\boldsymbol{P}$ and find a positive definite matrix solution $\boldsymbol{Q}$ to (C.27). It follows that the first term on the right-hand side of (C.28) is negative definite and the stability problem is reduced to searching a control law so that $\boldsymbol{k}(\boldsymbol{e})$ renders the total $\dot{V}(\boldsymbol{e})$ negative (semi-)definite.

It should be underlined that La Salle theorem does not hold for *time-varying* systems (also termed *non-autonomous*) in the form

$$\dot{\boldsymbol{e}} = f(\boldsymbol{e}, t).$$

In this case, a conceptually analogous result which might be useful is the following, typically referred to as *Barbalat lemma* — of which it is indeed a consequence. Given a scalar function $V(\boldsymbol{e}, t)$ so that

1. $V(\boldsymbol{e}, t)$ is lower bounded
2. $\dot{V}(\boldsymbol{e}, t) \leq 0$
3. $\dot{V}(\boldsymbol{e}, t)$ is *uniformly continuous*

then it is $\lim_{t\to\infty} \dot{V}(\boldsymbol{e}, t) = 0$. Conditions 1 and 2 imply that $V(\boldsymbol{e}, t)$ has a bounded limit for $t \to \infty$. Since it is not easy to verify the property of uniform continuity from the definition, Condition 3 is usually replaced by

3'. $\ddot{V}(\boldsymbol{e}, t)$ is bounded

which is sufficient to guarantee validity of Condition 3. Barbalat lemma can obviously be used for time-invariant (autonomous) dynamic systems as an alternative to La Salle theorem, with respect to which some conditions are relaxed; in particular, $V(\boldsymbol{e})$ needs not necessarily be positive definite.

Bibliography

Linear systems analysis can be found in classical texts such as [61]. For the control of these systems see [82, 171]. For the analysis of nonlinear systems see [109]. Control of nonlinear mechanical systems is dealt with in [215].

D

Differential Geometry

The analysis of mechanical systems subject to nonholonomic constraints, such as wheeled mobile robots, requires some basic concepts of differential geometry and nonlinear controllability theory, that are briefly recalled in this appendix.

D.1 Vector Fields and Lie Brackets

For simplicity, the case of vectors $\boldsymbol{x} \in \mathbb{R}^n$ is considered. The tangent space at $\boldsymbol{x}$ (intuitively, the space of velocities of trajectories passing through $\boldsymbol{x}$) is hence denoted by $T_{\boldsymbol{x}}(\mathbb{R}^n)$. The presented notions are however valid in the more general case in which a *differentiable manifold* (i.e., a space that is locally diffeomorphic to $\mathbb{R}^n$) is considered in place of a Euclidean space.

A *vector field* $\boldsymbol{g} : \mathbb{R}^n \mapsto T_x(\mathbb{R}^n)$ is a mapping that assigns to each point $\boldsymbol{x} \in \mathbb{R}^n$ a tangent vector $\boldsymbol{g}(\boldsymbol{x}) \in T_x(\mathbb{R}^n)$. In the following it is always assumed that vector fields are *smooth*, i.e., such that the associated mappings are of class C^∞.

If the vector field $\boldsymbol{g}(\boldsymbol{x})$ is used to define a differential equation as in

$$\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}), \tag{D.1}$$

the *flow* $\boldsymbol{\phi}_t^{\boldsymbol{g}}(\boldsymbol{x})$ of $\boldsymbol{g}$ is the mapping that associates to each point $\boldsymbol{x}$ the value at time t of the solution of (D.1) evolving from $\boldsymbol{x}$ at time 0, or

$$\frac{d}{dt}\boldsymbol{\phi}_t^{\boldsymbol{g}}(\boldsymbol{x}) = \boldsymbol{g}(\phi_t^{\boldsymbol{g}}(\boldsymbol{x})). \tag{D.2}$$

The family of mappings $\{\boldsymbol{\phi}_t^{\boldsymbol{g}}\}$ is a one-parameter (i.e., t) group under the composition operator

$$\boldsymbol{\phi}_{t_1}^{\boldsymbol{g}} \circ \boldsymbol{\phi}_{t_2}^{\boldsymbol{g}} = \boldsymbol{\phi}_{t_1+t_2}^{\boldsymbol{g}}.$$

For example, for time-invariant linear systems it is $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$ and the flow is the linear operator $\boldsymbol{\phi}_t^{\boldsymbol{g}} = e^{\boldsymbol{A}t}$.

