

Robot Kinematics

Bruno Siciliano

PRISMA Lab
Dipartimento di Informatica e Sistemistica
Università degli Studi di Napoli Federico II
Via Claudio 21, 80125 Napoli, Italy
`siciliano@unina.it`

Subsection for the CRC Press Handbook

Mechanical Systems Design

(Editors: Osita D.I. Nwokah and Y. Hurmuzlu)

Section: Robotics

(Editorial Advisory Board Member: Miomir Vukobratović)

1 Introduction

From a mechanical viewpoint, a robotic system is in general constituted by a locomotion apparatus (legs, wheels) to move in the environment and by a manipulation apparatus to operate on the objects present in the environment. It is then important to distinguish between the class of mobile robots and the class of robot manipulators.

The mechanical structure of a robot manipulator consists of a sequence of links connected by means of joints. Links and joints are usually made as rigid as possible so as to achieve high precision in robot positioning. The presence of elasticity at the joint transmission or the use of lightweight materials for the links poses a number of interesting issues which lead to separating the study of flexible robot manipulators from that of rigid robot manipulators; the latter are implicitly meant here by using the term “robots” all throughout.

This contribution is aimed at surveying the fundamentals of robot kinematics. Basic mathematical tools such the rotation matrix, the unit quaternion and the Euler angles are briefly recalled. They serve to describe the orientation of the robot’s end effector which, together with the position, can be expressed as a function of the joint variables. This is the direct kinematics equation which is derived through a systematic procedure based on the use of homogeneous transformations and the so-called Denavit-Hartenberg convention. The inverse kinematics problem is considered and closed-form solutions are found for simple geometries. Further, a treatment of differential kinematics is provided that is based on the robot’s Jacobian (geometric or analytical). Specific attention is paid to the occurrence of singularities or redundancy in the context of the differential kinematics inversion. The material ends with the presentation of inverse kinematics algorithms with special concern to the definition of the end-effector orientation error; both a pseudoinverse and a transpose of the Jacobian are considered.

2 Description of Orientation

Robot manipulation tasks are typically specified in terms of the position and orientation of an end-effector frame with respect to a base frame. Position is uniquely described by the Cartesian coordinates of the origin of the end-effector frame, whereas various representations of orientation exist. Therefore, as a natural prelude to deriving the direct kinematics equation of a robot, some basic concepts about the orientation of a rigid body in space are briefly recalled in the sequel.

2.1 Rotation Matrix

The location of a rigid body in space is typically described in terms of the (3×1) *position vector* \mathbf{p} and the (3×3) *rotation matrix* \mathbf{R} describing the origin and the orientation of a frame attached to the body with respect to a fixed reference frame, i.e.

$$\mathbf{R} = [\mathbf{x} \quad \mathbf{y} \quad \mathbf{z}] \quad (1)$$

where $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are the unit vectors expressing the direction cosines of the axes of the body frame with respect to the reference frame. It is straightforward to verify that the matrix \mathbf{R} is orthogonal, meaning that

$$\mathbf{R}^T \mathbf{R} = \mathbf{I} \quad (2)$$

and thus implying the useful result that the transpose of a rotation matrix is equal to its inverse, i.e. $\mathbf{R}^T = \mathbf{R}^{-1}$. Frame orientation is conventionally taken to be left-handed.

A rotation matrix possesses three equivalent geometrical meanings:

- It describes the mutual orientation between two coordinate frames (as above).
- 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 rotating a vector in the same coordinate frame.

Elementary rotations are those made about one of the coordinate axes, i.e.

$$\mathbf{R}_X(\alpha) = \begin{bmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

$$\mathbf{R}_Y(\beta) = \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \quad (4)$$

$$\mathbf{R}_Z(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \gamma & -\sin \gamma \\ 0 & \sin \gamma & \cos \gamma \end{bmatrix} \quad (5)$$

denote the *elementary rotation matrices* with respect to X, Y, Z axes. These are useful to describe rotations about an arbitrary axis in space, as shown below.

Rotation matrices between multiple frames —say frames 0, 1, 2— can be nicely composed according to the simple rule

$${}^0\mathbf{R}_2 = {}^0\mathbf{R}_1 {}^1\mathbf{R}_2 \quad (6)$$

where the notation ${}^j\mathbf{R}_i$ denotes the rotation matrix of frame i with respect to frame j , and successive rotations are composed with respect to the axes of the current frame. Note also that ${}^i\mathbf{R}_j = ({}^j\mathbf{R}_i)^T$.

It is often desired to express a rotation of a given angle about an arbitrary axis in space. Let $\mathbf{r} = [r_x \ r_y \ r_z]^T$ be the unit vector of a rotation axis with respect to the reference frame. In order to derive the rotation matrix $\mathbf{R}(\vartheta, \mathbf{r})$ expressing the rotation of an angle ϑ about axis \mathbf{r} , it is convenient to compose 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 \mathbf{r} .

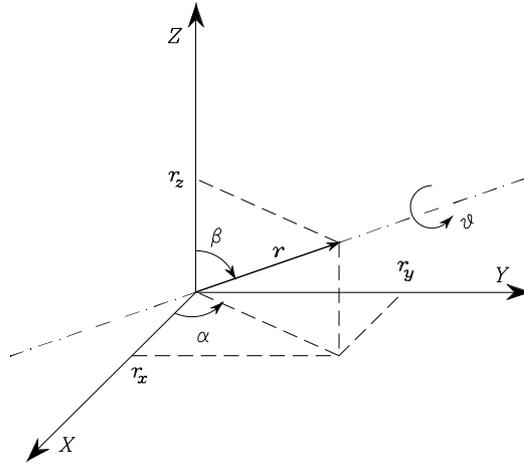


Figure 1: Rotation of a given angle about an arbitrary axis.

As shown in Fig. 1, a possible solution is obtained through the following sequence of rotations:

- align Z with \mathbf{r} , which is obtained as the sequence of a rotation by α about Z and a rotation by β about Y ;
- rotate by ϑ about Z ;
- realign with the initial direction of Z , which is obtained as the sequence of a rotation by $-\beta$ about Y and a rotation by $-\alpha$ about Z .

The resulting rotation matrix is

$$\mathbf{R}(\vartheta, \mathbf{r}) = \mathbf{R}_Z(\alpha)\mathbf{R}_Y(\beta)\mathbf{R}_Z(\vartheta)\mathbf{R}_Y(-\beta)\mathbf{R}_Z(-\alpha). \quad (7)$$

By using the following relations:

$$\begin{aligned}\sin \alpha &= \frac{r_y}{\sqrt{r_x^2 + r_y^2}} & \cos \alpha &= \frac{r_x}{\sqrt{r_x^2 + r_y^2}} \\ \sin \beta &= \sqrt{r_x^2 + r_y^2} & \cos \beta &= r_z.\end{aligned}$$

the rotation matrix of the *angle/axis description* in (7) can be expressed as

$$\mathbf{R}(\vartheta, \mathbf{r}) = \begin{bmatrix} r_x^2(1 - c_\vartheta) + c_\vartheta & r_x r_y(1 - c_\vartheta) - r_z s_\vartheta & r_x r_z(1 - c_\vartheta) + r_y s_\vartheta \\ r_x r_y(1 - c_\vartheta) + r_z s_\vartheta & r_y^2(1 - c_\vartheta) + c_\vartheta & r_y r_z(1 - c_\vartheta) - r_x s_\vartheta \\ r_x r_z(1 - c_\vartheta) - r_y s_\vartheta & r_y r_z(1 - c_\vartheta) + r_x s_\vartheta & r_z^2(1 - c_\vartheta) + c_\vartheta \end{bmatrix} \quad (8)$$

where standard abbreviation for $\cos \vartheta$ and $\sin \vartheta$ have been used. Equation (8) can be cast in the more compact form

$$\mathbf{R}(\vartheta, \mathbf{r}) = c_\vartheta \mathbf{I} + (1 - c_\vartheta) \mathbf{r} \mathbf{r}^\top - s_\vartheta \mathbf{S}(\mathbf{r}) \quad (9)$$

where \mathbf{I} is the (3×3) identity matrix and $\mathbf{S}(\cdot)$ is the matrix operator performing the cross product between two (3×1) vectors, i.e. $\mathbf{S}(\mathbf{a})\mathbf{b} = \mathbf{a} \times \mathbf{b}$

Although the axis can be arbitrary, the three components of \mathbf{r} are constrained by the unit norm condition

$$\mathbf{r}^\top \mathbf{r} = 1. \quad (10)$$

Also, it is clear that $\mathbf{R}(-\vartheta, -\mathbf{r}) = \mathbf{R}(\vartheta, \mathbf{r})$, i.e. a rotation by $-\vartheta$ about $-\mathbf{r}$ cannot be distinguished from a rotation by ϑ about \mathbf{r} ; hence, for $\vartheta = \pi$ the representation is not unique.

The angle and axis corresponding to a given rotation matrix

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (11)$$

are:

$$\begin{aligned}\vartheta &= \cos^{-1} \left(\frac{r_{11} + r_{22} + r_{33} - 1}{2} \right) \\ \mathbf{r} &= \frac{1}{2} \sin \vartheta \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}\end{aligned} \quad (12)$$

for $\sin \vartheta \neq 0$. Instead, if $\sin \vartheta = 0$, then it is necessary to refer directly to the particular expressions attained by \mathbf{R} and find the solving formulæ in the two cases: if $\vartheta = 0$ the unit vector is arbitrary (no rotation has occurred), while if $\vartheta = \pi$, the above nonuniqueness problem is encountered. This drawback can be overcome by adopting a different four-parameter description; namely, the unit quaternion which is introduced next.

2.2 Unit Quaternion

With reference to the above angle/axis description of orientation, the *unit quaternion* (viz. Euler parameters) is defined as:

$$\mathcal{Q} = \{\eta, \boldsymbol{\varepsilon}\} \quad (13)$$

where

$$\begin{aligned} \eta &= \cos \frac{\vartheta}{2} \\ \boldsymbol{\varepsilon} &= \sin \frac{\vartheta}{2} \mathbf{r}, \end{aligned} \quad (14)$$

with $\eta \geq 0$ for $\vartheta \in [-\pi, \pi]$; η is called the scalar part of the quaternion while $\boldsymbol{\varepsilon}$ is called the vector part of the quaternion.

The constraint (10) transforms into

$$\eta^2 + \boldsymbol{\varepsilon}^T \boldsymbol{\varepsilon} = 1. \quad (15)$$

It is worth remarking that, differently from the angle/axis description, a rotation by $-\vartheta$ about $-\mathbf{r}$ gives a vector part of the quaternion of opposite sign from the one associated with a rotation by ϑ about \mathbf{r} , while the scalar part does not change; this solves the above nonuniqueness problem. The rotation matrix corresponding to a given quaternion is

$$\mathbf{R}(\eta, \boldsymbol{\varepsilon}) = (\eta^2 - \boldsymbol{\varepsilon}^T \boldsymbol{\varepsilon}) \mathbf{I} + 2\boldsymbol{\varepsilon} \boldsymbol{\varepsilon}^T - 2\eta \mathbf{S}(\boldsymbol{\varepsilon}). \quad (16)$$

On the other hand, the unit quaternion corresponding to a given rotation matrix (11) is

$$\begin{aligned} \eta &= \frac{1}{2} \sqrt{r_{11} + r_{22} + r_{33} + 1} \\ \boldsymbol{\varepsilon} &= \begin{bmatrix} \frac{1}{2} \text{sgn}(r_{32} - r_{23}) \sqrt{r_{11} - r_{22} - r_{33} + 1} \\ \frac{1}{2} \text{sgn}(r_{13} - r_{31}) \sqrt{r_{22} - r_{33} - r_{11} + 1} \\ \frac{1}{2} \text{sgn}(r_{21} - r_{12}) \sqrt{r_{33} - r_{11} - r_{22} + 1} \end{bmatrix}. \end{aligned} \quad (17)$$

2.3 Euler Angles

Rotation matrices in general 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 in (2). Even in the case of describing orientation in terms of rotation about an arbitrary axis, or else a unit quaternion, a representation in terms of four parameters is obtained. These components are not independent but are constrained by

the either condition (10) or condition (15). This implies that the actual free parameters to describe orientation are *three*.

A minimal representation of orientation can be obtained by using a set of three *Euler angles* $\varphi = [\alpha \ \beta \ \gamma]^T$. Among the 12 possible definitions of Euler angles, without loss of generality, the *XYZ* representation is considered leading to the rotation matrix

$$\begin{aligned} \mathbf{R}(\varphi) &= \mathbf{R}_X(\alpha)\mathbf{R}_Y(\beta)\mathbf{R}_Z(\gamma) \\ &= \begin{bmatrix} c_\beta c_\gamma & -c_\beta s_\gamma & s_\beta \\ s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma & -s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma & -s_\alpha c_\beta \\ -c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma & c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma & c_\alpha c_\beta \end{bmatrix}. \end{aligned} \quad (18)$$

The set of the Euler angles corresponding to a given rotation matrix (11) is

$$\begin{aligned} \alpha &= \text{Atan2}(-r_{23}, r_{33}) \\ \beta &= \text{Atan2}\left(r_{13}, \sqrt{r_{11}^2 + r_{12}^2}\right) \\ \gamma &= \text{Atan2}(-r_{12}, r_{11}) \end{aligned} \quad (19)$$

with $\beta \in (-\pi/2, \pi/2)$, whereas the solution is

$$\begin{aligned} \alpha &= \text{Atan2}(r_{23}, -r_{33}) \\ \beta &= \text{Atan2}\left(r_{13}, -\sqrt{r_{11}^2 + r_{12}^2}\right) \\ \gamma &= \text{Atan2}(r_{12}, -r_{11}) \end{aligned} \quad (20)$$

with $\beta \in (\pi/2, 3\pi/2)$; 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.

Solutions (19) and (20) degenerate when $\beta = \pm\pi/2$; in this case, it is possible to determine only the sum or difference of α and γ , i.e.

$$\alpha \pm \gamma = \text{Atan2}(r_{21}, r_{22}) \quad (21)$$

where the plus sign applies for $\beta = +\pi/2$ and the minus sign applies for $\beta = -\pi/2$.

3 Direct Kinematics

A robot manipulator consists of a kinematic chain of $n + 1$ links connected by means of n joints. Joints can essentially be of two types: *revolute* and *prismatic*; complex joints can be decomposed into these simple joints. Revolute joints are usually preferred to prismatic joints in view of their compactness and reliability. One end of the chain is connected to the base link to which a suitable

base frame is attached, whereas an *end effector* is connected to the other end and a suitable end-effector frame is attached. The basic structure of a robot is the open kinematic chain which occurs when there is only one sequence of links connecting the two ends of the chain. Alternatively, a robot contains a closed kinematic chain when a sequence of links forms a loop. In Fig. 2, an open-chain robot manipulator is illustrated with conventional representation of revolute and prismatic joints.

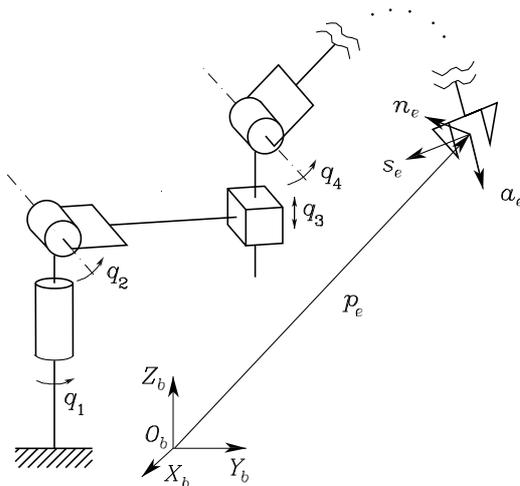


Figure 2: Schematic of an open-chain robot manipulator with base frame and end-effector frame.

Direct kinematics of a robot consists of determining the mapping between the joint variables and the position and orientation of the end-effector frame with respect to the base frame.

3.1 Homogeneous Transformation

As discussed above, 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).

The complete *coordinate transformation* between two frames—say frames 0, 1—is given by composing the translation ${}^0\mathbf{p}_1$ between the origins of the frames and the rotation ${}^0\mathbf{R}_1$ between the axes of the frames into a (4×4) *homogeneous*

transformation matrix

$${}^0\mathbf{T}_1 = \begin{bmatrix} & {}^0\mathbf{R}_1 & & {}^0\mathbf{p}_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (22)$$

Similarly to the composition of rotations expressed by (6), a sequence of coordinate transformations from frame 0 to frame n can be composed as in the product

$${}^0\mathbf{T}_n = {}^0\mathbf{T}_1 {}^1\mathbf{T}_2 \dots {}^{n-1}\mathbf{T}_n \quad (23)$$

where ${}^{i-1}\mathbf{T}_i$ denotes the homogeneous transformation expressing the position and orientation of frame i with respect to frame $i - 1$. The relationship (23) is the basic tool to deriving the direct kinematics equation of a robot.

3.2 Denavit-Hartenberg Convention

An effective procedure for computing the direct kinematics function for a general robot is based on the so-called modified *Denavit-Hartenberg* convention. According to this convention, a coordinate frame is attached to each link of the chain and the overall transformation matrix from link 0 to link n is derived by composition of transformations between consecutive frames. With reference to Fig. 3, let joint i connect link $i - 1$ to link i , where the links are assumed to be rigid; frame i is attached to link i and can be defined as follows.

- Choose axis Z_i aligned with the axis of joint i .
- Choose axis X_i along the common normal to axes Z_i and Z_{i+1} with direction from joint i to joint $i + 1$.
- Choose axis Y_i so as to complete a right-handed frame.

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 *kinematic parameters*:

α_i angle between Z_{i-1} and Z_i about X_{i-1} measured counter-clockwise,

ℓ_i distance between Z_{i-1} and Z_i along X_{i-1} ,

ϑ_i angle between X_{i-1} and X_i about Z_i measured counter-clockwise,

d_i distance between X_{i-1} and X_i along Z_i .

Let $\mathbf{Rot}(K, \delta)$ ($\mathbf{Trans}(K, \delta)$) denote the homogeneous transformation matrix expressing the rotation (translation) about (along) axis K by an angle

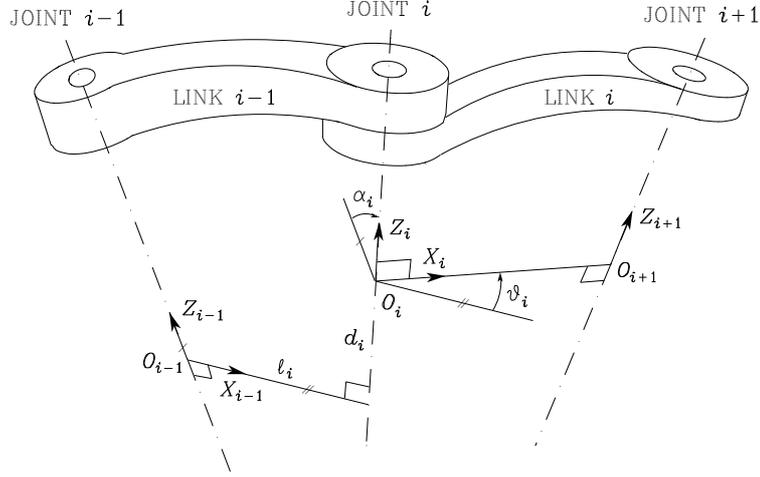


Figure 3: Kinematic parameters with modified Denavit-Hartenberg convention.

(distance) δ . Then, the coordinate transformation of frame i with respect to frame $i-1$ can be expressed in terms of the above four parameters by the matrix

$$\begin{aligned}
 {}^{i-1}\mathbf{T}_i &= \mathbf{Rot}(X, \alpha_i) \mathbf{Trans}(X, l_i) \mathbf{Rot}(Z, \vartheta_i) \mathbf{Trans}(Z, d_i) \\
 &= \begin{bmatrix} \cos \vartheta_i & -\sin \vartheta_i & 0 & l_i \\ \cos \alpha_i \sin \vartheta_i & \cos \alpha_i \cos \vartheta_i & -\sin \alpha_i & -d_i \sin \alpha_i \\ \sin \alpha_i \sin \vartheta_i & \sin \alpha_i \cos \vartheta_i & \cos \alpha_i & d_i \cos \alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} {}^{i-1}\mathbf{R}_i & {}^{i-1}\mathbf{p}_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{24}
 \end{aligned}$$

where ${}^{i-1}\mathbf{R}_i$ is the (3×3) matrix defining the orientation of frame i with respect to frame $i-1$, and ${}^{i-1}\mathbf{p}_i$ is the (3×1) vector defining the origin of frame i with respect to frame $i-1$.

Dually, the transformation matrix defining frame $i-1$ with respect to frame i is given by

$${}^i\mathbf{T}_{i-1} = \mathbf{Trans}(Z, -d_i) \mathbf{Rot}(Z, -\vartheta_i) \mathbf{Trans}(X, -l_i) \mathbf{Rot}(X, -\alpha_i)$$

$$= \begin{bmatrix} & & & -\ell_i \cos \vartheta_i \\ & {}^{i-1}\mathbf{R}_i^T & & \ell_i \sin \vartheta_i \\ 0 & 0 & 0 & -d_i \\ & & & 1 \end{bmatrix}. \quad (25)$$

Two of the four parameters (ℓ_i and α_i) are always constant and depend only on the size and shape of link i . Of the remaining two parameters, only one is variable (*degree of freedom*) depending on the type of joint that connects link $i - 1$ to link i . If q_i denotes joint i variable, then it is

$$q_i = \bar{\xi}_i \vartheta_i + \xi_i d_i \quad (26)$$

where $\bar{\xi}_i = 1 - \xi_i$, i.e.