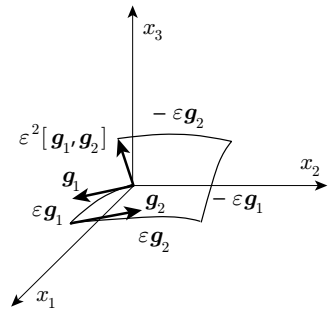

Fig. D.1. The net displacement of system (D.4) under the input sequence (D.5) is directed as the Lie bracket of the two vector fields $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$

Given two vector fields $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$, the composition of their flows is non-commutative in general:

$$\phi_t^{\boldsymbol{g}_1} \circ \phi_s^{\boldsymbol{g}_2} \neq \phi_s^{\boldsymbol{g}_2} \circ \phi_t^{\boldsymbol{g}_1}.$$

The vector field $[\boldsymbol{g}_1, \boldsymbol{g}_2]$ defined as

$$[\boldsymbol{g}_1, \boldsymbol{g}_2](\boldsymbol{x}) = \frac{\partial \boldsymbol{g}_2}{\partial x}\boldsymbol{g}_1(\boldsymbol{x}) - \frac{\partial \boldsymbol{g}_1}{\partial x}\boldsymbol{g}_2(\boldsymbol{x}) \tag{D.3}$$

is called *Lie bracket* of $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$. The two vector field $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ *commute* if $[\boldsymbol{g}_1, \boldsymbol{g}_2] = 0$.

The Lie bracket operation has an interesting interpretation. Consider the driftless dynamic system

$$\dot{\boldsymbol{x}} = \boldsymbol{g}_1(\boldsymbol{x})u_1 + \boldsymbol{g}_2(\boldsymbol{x})u_2 \tag{D.4}$$

associated with the vector fields $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$. If the inputs u_1 and u_2 are never active simultaneously, the solution of the differential equation (D.4) can be obtained by composing the flows of $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$. In particular, consider the following input sequence:

$$u(t) = \begin{cases} u_1(t) = +1, u_2(t) = 0 & t \in [0, \varepsilon) \\ u_1(t) = 0, u_2(t) = +1 & t \in [\varepsilon, 2\varepsilon) \\ u_1(t) = -1, u_2(t) = 0 & t \in [2\varepsilon, 3\varepsilon) \\ u_1(t) = 0, u_2(t) = -1 & t \in [3\varepsilon, 4\varepsilon), \end{cases} \tag{D.5}$$

where ε is an infinitesimal time interval. The solution of (D.4) at time $t = 4\varepsilon$ can be obtained by following first the flow of $\boldsymbol{g}_1$, then of $\boldsymbol{g}_2$, then of $-\boldsymbol{g}_1$, and finally of $-\boldsymbol{g}_2$ (see Fig. D.1). By computing $\boldsymbol{x}(\varepsilon)$ through a series expansion at $\boldsymbol{x}_0 = \boldsymbol{x}(0)$ along $\boldsymbol{g}_1$, then $\boldsymbol{x}(2\varepsilon)$ as a series expansion at $\boldsymbol{x}(\varepsilon)$ along $\boldsymbol{g}_2$, and so on, one obtains

$$\begin{aligned} \boldsymbol{x}(4\varepsilon) &= \phi_\varepsilon^{-\boldsymbol{g}_2} \circ \phi_\varepsilon^{-\boldsymbol{g}_1} \circ \phi_\varepsilon^{\boldsymbol{g}_2} \circ \phi_\varepsilon^{\boldsymbol{g}_1}(\boldsymbol{x}_0) \\ &= \boldsymbol{x}_0 + \varepsilon^2 \left(\frac{\partial \boldsymbol{g}_2}{\partial x}\, \boldsymbol{g}_1(\boldsymbol{x}_0) - \frac{\partial \boldsymbol{g}_1}{\partial x}\boldsymbol{g}_2(\boldsymbol{x}_0) \right) + O(\varepsilon^3). \end{aligned}$$

If $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ commute, the net displacement resulting from the input sequence (D.5) is zero.

The above expression shows that, at each point $\boldsymbol{x}$, infinitesimal motion of the driftless system (D.4) is possible not only in the directions belonging to the linear span of $\boldsymbol{g}_1(\boldsymbol{x})$ and $\boldsymbol{g}_2(\boldsymbol{x})$, but also in the direction of their Lie bracket $[\boldsymbol{g}_1, \boldsymbol{g}_2](\boldsymbol{x})$. It can be proven that more complicated input sequences can be used to generate motion in the direction of higher-order Lie brackets, such as $[\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]]$.

Similar constructive procedures can be given for systems with a *drift*[1] vector field, such as the following:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}_1(\boldsymbol{x})u_1 + \boldsymbol{g}_2(\boldsymbol{x})u_2. \tag{D.6}$$

Using appropriate input sequences, it is possible to generate motion in the direction of Lie brackets involving the vector field $\boldsymbol{f}$ as well as $\boldsymbol{g}_j$, $j = 1, 2$.

Example D.1

For a single-input linear system

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\,\boldsymbol{x} + \boldsymbol{b}\,u,$$

the drift and input vector fields are $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$ and $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{b}$, respectively. The following Lie brackets:

$$\begin{aligned} -[\boldsymbol{f}, \boldsymbol{g}] &= \boldsymbol{A}\boldsymbol{b} \\ [\boldsymbol{f}, [\boldsymbol{f}, \boldsymbol{g}]] &= \boldsymbol{A}^2\boldsymbol{b} \\ -[\boldsymbol{f}, [\boldsymbol{f}, [\boldsymbol{f}, \boldsymbol{g}]]] &= \boldsymbol{A}^3\boldsymbol{b} \\ &\vdots \end{aligned}$$

represent well-known directions in which it is possible to move the system.

The *Lie derivative* of the scalar function $\alpha : \mathbb{R}^n \mapsto \mathbb{R}$ along vector field $\boldsymbol{g}$ is defined as

$$L_{\boldsymbol{g}}\, \alpha(\boldsymbol{x}) = \frac{\partial \alpha}{\partial \boldsymbol{x}}\, \boldsymbol{g}(\boldsymbol{x}). \tag{D.7}$$

The following properties of Lie brackets are useful in computation:

$$\begin{aligned} &[\boldsymbol{f}, \boldsymbol{g}] = -[\boldsymbol{g}, \boldsymbol{f}] && \text{(skew-symmetry)} \\ &[\boldsymbol{f}, [\boldsymbol{g}, \boldsymbol{h}]] + [\boldsymbol{h}, [\boldsymbol{f}, \boldsymbol{g}]] + [\boldsymbol{g}, [\boldsymbol{h}, \boldsymbol{f}]] = 0 && \text{(Jacobi identity)} \\ &[\alpha\boldsymbol{f}, \beta\boldsymbol{g}] = \alpha\beta[\boldsymbol{f}, \boldsymbol{g}] + \alpha(L_{\boldsymbol{f}}\beta)\boldsymbol{g} - \beta(L_{\boldsymbol{g}}\alpha)\boldsymbol{f} && \text{(chain rule)} \end{aligned}$$

[1] This term emphasizes how the presence of $\boldsymbol{f}$ will in general force the system to move ($\dot{\boldsymbol{x}} \neq \boldsymbol{0}$) even in the absence of inputs.

with $\alpha, \beta: \mathbb{R}^n \mapsto \mathbb{R}$. The vector space $\mathcal{V}(\mathbb{R}^n)$ of smooth vector fields on $\mathbb{R}^n$, equipped with the Lie bracket operation, is a *Lie algebra*.