- $\xi_i = 0$ if joint i is *revolute* ($q_i = \vartheta_i$),
- $\xi_i = 1$ if joint i is *prismatic* ($q_i = d_i$).

In view of (26), the equation

$$\bar{q}_i = \xi_i \vartheta_i + \bar{\xi}_i d_i \quad (27)$$

gives the constant parameter at each joint to add to α_i and ℓ_i .

The above procedure does not yield a unique definition of frames 0 and n which can be chosen arbitrarily. Also, in all cases of nonuniqueness in the definition of the frames, it is convenient to make as many link parameters zero as possible, since this will simplify kinematics computation. A number of remarks are in order.

- A simple choice to define frame 0 is to take it coincident with frame 1 when $q_1 = 0$; this makes $\alpha_1 = 0$ and $\ell_1 = 0$, and $\bar{q}_1 = 0$.
- A similar choice for frame n is to take X_n along X_{n-1} when $q_n = 0$; this makes $\bar{q}_n = 0$.
- If joint i is prismatic, the direction of Z_i is fixed while its location is arbitrary; it is convenient to locate Z_i either at the origin of frame $i - 1$ ($\ell_i = 0$) or at the origin of frame $i + 1$ ($\ell_{i+1} = 0$).
- When the joint axes i and $i + 1$ are parallel, it is convenient to locate X_i so as to achieve either $d_i = 0$ or $d_{i+1} = 0$ if either joint is revolute.

In view of (23), by composition of the individual link transformations, the coordinate transformation describing the position and orientation of frame n with respect to frame 0 is given by

$${}^0\mathbf{T}_n(\mathbf{q}) = {}^0\mathbf{T}_1(q_1) {}^1\mathbf{T}_2(q_2) \dots {}^{n-1}\mathbf{T}_n(q_n), \quad (28)$$

where \mathbf{q} denotes the $(n \times 1)$ vector of joint variables. In order to derive the direct kinematics, two further *constant* transformations have to be introduced; namely, the transformation from the base frame b to frame 0 (bT_0) and the transformation from frame n to the end-effector frame e (nT_e), i.e.

$$\begin{aligned} {}^bT_e(\mathbf{q}) &= {}^bT_0 {}^0T_n(\mathbf{q}) {}^nT_e \\ &= \begin{bmatrix} {}^b\mathbf{n}_e(\mathbf{q}) & {}^b\mathbf{s}_e(\mathbf{q}) & {}^b\mathbf{a}_e(\mathbf{q}) & {}^b\mathbf{p}_e(\mathbf{q}) \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (29)$$

where the normal, sliding and approach unit vectors $\mathbf{n}, \mathbf{s}, \mathbf{a}$ have been formally introduced (Fig. 2). Subscripts and superscripts can be omitted when the relevant frames are clear from the context.

The terminology “modified” Denavit-Hartenberg convention stems from the fact that, in the “classical” convention, axis Z_i is aligned with the axis of joint $i + 1$ and the kinematic parameters differ accordingly.

As an example of open-chain robot, consider the *anthropomorphic robot*. With reference to the frames illustrated in Fig. 4, the Denavit-Hartenberg parameters are specified in Tab. 1.

i	α_i	ℓ_i	ϑ_i	d_i
1	0	0	q_1	0
2	$\pi/2$	0	q_2	0
3	0	ℓ_3	q_3	0
4	$-\pi/2$	0	q_4	d_4
5	$\pi/2$	0	q_5	0
6	$-\pi/2$	0	q_6	0

Table 1: Denavit-Hartenberg parameters of the anthropomorphic robot.

Computing the transformation matrices in (24) and composing them as in (28) gives

$${}^0T_6 = \begin{pmatrix} {}^0\mathbf{n}_6 & {}^0\mathbf{s}_6 & {}^0\mathbf{a}_6 & {}^0\mathbf{p}_6 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (30)$$

where

$${}^0\mathbf{p}_6 = \begin{bmatrix} c_1(c_2\ell_3 - s_{23}d_4) \\ s_1(c_2\ell_3 - s_{23}d_4) \\ s_2\ell_3 + c_{23}d_4 \end{bmatrix} \quad (31)$$

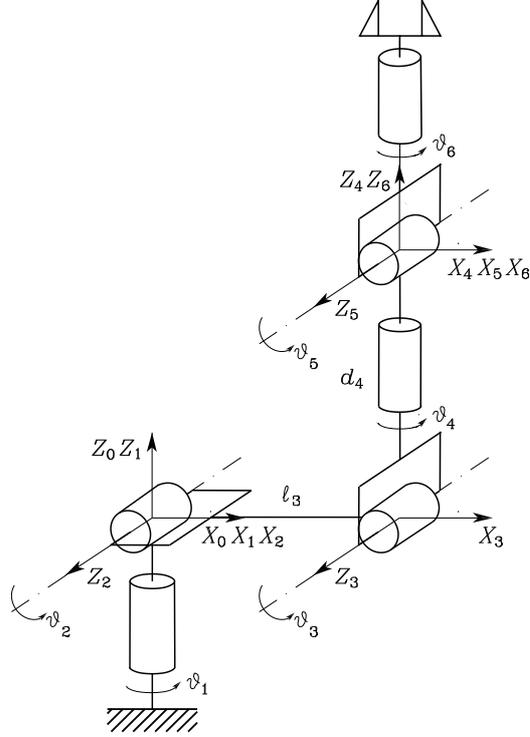


Figure 4: Anthropomorphic robot with frame assignment.

for the position, and

$${}^0\mathbf{n}_6 = \begin{bmatrix} c_1(c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6) - s_1(s_4c_5c_6 + c_4s_6) \\ s_1(c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6) + c_1(s_4c_5c_6 + c_4s_6) \\ s_{23}(c_4c_5c_6 - s_4s_6) + c_{23}s_5c_6 \end{bmatrix} \quad (32)$$

$${}^0\mathbf{s}_6 = \begin{bmatrix} c_1(-c_{23}(c_4c_5s_6 + s_4c_6) + s_{23}s_5s_6) + s_1(s_4c_5s_6 - c_4c_6) \\ s_1(-c_{23}(c_4c_5s_6 + s_4c_6) + s_{23}s_5s_6) - c_1(s_4c_5s_6 - c_4c_6) \\ -s_{23}(c_4c_5s_6 + s_4c_6) - c_{23}s_5s_6 \end{bmatrix} \quad (33)$$

$${}^0\mathbf{a}_6 = \begin{bmatrix} -c_1(c_{23}c_4s_5 + s_{23}c_5) + s_1s_4s_5 \\ -s_1(c_{23}c_4s_5 + s_{23}c_5) - c_1s_4s_5 \\ -s_{23}c_4s_5 + c_{23}c_5 \end{bmatrix} \quad (34)$$

for the orientation, where $c_i = \cos \vartheta_i$, $s_i = \sin \vartheta_i$, $c_{23} = \cos(\vartheta_2 + \vartheta_3)$ and $s_{23} = \sin(\vartheta_2 + \vartheta_3)$.

3.3 Joint Space and Task Space

If a task has to be assigned to the end effector, it is necessary to specify both end-effector position and orientation. This is easy for the position \mathbf{p}_e . Instead, specifying the orientation through the unit vector triple $(\mathbf{n}_e, \mathbf{s}_e, \mathbf{a}_e)$ is difficult, since their nine components must be guaranteed to satisfy the orthonormality constraints imposed by (2). Even with a four-parameter description of orientation, still one constraint in the form either (10) or (15) should be satisfied.

On the other hand, if a minimal representation is adopted in terms of the Euler angles describing the orientation of the end-effector frame with respect to the base frame, a suitable $(m \times 1)$ vector can be considered as

$$\mathbf{x} = \begin{bmatrix} \mathbf{p}_e \\ \boldsymbol{\varphi}_e \end{bmatrix}, \quad (35)$$

where \mathbf{p}_e describes the end-effector position and $\boldsymbol{\varphi}_e$ its orientation. This representation of position and orientation allows the description of the end-effector task in terms of a number of inherently independent parameters. The vector \mathbf{x} is defined in the space in which the robot task is specified; hence, this space is typically called *task space* (operational space). The dimension of the task space is at most $m = 6$, since 3 coordinates specify position and 3 angles specify orientation. Nevertheless, depending on the geometry of the task, a reduced number of task space variables may be specified; for instance, for a planar robot it is $m = 3$, since two coordinates specify position and one angle specifies orientation.

On the other hand, the *joint space* (configuration space) denotes the space in which the $(n \times 1)$ vector of joint variables \mathbf{q} is defined. Accounting for the dependence of position and orientation from the joint variables, the direct kinematics equation can be written in a form other than (24), i.e.

$$\mathbf{x} = \mathbf{k}(\mathbf{q}). \quad (36)$$

It is worth noticing that the explicit dependence of the function $\mathbf{k}(\mathbf{q})$ from the joint variables for the orientation components is not available except for simple cases. In fact, on the most general assumption of a six-dimensional task space ($m = 6$), the computation of the three components of the function $\boldsymbol{\varphi}_e(\mathbf{q})$ cannot be performed in closed form but goes through the computation of the elements of the rotation matrix.

The notion of joint space and task space naturally allows introducing the concept of *kinematic redundancy*. This occurs when the dimension of the task space is smaller than the dimension of the joint space ($m < n$). Redundancy is, anyhow, a concept *relative* to the task assigned to the robot; a robot can be redundant with respect to a task and nonredundant with respect to another, depending on the number of task space variables of interest.

For instance, a three-degree-of-freedom planar robot becomes redundant if end-effector orientation is of no concern ($m = 2, n = 3$). Yet, the typical

example of redundant robot is constituted by the human arm that has seven degrees of freedom: three in the shoulder, one in the elbow and three in the wrist, without considering the degrees of freedom in the fingers ($m = 6$, $n = 7$).

4 Inverse Kinematics

The direct kinematics equation, either in the form (24) or in the form (36), establishes the functional relationship between the joint variables and the end-effector position and orientation. *Inverse kinematics* concerns the determination of the joint variables \mathbf{q} corresponding to a given end-effector position \mathbf{p}_e and orientation \mathbf{R}_e . The solution to this problem is of fundamental importance in order to translate the specified motion, naturally assigned in the task space, into the equivalent joint space motion that allows execution of the desired task.

With regard to the direct kinematics equation (24), the end-effector position and rotation matrix are uniquely computed, once the joint variables are known. In general, this cannot be said for eq. (36) too, since the Euler angles are not uniquely defined. On the other hand, the inverse kinematics problem is much more complex for the following reasons.

- The equations to solve are in general nonlinear equations for which it is not always possible to find closed-form solutions.
- Multiple solutions may exist.
- Infinite solutions may exist, e.g. in the case of a kinematically redundant robot.
- There might not be admissible solutions, in view of the robot kinematic structure.

For what concerns existence of solutions, this is guaranteed if the given end-effector position and orientation belong to the robot workspace.

On the other hand, the problem of multiple solutions depends not only on the number of degrees of freedom but also on the Denavit-Hartenberg parameters; in general, the greater is the number of nonnull parameters, the greater is the number of admissible solutions. For a 6-degree-of-freedom robot without mechanical joint limits, there are in general up to 16 admissible solutions. This occurrence demands some criterion to choose among admissible solutions.

The computation of closed-form solutions requires either algebraic intuition to find out those significant equations containing the unknowns or geometric intuition to find out those significant points on the structure with respect to which it is convenient to express position and orientation. 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 to be applicable to any kinematic structure, but in general they do not allow computation of all admissible solutions.

4.1 Closed-Form Solutions

Most of the existing robots are kinematically simple, since they are typically formed by an arm (three or more degrees of freedom) which provides mobility and by a wrist which provides dexterity (three degrees of freedom). This choice is partly motivated by the difficulty to find solutions to the inverse kinematics problem in the general case. In particular, a *six*-degree-of-freedom robot has closed-form inverse kinematics solutions if three consecutive revolute joint axes intersect at a common point. This situation occurs when a robot has a so-called *spherical wrist* which is characterized by

$$\ell_5 = d_5 = \ell_6 = 0 \quad \xi_4 = \xi_5 = \xi_6 = 0, \quad (37)$$

with $\sin \alpha_5 \neq 0$ and $\sin \alpha_6 \neq 0$ so as to avoid parallel axes (degenerate robot). In that case, it is possible to articulate the inverse kinematics problem into two subproblems, since the solution for the *position* is *decoupled* from that for the *orientation*.

In the case of a three-degree-of-freedom arm, for given end-effector position ${}^0\mathbf{p}_e$ and orientation ${}^0\mathbf{R}_e$, the inverse kinematics can be solved according to the following steps:

- compute the wrist position ${}^0\mathbf{p}_4$ from ${}^0\mathbf{p}_e$;
- solve inverse kinematics for (q_1, q_2, q_3) ;
- compute ${}^0\mathbf{R}_3(q_1, q_2, q_3)$;
- compute ${}^3\mathbf{R}_6(q_4, q_5, q_6) = {}^3\mathbf{R}_0{}^0\mathbf{R}_e{}^e\mathbf{R}_6$;
- solve inverse kinematics for (q_4, q_5, q_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.

Consider the anthropomorphic robot in Fig. 4, whose direct kinematics was given in (30). It is desired to find the vector of joint variables \mathbf{q} corresponding to given end-effector position ${}^0\mathbf{p}_e$ and orientation ${}^0\mathbf{R}_e$; without loss of generality, assume that ${}^0\mathbf{p}_e = {}^0\mathbf{p}_6$ and ${}^6\mathbf{R}_e = I$.

Observing that ${}^0\mathbf{p}_6 = {}^0\mathbf{p}_4$, the first three joint variables can be solved from (31) which can be rewritten as

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} c_1(c_2\ell_3 - s_{23}d_4) \\ s_1(c_2\ell_3 - s_{23}d_4) \\ s_2\ell_3 + c_{23}d_4 \end{bmatrix}. \quad (38)$$

From the first two components of (38), it is

$$q_1 = \text{Atan2}(p_y, p_x). \quad (39)$$

Notice that another solution is

$$q_1 = \pi + \text{Atan2}(p_y, p_x). \quad (40)$$

The second joint variable can be found by squaring and summing the first two components of (38), i.e.

$$p_x^2 + p_y^2 = (c_2 \ell_3 - s_{23} d_4)^2; \quad (41)$$

then, squaring the third component and summing it to (41) leads to the solution

$$q_3 = \text{Atan2}(s_3, c_3) \quad (42)$$

where

$$s_3 = \frac{\ell_3^2 + d_4^2 - p_x^2 - p_y^2 - p_z^2}{2\ell_3 d_4} \quad c_3 = \pm \sqrt{1 - s_3^2}.$$

Substituting q_3 in (41), taking the square root thereof and combining the result with the third component of (38) leads to a system of equations in the unknowns s_2 and c_2 ; its solution can be found as

$$s_2 = \frac{(\ell_3 - s_3 d_4) p_z - c_3 d_4 \sqrt{p_x^2 + p_y^2}}{p_x^2 + p_y^2 + p_z^2}$$

$$c_2 = \frac{(\ell_3 - s_3 d_4) \sqrt{p_x^2 + p_y^2} + c_3 d_4 p_z}{p_x^2 + p_y^2 + p_z^2},$$

and thus the second joint variable is

$$q_2 = \text{Atan2}(s_2, c_2). \quad (43)$$

Notice that four admissible solutions are obtained according to the values of q_1 , q_2 , q_3 ; namely, shoulder-right/elbow-up, shoulder-left/elbow-up, shoulder-right/elbow-down, shoulder-left/elbow-down.

In order to solve for the three joint variables of the wrist, the following procedure can be followed. Given the matrix

$${}^0\mathbf{R}_6 = \begin{pmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{pmatrix}, \quad (44)$$

the matrix ${}^0\mathbf{R}_3$ can be computed from the first three joint variables via (24), and thus the following equation is to be considered

$$\begin{bmatrix} {}^3n_x & {}^3s_x & {}^3a_x \\ {}^3n_y & {}^3s_y & {}^3a_y \\ {}^3n_z & {}^3s_z & {}^3a_z \end{bmatrix} = \begin{bmatrix} c_4 c_5 c_6 - s_4 s_6 & -c_4 c_5 s_6 - s_4 c_6 & -c_4 s_5 \\ s_5 c_6 & -s_5 s_6 & c_5 \\ -s_4 c_5 c_6 - c_4 s_6 & s_4 c_5 s_6 - c_4 c_6 & s_4 s_5 \end{bmatrix}. \quad (45)$$

The elements of the matrix on the right-hand side of (45) have been obtained by computing ${}^3\mathbf{R}_6$ via (24), whereas the elements of the matrix on the left-hand side of (45) can be computed as ${}^3\mathbf{R}_0{}^0\mathbf{R}_6$ with ${}^0\mathbf{R}_6$ as in (44), i.e.

$$\begin{aligned} {}^3n_x &= c_{23}(c_1n_x + s_1n_y) + s_{23}n_z \\ {}^3n_y &= -s_{23}(c_1n_x + s_1n_y) + s_{23}n_z \\ {}^3n_z &= s_1n_x - c_1n_y; \end{aligned} \quad (46)$$

the other elements $({}^3s_x, {}^3s_y, {}^3s_z)$ and $({}^3a_x, {}^3a_y, {}^3a_z)$ can be computed from (46) by replacing (n_x, n_y, n_z) with (s_x, s_y, s_z) and (a_x, a_y, a_z) , respectively.

At this point, inspecting (45) reveals that from the elements [1, 3] and [3, 3], q_4 can be computed as

$$q_4 = \text{Atan2}({}^3a_z, -{}^3a_x). \quad (47)$$

Then, q_5 can be computed by squaring and summing the elements [1, 3] and [3, 3], and from the element [2, 3] as

$$q_5 = \text{Atan2}(\sqrt{({}^3a_x)^2 + ({}^3a_z)^2}, {}^3a_y). \quad (48)$$

Finally, q_6 can be computed from the elements [2, 1] and [2, 2] as

$$q_6 = \text{Atan2}(-{}^3s_y, {}^3n_y). \quad (49)$$

It is worth noticing that another set of solutions is given by the triplet

$$q_4 = \text{Atan2}(-{}^3a_z, {}^3a_x) \quad (50)$$

$$q_5 = \text{Atan2}(-\sqrt{({}^3a_x)^2 + ({}^3a_z)^2}, {}^3a_y) \quad (51)$$

$$q_6 = \text{Atan2}({}^3s_y, -{}^3n_y). \quad (52)$$

Notice that both sets of solutions degenerate when ${}^3a_x = {}^3a_z = 0$; in this case, q_4 is arbitrary and simpler expressions can be found for q_5 and q_6 .

In conclusion, 4 admissible solutions have been found for the arm and 2 admissible solutions have been found for the wrist, resulting in a total of 8 admissible inverse kinematics solutions for the anthropomorphic robot with a spherical wrist.

5 Differential Kinematics

The (3×1) vector $\dot{\mathbf{p}}$ of linear velocity of a rigid body in space is given by the time derivative of the position vector, while the (3×1) vector $\boldsymbol{\omega}$ of angular velocity can be defined through the time derivative of the rotation matrix in the form

$$\dot{\mathbf{R}} = \mathbf{S}(\boldsymbol{\omega})\mathbf{R}. \quad (53)$$

With reference to the other descriptions of orientation, the relationship between the angular velocity and the time derivative of the unit quaternion is

$$\begin{bmatrix} \dot{\eta} \\ \dot{\boldsymbol{\varepsilon}} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -\mathbf{S}(\boldsymbol{\omega}) \end{bmatrix} \begin{bmatrix} \eta \\ \boldsymbol{\varepsilon} \end{bmatrix} \quad (54)$$

which is known as quaternion propagation rule, whereas that between the angular velocity and the time derivative of the Euler angles is

$$\boldsymbol{\omega} = \mathbf{T}(\boldsymbol{\varphi})\dot{\boldsymbol{\varphi}} \quad (55)$$

where $\mathbf{T}(\boldsymbol{\varphi})$ depends on the particular choice of Euler angles.

The mapping between the $(n \times 1)$ vector of joint velocities $\dot{\mathbf{q}}$ and the (6×1) vector of end-effector (linear and angular) velocities \mathbf{v} is established by the *differential kinematics* equation

$$\mathbf{v} = \begin{bmatrix} \dot{\mathbf{p}} \\ \boldsymbol{\omega} \end{bmatrix} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \quad (56)$$

where $\mathbf{J}(\mathbf{q})$ is the $(6 \times n)$ Jacobian matrix. The computation of this matrix usually follows a geometric procedure that is based on computing the contributions of each joint velocity to the linear and angular end-effector velocities. Hence, $\mathbf{J}(\mathbf{q})$ can be termed as the *geometric Jacobian* of the robot.