The *distribution* Δ associated with the m vector fields $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_m\}$ is the mapping that assigns to each point $\boldsymbol{x} \in \mathbb{R}^n$ the subspace of $T_x(\mathbb{R}^n)$ defined as

$$\Delta(\boldsymbol{x}) = \text{span}\{\boldsymbol{g}_1(\boldsymbol{x}), \ldots, \boldsymbol{g}_m(\boldsymbol{x})\}. \tag{D.8}$$

Often, a shorthand notation is used:

$$\Delta = \text{span}\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_m\}.$$

The distribution Δ is *nonsingular* if $\dim \Delta(\boldsymbol{x}) = r$, with r constant for all $\boldsymbol{x}$. In this case, r is called the *dimension* of the distribution. Moreover, Δ is called *involutive* if it is closed under the Lie bracket operation:

$$[\boldsymbol{g}_i, \boldsymbol{g}_j] \in \Delta \qquad \forall\, \boldsymbol{g}_i, \boldsymbol{g}_j \in \Delta.$$

The *involutive closure* $\bar{\Delta}$ of a distribution Δ is its closure under the Lie bracket operation. Hence, Δ is involutive if and only if $\bar{\Delta} = \Delta$. Note that the distribution $\Delta = \text{span}\{\boldsymbol{g}\}$ associated with a single vector field is always involutive, because $[\boldsymbol{g}, \boldsymbol{g}](\boldsymbol{x}) = \boldsymbol{0}$.

Example D.2

The distribution

$$\Delta = \text{span}\{\boldsymbol{g}_1, \boldsymbol{g}_2\} = \text{span}\left\{ \begin{bmatrix} \cos x_3 \\ \sin x_3 \\ 0 \end{bmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right\}$$

is nonsingular and has dimension 2. It is not involutive, because the Lie bracket

$$[\boldsymbol{g}_1, \boldsymbol{g}_2](\boldsymbol{x}) = \begin{bmatrix} \sin x_3 \\ -\cos x_3 \\ 0 \end{bmatrix}$$

is always linearly independent of $\boldsymbol{g}_1(\boldsymbol{x})$ and $\boldsymbol{g}_2(\boldsymbol{x})$. Its involutive closure is therefore

$$\bar{\Delta} = \text{span}\{\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]\}.$$

D.2 Nonlinear Controllability

Consider a nonlinear dynamic system of the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \sum_{j=1}^{m} \boldsymbol{g}_j(\boldsymbol{x}) u_j, \tag{D.9}$$

that is called *affine* in the inputs u_j. The state $\boldsymbol{x}$ takes values in $\mathbb{R}^n$, while each component u_j of the control input $\boldsymbol{u} \in \mathbb{R}^m$ takes values in the class $\mathcal{U}$ of piecewise-constant functions.

Denote by $\boldsymbol{x}(t, 0, \boldsymbol{x}_0, \boldsymbol{u})$ the solution of (D.9) at time $t \geq 0$, corresponding to an input $\boldsymbol{u}: [0, t] \to \mathcal{U}$ and an initial condition $\boldsymbol{x}(0) = \boldsymbol{x}_0$. Such a solution exists and is unique provided that the drift vector field $\boldsymbol{f}$ and the input vector fields $\boldsymbol{g}_j$ are of class C^∞. System (D.9) is said to be *controllable* if, for any choice of $\boldsymbol{x}_1, \boldsymbol{x}_2$ in $\mathbb{R}^n$, there exists a time instant T and an input $\boldsymbol{u}: [0, T] \to \mathcal{U}$ such that $\boldsymbol{x}(T, 0, \boldsymbol{x}_1, \boldsymbol{u}) = \boldsymbol{x}_2$.