5.1 Geometric Jacobian

In view of simple geometry, the velocity contributions of each joint to the linear and angular velocities of link n give the following relationship

$$\begin{aligned} \begin{bmatrix} \dot{\mathbf{p}}_n \\ \boldsymbol{\omega}_n \end{bmatrix} &= \begin{bmatrix} \xi_1 z_1 + \bar{\xi}_1 (z_1 \times \mathbf{p}_{1n}) & \dots & \xi_n z_n + \bar{\xi}_n (z_n \times \mathbf{p}_{nn}) \\ \bar{\xi}_1 z_1 & \dots & \bar{\xi}_n z_n \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix} \\ &= \mathbf{J}_n(\mathbf{q})\dot{\mathbf{q}} \end{aligned} \quad (57)$$

where z_k is the unit vector of axis Z_k and \mathbf{p}_{kn} denotes the vector from the origin of frame k to the origin of frame n . Notice that \mathbf{J}_n is a function of \mathbf{q} through the vectors z_k and \mathbf{p}_{kn} which can be computed on the basis of direct kinematics.

The geometric Jacobian can be computed with respect to any frame i ; in that case, the k -th column of ${}^i\mathbf{J}_n$ is given by

$${}^i\mathbf{J}_{nk} = \begin{bmatrix} \xi_k {}^i z_k + \bar{\xi}_k {}^i \mathbf{R}_k \mathbf{S}({}^k z_k) {}^k \mathbf{p}_n \\ \bar{\xi}_k {}^i z_k \end{bmatrix} \quad (58)$$

where ${}^k \mathbf{p}_n = {}^k \mathbf{p}_{kn}$. In view of the expression of ${}^k z_k = [0 \ 0 \ 1]$, eq. (58) can be rewritten as

$${}^i\mathbf{J}_{nk} = \begin{bmatrix} \xi_k {}^i z_k + \bar{\xi}_k (-{}^k p_{ny} {}^i \mathbf{x}_k + {}^k p_{nx} {}^i \mathbf{y}_k) \\ \bar{\xi}_k {}^i z_k \end{bmatrix} \quad (59)$$

where ${}^k p_{nx}$ and ${}^k p_{ny}$ are the x and y components of ${}^k \mathbf{p}_n$. A number of remarks are in order.

- The transformation of the Jacobian from frame i to a different frame l can be obtained as

$${}^l \mathbf{J}_n = \begin{bmatrix} {}^l \mathbf{R}_i & \mathbf{0} \\ \mathbf{0} & {}^l \mathbf{R}_i \end{bmatrix} {}^i \mathbf{J}_n. \quad (60)$$

- The Jacobian relating the end-effector velocity to the joint velocities can be computed either by using (57) and replacing \mathbf{p}_{kn} with \mathbf{p}_{ke} , or by using the relationship

$${}^i \mathbf{J}_e = \begin{bmatrix} \mathbf{I} & -\mathbf{S}({}^i \mathbf{p}_{ne}) \\ \mathbf{O} & \mathbf{I} \end{bmatrix} {}^i \mathbf{J}_n. \quad (61)$$

A Jacobian ${}^i \mathbf{J}_n$ can be decomposed as the product of three matrices, where the first two are full-rank, while the third one has the same rank as ${}^i \mathbf{J}_n$ but contains simpler elements to compute. To achieve this, the Jacobian of link n can be expressed as a function of a generic Jacobian

$$\mathbf{J}_{n,h} = \begin{bmatrix} \xi_1 \mathbf{z}_1 + \bar{\xi}_1 (\mathbf{z}_1 \times \mathbf{p}_{1h}) & \dots & \xi_n \mathbf{z}_n + \bar{\xi}_n (\mathbf{z}_n \times \mathbf{p}_{nh}) \\ \bar{\xi}_1 \mathbf{z}_1 & \dots & \bar{\xi}_n \mathbf{z}_n \end{bmatrix} \quad (62)$$

giving the velocity of a frame fixed to link n attached instantaneously to frame h . Then \mathbf{J}_n can be computed via (61) as

$$\mathbf{J}_n = \begin{bmatrix} \mathbf{I} & -\mathbf{S}(\mathbf{p}_{hn}) \\ \mathbf{O} & \mathbf{I} \end{bmatrix} \mathbf{J}_{n,h} \quad (63)$$

which can be expressed with respect to frame i , giving

$${}^i \mathbf{J}_n = \begin{bmatrix} \mathbf{I} & -\mathbf{S}({}^i \mathbf{R}_h {}^h \mathbf{p}_n) \\ \mathbf{O} & \mathbf{I} \end{bmatrix} {}^i \mathbf{J}_{n,h}. \quad (64)$$

Combining (60) with (64) yields the result that the matrix ${}^l \mathbf{J}_n$ can be computed as the product of three matrices

$${}^l \mathbf{J}_n = \begin{bmatrix} {}^l \mathbf{R}_i & \mathbf{O} \\ \mathbf{O} & {}^l \mathbf{R}_i \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{S}({}^i \mathbf{R}_h {}^h \mathbf{p}_n) \\ \mathbf{O} & \mathbf{I} \end{bmatrix} {}^i \mathbf{J}_{n,h}, \quad (65)$$

where remarkably the first two matrices are full-rank. In general, the values of h and i leading to the Jacobian ${}^i \mathbf{J}_{n,h}$ of simplest expression are given by

$$i = \text{int}(n/2) \quad h = \text{int}(n/2) + 1.$$

Hence, for a robot with 6 degrees of freedom, the matrix ${}^3 \mathbf{J}_{6,4}$ is expected to have the simplest expression; if the wrist is spherical ($\mathbf{p}_{46} = \mathbf{0}$), then the second matrix in (65) is identity and ${}^3 \mathbf{J}_{6,4} = {}^3 \mathbf{J}_6$.

As an example, the geometric Jacobian for the anthropomorphic robot in Fig. 4 can be computed on the basis of the matrix

$${}^3\mathbf{J}_6 = \begin{bmatrix} 0 & \ell_3 s_3 - d_4 & -d_4 & 0 & 0 & 0 \\ 0 & \ell_3 c_3 & 0 & 0 & 0 & 0 \\ -\ell_3 c_2 + d_4 s_{23} & 0 & 0 & 0 & 0 & 0 \\ s_{23} & 0 & 0 & 0 & s_4 & -c_4 s_5 \\ c_{23} & 0 & 0 & 1 & 0 & c_5 \\ 0 & 1 & 1 & 0 & c_4 & s_4 s_5 \end{bmatrix}. \quad (66)$$

5.2 Analytical Jacobian

If the end-effector position and orientation are specified in terms of a minimum number of parameters in the task space as in (36), it is possible to compute the Jacobian matrix also by direct differentiation of the direct kinematics equation, i.e.

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{p}}_e \\ \dot{\boldsymbol{\varphi}}_e \end{bmatrix} = \mathbf{J}_a(\mathbf{q})\dot{\mathbf{q}}, \quad (67)$$

where the matrix $\mathbf{J}_a(\mathbf{q}) = \partial \mathbf{k} / \partial \mathbf{q}$ is termed *analytical Jacobian*.

The relationship between the analytical Jacobian and the geometric Jacobian is expressed as

$$\mathbf{J} = \begin{bmatrix} \mathbf{I} & \mathbf{O} \\ \mathbf{O} & \mathbf{T}(\boldsymbol{\varphi}_e) \end{bmatrix} = \mathbf{T}_a(\boldsymbol{\varphi}_e)\mathbf{J}_a, \quad (68)$$

where $\mathbf{T}(\boldsymbol{\varphi}_e)$ is the transformation matrix defined in (55) which depends on the particular set of Euler angles used to represent end-effector orientation.

It can be easily recognized that the two Jacobians are in general different; note, however, that the two coincide for the positioning part. Concerning their use, the geometric Jacobian is adopted when physical quantities are of interest while the analytical Jacobian is adopted when task space quantities are of interest. It is always possible to pass from one Jacobian to the other, except when the transformation matrix is singular; the orientations at which the determinant of $\mathbf{T}(\boldsymbol{\varphi}_e)$ vanishes are called *representation singularities* of $\boldsymbol{\varphi}_e$. For instance, with reference to the *XYZ* representation in (18), the transformation matrix is

$$\mathbf{T}(\boldsymbol{\varphi}_e) = \begin{bmatrix} 1 & 0 & s_\beta \\ 0 & c_\alpha & -s_\alpha c_\beta \\ 0 & s_\alpha & c_\alpha c_\beta \end{bmatrix}. \quad (69)$$

It can be recognized that \mathbf{T} becomes singular at the representation singularities $\beta = \pm\pi/2$; notice that, in these configurations, it is not possible to describe an arbitrary angular velocity with a set of Euler angles time derivatives. It should be remarked that each of the other Euler angles descriptions suffers from the occurrence of two representation singularities.

5.3 Singularities

The differential kinematics equation (56) defines a linear mapping between the vector of joint velocities $\dot{\mathbf{q}}$ and the vector of end-effector velocities \mathbf{v} . The Jacobian is in general a function of the robot configuration \mathbf{q} ; those configurations at which \mathbf{J} is rank-deficient are called *kinematic singularities*.

The simplest means to find singularities is to compute the determinant of the Jacobian matrix. For instance, for the above Jacobian in (66) it is

$$\det({}^3\mathbf{J}_6) = \ell_3 d_4 c_3 s_5 (d_4 s_{23} - \ell_3 c_2) \quad (70)$$

leading to three types of singularities ($\ell_3, d_4 \neq 0$). These are the *elbow singularity*

$$c_3 = 0$$

occurring when link 2 and link 3 are aligned; the *shoulder singularity*

$$d_4 s_{23} - \ell_3 c_2 = 0$$

occurring when origin of frame 4 is along axis Z_0 ; and the *wrist singularity*

$$s_5 = 0$$

occurring when axes Z_4 and Z_6 are aligned. Notice that the elbow singularity is not troublesome since it occurs at the boundary of the robot workspace ($q_3 = \pm\pi/2$). The shoulder singularity is characterized in the task space and thus it can be avoided when planning an end-effector trajectory. Instead, the wrist singularity is characterized in the joint space ($q_5 = 0, \pi$), and thus it is difficult to predict when planning an end-effector trajectory.

An effective tool to analyze the linear mapping from the joint velocity space into the task velocity space defined by (56) is offered by the *singular value decomposition* (SVD) of the Jacobian matrix; this is given by

$$\mathbf{J} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_{i=1}^m \sigma_i \mathbf{u}_i \mathbf{v}_i^T, \quad (71)$$

where \mathbf{U} is the ($m \times m$) matrix of the output singular vectors \mathbf{u}_i , \mathbf{V} is the ($n \times n$) matrix of the input singular vectors \mathbf{v}_i , and $\mathbf{\Sigma} = [\mathbf{S} \ \mathbf{O}]$ is the ($m \times n$) matrix whose ($m \times m$) diagonal submatrix \mathbf{S} contains the singular values σ_i of the matrix \mathbf{J} . If r denotes the rank of \mathbf{J} , the following properties hold:

- $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_m = 0$,
- $\mathcal{R}(\mathbf{J}) = \text{span}\{\mathbf{u}_1, \dots, \mathbf{u}_r\}$,
- $\mathcal{N}(\mathbf{J}) = \text{span}\{\mathbf{v}_{r+1}, \dots, \mathbf{v}_n\}$.

The null space $\mathcal{N}(\mathbf{J})$ is the set of joint velocities that yield null task velocities at the current configuration; these joint velocities are termed *null space joint velocities*. A base of $\mathcal{N}(\mathbf{J})$ is given by the $(n - r)$ last input singular vectors, which represent independent linear combinations of the joint velocities. Hence, one effect of a singularity is to increase the dimension of $\mathcal{N}(\mathbf{J})$ by introducing a linear combination of joint velocities that produce a null task velocity.

The range space $\mathcal{R}(\mathbf{J})$ is the set of task velocities that can be obtained as a result of all possible joint velocities; these task velocities are termed *feasible space task velocities*. A base of $\mathcal{R}(\mathbf{J})$ is given by the first r output singular vectors, which represent independent linear combinations of the single components of task velocities. Accordingly, another effect of a singularity is to decrease the dimension of $\mathcal{R}(\mathbf{J})$ by eliminating a linear combination of task velocities from the space of feasible velocities.

The singular value decomposition (71) shows that the i -th singular value of \mathbf{J} can be viewed as a gain factor relating the joint velocity along the \mathbf{v}_i direction to the task velocity along the \mathbf{u}_i direction. When a singularity is approached, the r -th singular value tends to zero and the task velocity produced by a fixed joint velocity along \mathbf{v}_r is decreased proportionally to σ_r . At the singular configuration, the joint velocity along \mathbf{v}_r is in the null space and the task velocity along \mathbf{u}_r becomes infeasible.

In the general case, the joint velocity has components in any \mathbf{v}_i direction, and the resulting task velocity can be obtained as a combination of the single components along each output singular vector direction.

6 Differential Kinematics Inversion

The differential kinematics equation, in terms of either the geometric or the analytical Jacobian establishes a linear mapping between joint space velocities and task space velocities, even if the Jacobian is a function of the joint configuration. This feature suggests the use of the differential kinematics equation (56) to solve the inverse kinematics problem.

Assume that a task space trajectory is given $(\mathbf{x}(t), \mathbf{v}(t))$. The goal is to find a feasible joint space trajectory $(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ that reproduces the given trajectory. Joint velocities can be obtained by solving the differential kinematics equation for $\dot{\mathbf{q}}$ at the current joint configuration; then, joint positions $\mathbf{q}(t)$ can be computed by integrating the velocity solution over time with known initial conditions. This approach is based on the knowledge of the robot Jacobian and thus is applicable to any robot structure, on condition that a suitable inverse for the matrix \mathbf{J} can be found.

6.1 Pseudoinverse

With reference to the geometric Jacobian, the basic inverse solution to (56) is obtained by using the *pseudoinverse* \mathbf{J}^\dagger of the matrix \mathbf{J} ; this is a unique matrix satisfying the Moore-Penrose conditions

$$\begin{aligned} \mathbf{J}\mathbf{J}^\dagger\mathbf{J} &= \mathbf{J} & \mathbf{J}^\dagger\mathbf{J}\mathbf{J}^\dagger &= \mathbf{J}^\dagger \\ (\mathbf{J}\mathbf{J}^\dagger)^\top &= \mathbf{J}\mathbf{J}^\dagger & (\mathbf{J}^\dagger\mathbf{J})^\top &= \mathbf{J}^\dagger\mathbf{J} \end{aligned} \quad (72)$$

or, alternatively, the equivalent conditions

$$\begin{aligned} \mathbf{J}^\dagger\mathbf{a} &= \mathbf{a} & \forall \mathbf{a} \in \mathcal{N}^\perp(\mathbf{J}) \\ \mathbf{J}^\dagger\mathbf{b} &= \mathbf{0} & \forall \mathbf{b} \in \mathcal{R}^\perp(\mathbf{J}) \\ \mathbf{J}^\dagger(\mathbf{a} + \mathbf{b}) &= \mathbf{J}^\dagger\mathbf{a} + \mathbf{J}^\dagger\mathbf{b} & \forall \mathbf{a} \in \mathcal{R}(\mathbf{J}), \forall \mathbf{b} \in \mathcal{R}^\perp(\mathbf{J}). \end{aligned} \quad (73)$$

The inverse solution can then be written as

$$\dot{\mathbf{q}} = \mathbf{J}^\dagger(\mathbf{q})\mathbf{v} \quad (74)$$

that provides a least-squares solution with minimum norm to equation (56); in detail, solution (74) satisfies the condition

$$\min_{\dot{\mathbf{q}}} \|\dot{\mathbf{q}}\| \quad (75)$$

of all $\dot{\mathbf{q}}$ that fulfill

$$\min_{\dot{\mathbf{q}}} \|\mathbf{v} - \mathbf{J}\dot{\mathbf{q}}\|. \quad (76)$$

If the Jacobian matrix is full-rank, the right pseudoinverse of \mathbf{J} can be computed as

$$\mathbf{J}^\dagger = \mathbf{J}^\top(\mathbf{J}\mathbf{J}^\top)^{-1}, \quad (77)$$

and (74) provides an exact solution to (56); further, if \mathbf{J} square, the pseudoinverse (77) reduces to the standard inverse Jacobian matrix \mathbf{J}^{-1} .

To gain insight into the properties of the inverse mapping described by (74), it is useful to consider the singular value decomposition (71) of \mathbf{J} , and thus

$$\mathbf{J}^\dagger = \mathbf{V}\boldsymbol{\Sigma}^\dagger\mathbf{U}^\top = \sum_{i=1}^r \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{u}_i^\top \quad (78)$$

where r denotes the rank of \mathbf{J} . The following properties hold:

- $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_m = 0$,
- $\mathcal{R}(\mathbf{J}^\dagger) = \mathcal{N}^\perp(\mathbf{J}) = \text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_r\}$,
- $\mathcal{N}(\mathbf{J}^\dagger) = \mathcal{R}^\perp(\mathbf{J}) = \text{span}\{\mathbf{u}_{r+1}, \dots, \mathbf{u}_n\}$.

The null space $\mathcal{N}(\mathbf{J}^\dagger)$ is the set of task velocities that yield null joint space velocities at the current configuration; these task velocities belong to the orthogonal complement of the feasible space task velocities. Hence, one effect of the pseudoinverse solution (74) is to filter the infeasible components of the given task velocities while allowing exact tracking of the feasible components; this is due to the minimum norm property (75).

The range space $\mathcal{R}(\mathbf{J}^\dagger)$ is the set of joint velocities that can be obtained as a result of all possible task velocities. Since these joint velocities belong to the orthogonal complement of the null space joint velocities, the pseudoinverse solution (74) satisfies the least-squares condition (76).

If a task velocity is assigned along \mathbf{u}_i , the corresponding joint velocity computed via (74) lies along \mathbf{v}_i and is magnified by the factor $1/\sigma_i$. When a *singularity* is approached, the r -th singular value tends to zero and a fixed task velocity along \mathbf{u}_r requires large joint velocities. At a singular configuration, the \mathbf{u}_r direction becomes infeasible and \mathbf{v}_r adds to the set of null space velocities of the robot.

6.2 Redundancy

For a kinematically *redundant* robot a nonempty null space $\mathcal{N}(\mathbf{J})$ exists which is available to set up systematic procedures for an effective handling of redundant degrees of freedom. The general inverse solution can be written as

$$\dot{\mathbf{q}} = \mathbf{J}^\dagger(\mathbf{q})\mathbf{v} + (\mathbf{I} - \mathbf{J}^\dagger(\mathbf{q})\mathbf{J}(\mathbf{q}))\dot{\mathbf{q}}_0 \quad (79)$$

which satisfies the least-squares condition (76) but loses the minimum norm property (75), by virtue of the addition of the homogeneous term $(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\dot{\mathbf{q}}_0$; the matrix $(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})$ is a projector of the joint vector $\dot{\mathbf{q}}_0$ onto $\mathcal{N}(\mathbf{J})$.

In terms of the singular value decomposition, solution (79) can be written in the form

$$\dot{\mathbf{q}} = \sum_{i=1}^r \mathbf{v}_i \mathbf{u}_i^T \mathbf{v} + \sum_{i=r+1}^m \mathbf{v}_i \mathbf{v}_i^T \dot{\mathbf{q}}_0 + \sum_{i=m+1}^n \mathbf{v}_i \mathbf{v}_i^T \dot{\mathbf{q}}_0. \quad (80)$$

Three contributions can be recognized in (80); namely, the least-squares joint velocities, the null space joint velocities due to singularities (if $r < m$), and the null space joint velocities due to redundant degrees of freedom (if $m < n$).

This result is of fundamental importance for redundancy resolution, since solution (79) evidences the possibility of choosing the vector $\dot{\mathbf{q}}_0$ so as to exploit the redundant degrees of freedom. In fact, the contribution of $\dot{\mathbf{q}}_0$ is to generate null space motions of the structure that do not alter the task space configuration but allow the robot to reach postures which are more dexterous for the execution of the given task.

A typical choice of the null space joint velocity vector is

$$\dot{\mathbf{q}}_0 = \alpha \left(\frac{\partial w(\mathbf{q})}{\partial \mathbf{q}} \right)^T \quad (81)$$

with $\alpha > 0$; $w(\mathbf{q})$ is a scalar objective function of the joint variables and $(\partial w(\mathbf{q})/\partial \mathbf{q})^T$ is the vector function representing the *gradient* of w . In this way, it is sought to *locally* optimize w in accordance with the kinematic constraint expressed by (56). Usual objective functions are:

- the *manipulability measure* defined as

$$w(\mathbf{q}) = \sqrt{\det(\mathbf{J}(\mathbf{q})\mathbf{J}^T(\mathbf{q}))}, \quad (82)$$

which vanishes at a singular configuration, and thus redundancy may be exploited to escape singularities;

- the *distance from mechanical joint limits* defined as

$$w(\mathbf{q}) = -\frac{1}{2n} \sum_{i=1}^n \left(\frac{q_i - \bar{q}_i}{q_{iM} - q_{im}} \right)^2, \quad (83)$$

where q_{iM} (q_{im}) denotes the maximum (minimum) limit for q_i and \bar{q}_i the middle of the joint range, and thus redundancy may be exploited to keep the robot off joint limits;

- the *distance from an obstacle* defined as

$$w(\mathbf{q}) = \min_{\mathbf{p}, \mathbf{o}} \|\mathbf{p}(q) - \mathbf{o}\|, \quad (84)$$

where \mathbf{o} is the position vector of an opportune point on the obstacle and \mathbf{p} is the position vector of the closest robot point to the obstacle, and thus redundancy may be exploited to avoid collisions with obstacles.