The *accessibility algebra* $\mathcal{A}$ of system (D.9) is the smallest subalgebra of $\mathcal{V}(\mathbb{R}^n)$ that contains $\boldsymbol{f}, \boldsymbol{g}_1, \ldots, \boldsymbol{g}_m$. By definition, all the Lie brackets that can be generated using these vector fields belong to $\mathcal{A}$. The *accessibility distribution* $\Delta_{\mathcal{A}}$ of system (D.9) is defined as

$$\Delta_{\mathcal{A}} = \text{span}\{\boldsymbol{v} | \boldsymbol{v} \in \mathcal{A}\}. \tag{D.10}$$

In other words, $\Delta_{\mathcal{A}}$ is the involutive closure of $\Delta = \text{span}\{\boldsymbol{f}, \boldsymbol{g}_1, \ldots, \boldsymbol{g}_m\}$.

The computation of $\Delta_{\mathcal{A}}$ may be organized as an iterative procedure

$$\Delta_{\mathcal{A}} = \text{span}\left\{\boldsymbol{v} | \boldsymbol{v} \in \Delta_i, \forall i \geq 1\right\},$$

with

$$\begin{aligned} \Delta_1 &= \Delta = \text{span}\{\boldsymbol{f}, \boldsymbol{g}_1, \ldots, \boldsymbol{g}_m\} \\ \Delta_i &= \Delta_{i-1} + \text{span}\{[\boldsymbol{g}, \boldsymbol{v}] |\, \boldsymbol{g} \in \Delta_1, \boldsymbol{v} \in \Delta_{i-1}\}, \quad i \geq 2. \end{aligned}$$

This procedure stops after κ steps, where κ is the smallest integer such that $\Delta_{\kappa+1} = \Delta_\kappa = \Delta_{\mathcal{A}}$. This number is called the *nonholonomy degree* of the system and is related to the 'level' of Lie brackets that must be included in $\Delta_{\mathcal{A}}$. Since $\dim \Delta_{\mathcal{A}} \leq n$, it is $\kappa \leq n - m$ necessarily.

If system (D.9) is driftless

$$\dot{\boldsymbol{x}} = \sum_{i=1}^{m} \boldsymbol{g}_i(\boldsymbol{x}) u_i, \tag{D.11}$$

the accessibility distribution $\Delta_{\mathcal{A}}$ associated with vector fields $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_m$ characterizes its controllability. In particular, system (D.11) is controllable if and only if the following *accessibility rank condition* holds:

$$\dim \Delta_{\mathcal{A}}(\boldsymbol{x}) = n. \tag{D.12}$$

Note that for driftless systems the iterative procedure for building $\Delta_{\mathcal{A}}$ starts with $\Delta_1 = \Delta = \text{span}\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_m\}$, and therefore $\kappa \leq n - m + 1$.

For systems in the general form (D.9), condition (D.12) is only necessary for controllability. There are, however, two notable exceptions:

- If system (D.11) is controllable, the system with drift obtained by performing a *dynamic extension* of (D.11)

$$\dot{\boldsymbol{x}} = \sum_{i=1}^{m} \boldsymbol{g}_i(\boldsymbol{x}) v_i \tag{D.13}$$

$$\dot{v}_i = u_i, \qquad i = 1, \ldots, m, \tag{D.14}$$

  i.e., by adding an integrator on each input channel, is also controllable.
- For a linear system

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \sum_{j=1}^{m} \boldsymbol{b}_j u_j = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}$$

  (D.12) becomes

$$\varrho\left([\,\boldsymbol{B} \quad \boldsymbol{A}\boldsymbol{B} \quad \boldsymbol{A}^2\boldsymbol{B} \quad \ldots \quad \boldsymbol{A}^{n-1}\boldsymbol{B}\,]\right) = n, \tag{D.15}$$

  i.e., the well-known necessary and sufficient condition for controllability due to Kalman.

Bibliography

The concepts briefly recalled in this appendix can be studied in detail in various tests of differential geometry [94, 20] and nonlinear control theory [104, 168, 195].

Index