6.3 Damped Least-Squares Inverse

In the neighborhood of singular configurations the use of a pseudoinverse is not adequate and a numerically robust solution is achieved by the *damped least-squares inverse* technique. This is based on the solution to the modified differential kinematics equation

$$\mathbf{J}^T \mathbf{v} = (\mathbf{J}^T \mathbf{J} + \lambda^2 \mathbf{I}) \dot{\mathbf{q}} \quad (85)$$

in place of equation (56); in (85) the scalar λ is the so-called *damping factor*. Note that, when $\lambda = 0$, equation (85) reduces to (56).

The solution to (85) can be written in either of the equivalent forms

$$\dot{\mathbf{q}} = \mathbf{J}^T (\mathbf{J} \mathbf{J}^T + \lambda^2 \mathbf{I})^{-1} \mathbf{v} \quad (86)$$

$$\dot{\mathbf{q}} = (\mathbf{J}^T \mathbf{J} + \lambda^2 \mathbf{I})^{-1} \mathbf{J}^T \mathbf{v}. \quad (87)$$

The computational load of (86) is lower than that of (87), being usually $n \geq m$. Let then

$$\dot{\mathbf{q}} = \mathbf{J}^\#(\mathbf{q})\mathbf{v} \quad (88)$$

indicate the damped least-squares inverse solution computed with either of the above forms. Solution (88) satisfies the condition

$$\min_{\dot{\mathbf{q}}} \|\mathbf{v} - \mathbf{J}\dot{\mathbf{q}}\|^2 + \lambda^2 \|\dot{\mathbf{q}}\|^2 \quad (89)$$

that gives a trade-off between the least-squares condition (76) and the minimum norm condition (75). In fact, condition (89) accounts for both accuracy and feasibility in choosing the joint space velocity $\dot{\mathbf{q}}$ required to produce the given task space velocity \mathbf{v} . In this regard, it is essential to select a suitable value for the damping factor; small values of λ give accurate solutions but low robustness in the neighborhood of singular configurations, while large values of λ result in low tracking accuracy even if feasible and accurate solutions would be possible.

Resorting to the singular value decomposition, the damped least-squares inverse solution (88) can be written as

$$\dot{\mathbf{q}} = \sum_{i=1}^r \frac{\sigma_i}{\sigma_i^2 + \lambda^2} \mathbf{v}_i \mathbf{u}_i^T \mathbf{v}. \quad (90)$$

Remarkably, it is:

- $\mathcal{R}(\mathbf{J}^\#) = \mathcal{R}(\mathbf{J}^\dagger) = \mathcal{N}^\perp(\mathbf{J}) = \text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_r\}$,
- $\mathcal{N}(\mathbf{J}^\#) = \mathcal{N}(\mathbf{J}^\dagger) = \mathcal{R}^\perp(\mathbf{J}) = \text{span}\{\mathbf{u}_{r+1}, \dots, \mathbf{u}_n\}$,

that is, the structural properties of the damped least-squares inverse solution are analogous to those of the pseudoinverse solution.

It is clear that, with respect to the pure least-squares solution (74), the components for which $\sigma_i \gg \lambda$ are little influenced by the damping factor, since in this case it is

$$\frac{\sigma_i}{\sigma_i^2 + \lambda^2} \approx \frac{1}{\sigma_i}. \quad (91)$$

On the other hand, when a singularity is approached, the smallest singular value tends to zero while the associated component of the solution is driven to zero by the factor σ_i/λ^2 ; this progressively reduces the joint velocity to achieve near-degenerate components of the commanded task velocity. At the singularity, solutions (88) and (74) behave identically as long as the remaining singular values are significantly larger than the damping factor. Note that an upper bound of $1/2\lambda$ is set on the magnification factor relating the task velocity component along \mathbf{u}_i to the resulting joint velocity along \mathbf{v}_i ; this bound is reached when $\sigma_i = \lambda$.

The damping factor λ determines the degree of approximation introduced with respect to the pure least-squares solution; then, using a constant value for λ may turn out to be inadequate for obtaining good performance over the entire robot workspace. An effective choice is to adjust λ as a function of some measure of closeness to the singularity at the current configuration of the robot; to this purpose, manipulability measures or estimates of the smallest singular value can be adopted. Remarkably, currently available microprocessors even allow real-time computation of full singular value decomposition.

A singular region can be defined on the basis of the estimate of the smallest singular value of \mathbf{J} ; outside the region the exact solution is used, while inside the region a configuration-varying damping factor is introduced to obtain the desired approximate solution. The factor must be chosen so that continuity of joint velocity $\dot{\mathbf{q}}$ is ensured in the transition at the border of the singular region.

Without loss of generality, for a 6-degree-of-freedom robot, the damping factor can be selected according to the following law:

$$\lambda^2 = \begin{cases} 0 & \hat{\sigma}_6 \geq \varepsilon \\ \left(1 - \left(\frac{\hat{\sigma}_6}{\varepsilon}\right)^2\right) \lambda_{\max}^2 & \hat{\sigma}_6 < \varepsilon, \end{cases} \quad (92)$$

where $\hat{\sigma}_6$ is the estimate of the smallest singular value, and ε defines the size of the singular region; the value of λ_{\max} is at user's disposal to suitably shape the solution in the neighborhood of a singularity.

Equation (92) requires computation of the smallest singular value. In order to avoid a full singular value decomposition, we can resort to a recursive algorithm to find an estimate of the smallest singular value. Suppose that an estimate $\hat{\mathbf{v}}_6$ of the last input singular vector is available, so that $\hat{\mathbf{v}}_6 \approx \mathbf{v}_6$ and $\|\hat{\mathbf{v}}_6\| = 1$. This estimate is used to compute the vector $\hat{\mathbf{v}}'_6$ from

$$(\mathbf{J}^T \mathbf{J} + \lambda^2 \mathbf{I}) \hat{\mathbf{v}}'_6 = \hat{\mathbf{v}}_6. \quad (93)$$

Then the square of the estimate $\hat{\sigma}_6$ of the smallest singular value can be found as

$$\hat{\sigma}_6^2 = \frac{1}{\|\hat{\mathbf{v}}'_6\|} - \lambda^2, \quad (94)$$

while the estimate of \mathbf{v}_6 is updated using

$$\hat{\mathbf{v}}_6 = \frac{\hat{\mathbf{v}}'_6}{\|\hat{\mathbf{v}}'_6\|}. \quad (95)$$

The above estimation scheme is based on the assumption that \mathbf{v}_6 is slowly rotating, which is normally the case. However, if the robot is close to a double singularity (e.g. a shoulder and a wrist singularity for the anthropomorphic robot), the vector \mathbf{v}_6 will instantaneously rotate if the two smallest singular values cross. The estimate of the smallest singular value will then track σ_5

initially, before $\hat{\boldsymbol{v}}_6$ converges again to \boldsymbol{v}_6 . Therefore, it is worth extending the scheme by estimating not only the smallest but also the second smallest singular value. Assume that the estimates $\hat{\boldsymbol{v}}_6$ and $\hat{\sigma}_6$ are available and define the matrix

$$\boldsymbol{M} = \boldsymbol{J}^T \boldsymbol{J} + \lambda^2 \boldsymbol{I} - (\hat{\sigma}_6^2 + \lambda^2) \hat{\boldsymbol{v}}_6 \hat{\boldsymbol{v}}_6^T. \quad (96)$$

With this choice, the second smallest singular value of \boldsymbol{J} plays in

$$\boldsymbol{M} \hat{\boldsymbol{v}}'_5 = \hat{\boldsymbol{v}}_5 \quad (97)$$

the same role as σ_6 in (93) and then will provide a convergent estimate of $\hat{\boldsymbol{v}}_5$ to \boldsymbol{v}_5 and $\hat{\sigma}_5$ to σ_5 .

At this point, suppose that $\hat{\boldsymbol{v}}_5$ is an estimate of \boldsymbol{v}_5 so that $\hat{\boldsymbol{v}}_5 \approx \boldsymbol{v}_5$ and $\|\hat{\boldsymbol{v}}_5\| = 1$. This estimate is used to compute $\hat{\boldsymbol{v}}'_5$ from (97). Then, an estimate of the square of the second smallest singular value of \boldsymbol{J} is found from

$$\hat{\sigma}_5^2 = \frac{1}{\|\hat{\boldsymbol{v}}'_5\|} - \lambda^2, \quad (98)$$

and the estimate of \boldsymbol{v}_5 is updated using

$$\hat{\boldsymbol{v}}_5 = \frac{\hat{\boldsymbol{v}}'_5}{\|\hat{\boldsymbol{v}}'_5\|}. \quad (99)$$

On the basis of this modified estimation algorithm, crossing of singularities can be effectively detected; also, by switching the two singular values and the associated estimates $\hat{\boldsymbol{v}}_5$ and $\hat{\boldsymbol{v}}_6$, the estimation of the smallest singular value will be accurate even when the two smallest singular values cross.

6.4 User-Defined Accuracy

The above damped least-squares inverse method achieves a compromise between accuracy and robustness of the solution. This is performed without specific regard to the components of the particular task assigned to the robot's end-effector. The *user-defined accuracy* strategy based on the weighted damped least-squares inverse method allows discriminating between directions in the task space where higher accuracy is desired and directions where lower accuracy can be tolerated. This is the case, for instance, of spot welding or spray painting in which the tool angle about the approach direction is not essential to the fulfillment of the task.

Let a weighted end-effector velocity vector be defined as

$$\bar{\boldsymbol{v}} = \boldsymbol{W} \boldsymbol{v} \quad (100)$$

where \boldsymbol{W} is the $(m \times m)$ task-dependent weighting matrix taking into account the anisotropy of the task requirements. Substituting (100) into (56) gives

$$\bar{\boldsymbol{v}} = \bar{\boldsymbol{J}}(\boldsymbol{q}) \dot{\boldsymbol{q}} \quad (101)$$

where $\bar{\mathbf{J}} = \mathbf{W}\mathbf{J}$. It is worth noticing that if \mathbf{W} is full-rank, solving (56) is equivalent to solving (101), but with different conditioning of the system of equations to solve. This suggests selecting only the strictly necessary weighting action in order to avoid undesired ill-conditioning of $\bar{\mathbf{J}}$.

Equation (101) can be solved by using the weighted damped least-squares inverse technique, i.e.

$$\bar{\mathbf{J}}^T(\mathbf{q})\bar{\mathbf{v}} = (\bar{\mathbf{J}}^T(\mathbf{q})\bar{\mathbf{J}}(\mathbf{q}) + \lambda^2\mathbf{I})\dot{\mathbf{q}}. \quad (102)$$

Again, the singular value decomposition of the matrix $\bar{\mathbf{J}}$ is helpful, i.e.

$$\bar{\mathbf{J}} = \sum_{i=1}^r \bar{\sigma}_i \bar{\mathbf{u}}_i \bar{\mathbf{v}}_i^T \quad (103)$$

and the solution to (102) can be written as

$$\dot{\mathbf{q}} = \sum_{i=1}^r \frac{\bar{\sigma}_i}{\bar{\sigma}_i^2 + \lambda^2} \bar{\mathbf{v}}_i \bar{\mathbf{u}}_i^T \bar{\mathbf{v}}. \quad (104)$$

It is clear that the singular values $\bar{\sigma}_i$ and the singular vectors $\bar{\mathbf{u}}_i$ and $\bar{\mathbf{v}}_i$ depend on the choice of the weighting matrix \mathbf{W} . While this has no effect on the solution $\dot{\mathbf{q}}$ as long as $\bar{\sigma}_r \gg \lambda$, close to singularities where $\bar{\sigma}_r \ll \lambda$, for some $r < m$, the solution can be shaped by properly selecting the matrix \mathbf{W} .

For a 6-degree-of-freedom robot with spherical wrist, it is worthwhile to devise a special handling of the wrist singularity, since such a singularity is difficult to predict at the planning level in the task space. It can be recognized that, at the wrist singularity, there are only two components of the angular velocity vector that can be generated by the wrist itself. The remaining component might be generated by the inner joints, at the expense of loss of accuracy along some other task space directions, though. For this reason, lower weight should be put on the angular velocity component that is infeasible to the wrist. For the anthropomorphic robot, this is easily expressed in the frame attached to link 4; let \mathbf{R}_4 denote the rotation matrix describing orientation of this frame with respect to the base frame, so that the infeasible component is aligned with the x -axis. Then the weighting matrix can be chosen as

$$\mathbf{W} = \begin{bmatrix} \mathbf{I} & \mathbf{O} \\ \mathbf{O} & \mathbf{R}_4 \text{diag}\{w, 1, 1\} \mathbf{R}_4^T \end{bmatrix}. \quad (105)$$

Similarly to the choice of the damping factor as in (92), the weighting factor w is selected according to the following expression:

$$(1-w)^2 = \begin{cases} 0 & \hat{\sigma}_6 \geq \varepsilon \\ \left(1 - \left(\frac{\hat{\sigma}_6}{\varepsilon}\right)^2\right) (1-w_{\min})^2 & \hat{\sigma}_6 < \varepsilon, \end{cases} \quad (106)$$

where $w_{\min} > 0$ is a design parameter.

7 Inverse Kinematics Algorithms

The differential kinematics equation has been utilized above to solve for joint velocities. Open-loop reconstruction of joint variables through numerical integration unavoidably leads to solution drift and then to task space errors. This drawback can be overcome by devising a closed-loop *inverse kinematics algorithm* based on the task space error between the desired and actual end-effector locations \mathbf{x}_d and \mathbf{x} , i.e. $\mathbf{e} = \mathbf{x}_d - \mathbf{x}(\mathbf{q})$. It is worth considering also the differential kinematics equation in the form (67) where the definition of the task error has required consideration of the analytical Jacobian \mathbf{J}_a in lieu of the geometric Jacobian.

7.1 Jacobian Pseudoinverse

The joint velocity vector shall be chosen so that the task error tends to zero. The simplest algorithm is obtained by using the *Jacobian pseudoinverse*

$$\dot{\mathbf{q}} = \mathbf{J}_a^\dagger(\mathbf{q}) (\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}), \quad (107)$$

which plugged into (67) gives

$$\dot{\mathbf{e}} + \mathbf{K}\mathbf{e} = 0. \quad (108)$$

If \mathbf{K} is a positive definite (diagonal) matrix, the linear system (108) is asymptotically stable; the tracking error along the given trajectory converges to zero with a rate depending on the eigenvalues of \mathbf{K} .

A block scheme of the inverse kinematics algorithm based on the Jacobian pseudoinverse is illustrated in Fig. 5.

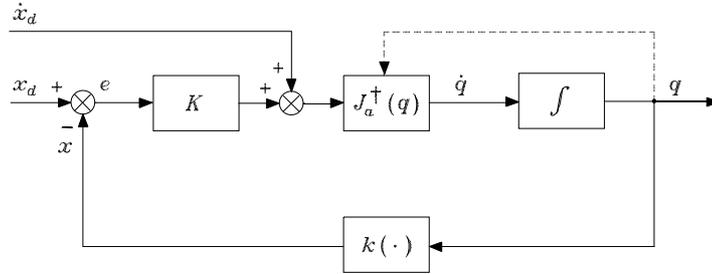


Figure 5: Block scheme of the inverse kinematics algorithm with Jacobian pseudoinverse.

If it is desired to exploit redundant degrees of freedom, solution (107) can be generalized to

$$\dot{\mathbf{q}} = \mathbf{J}_a^\dagger(\mathbf{q}) (\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}) + (\mathbf{I} - \mathbf{J}_a^\dagger(\mathbf{q})\mathbf{J}_a(\mathbf{q}))\dot{\mathbf{q}}_0 \quad (109)$$

that logically corresponds to (79). In case of numerical problems in the neighborhood of singularities, the pseudoinverse can be replaced with a suitable damped least-squares inverse.

7.2 Jacobian Transpose

A computationally efficient inverse kinematics algorithm can be derived by considering the *Jacobian transpose* in lieu of the pseudoinverse.

Consider the joint velocity vector

$$\dot{\mathbf{q}} = \mathbf{J}_a^T(\mathbf{q})\mathbf{K}\mathbf{e} \quad (110)$$

where \mathbf{K} is a symmetric positive definite matrix. A simple Lyapunov argument can be used to analyze the convergence of the algorithm. Consider the positive definite function candidate

$$V = \frac{1}{2}\mathbf{e}^T\mathbf{K}\mathbf{e}; \quad (111)$$

its time derivative along the trajectories of the system (67) and (110) is

$$\dot{V} = \mathbf{e}^T\mathbf{K}\dot{\mathbf{x}}_d - \mathbf{e}^T\mathbf{K}\mathbf{J}_a(\mathbf{q})\mathbf{J}_a^T(\mathbf{q})\mathbf{K}\mathbf{e}. \quad (112)$$

If $\dot{\mathbf{x}}_d = \mathbf{0}$, it is easy to see that \dot{V} is negative definite as long as \mathbf{J}_a is full-rank, and then it can be concluded that $\mathbf{e} = \mathbf{0}$ is an asymptotically stable equilibrium point for the system (67) and (110) as long as \mathbf{J}_a is full-rank for all joint configurations \mathbf{q} . A number of remarks are in order.

- If $\dot{\mathbf{x}}_d \neq \mathbf{0}$, only boundedness of tracking errors can be established; an estimate of the bound is given by

$$\|\mathbf{e}\|_{\max} = \frac{\|\dot{\mathbf{x}}_d\|_{\max}}{k\sigma_r^2(\mathbf{J}_a)} \quad (113)$$

where \mathbf{K} has been conveniently chosen as a diagonal matrix $\mathbf{K} = k\mathbf{I}$. It is anticipated that k can be increased to diminish the errors, but in practice upper bounds exist due to discrete-time implementation of the algorithm.

- When a singularity is encountered, $\mathcal{N}(\mathbf{J}_a^T)$ is non-empty and \dot{V} is only semi-definite; $\dot{V} = 0$ for $\mathbf{e} \neq \mathbf{0}$ with $\mathbf{K}\mathbf{e} \in \mathcal{N}(\mathbf{J}_a^T)$, and the algorithm may get stuck. It can be shown, however, that such an equilibrium point is unstable as long as $\dot{\mathbf{x}}_d$ drives $\mathbf{K}\mathbf{e}$ outside $\mathcal{N}(\mathbf{J}_a^T)$. An enhancement of the algorithm can be achieved by rendering the matrix $\mathbf{J}_a^T\mathbf{K}$ less sensitive to variations of joint configurations along the task trajectory; this is accomplished by choosing a configuration-dependent \mathbf{K} that compensates for variations of \mathbf{J}_a .

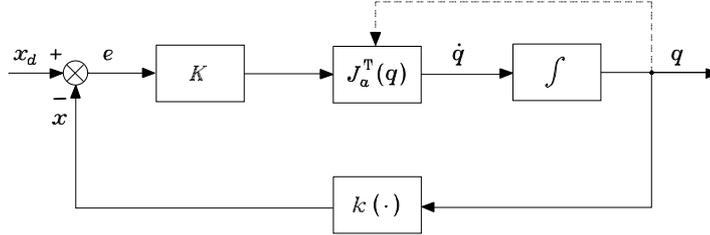


Figure 6: Block scheme of the inverse kinematics algorithm with Jacobian transpose.

A block scheme of the inverse kinematics algorithm based on the Jacobian transpose is illustrated in Fig. 6.

The most attractive feature of the Jacobian transpose algorithm is certainly the need of computing only direct kinematics functions $\mathbf{k}(\mathbf{q})$ and $\mathbf{J}_a(\mathbf{q})$. Further insight into the performance of solution (110) can be gained by considering the singular value decomposition of the Jacobian transpose, and thus

$$\mathbf{J}^T = \sum_{i=1}^m \sigma_i \mathbf{v}_i \mathbf{u}_i^T \quad (114)$$

which reveals a continuous, smooth behavior of the solution close and through singular configurations; note that in (114) the geometric Jacobian has been considered and it has been assumed that no representation singularities are introduced.

7.3 Use of Redundancy

In case of redundant degrees of freedom, it is possible to combine the Jacobian pseudoinverse solution with the Jacobian transpose solution as illustrated below. This is carried out in the framework of the so-called *augmented task space* approach to exploit redundancy in robotic systems. The idea is to introduce an additional constraint task by specifying a $(p \times 1)$ vector \mathbf{x}_c as a function of the robot joint variables, i.e.

$$\mathbf{x}_c = \mathbf{k}_c(\mathbf{q}), \quad (115)$$

with $p \leq n - m$ so as to constrain at most all the available redundant degrees of freedom. The constraint task vector \mathbf{x}_c can be chosen by embedding scalar objective functions of the kind introduced in (82)–(84).

Differentiating (115) with respect to time gives

$$\dot{\mathbf{x}}_c = \mathbf{J}_c(\mathbf{q})\dot{\mathbf{q}} \quad (116)$$

where $\mathbf{J}_c(\mathbf{q}) = \partial \mathbf{k}_c / \partial \mathbf{q}$ is the constraint Jacobian. The result is an augmented differential kinematics equation given by (67) and (116), based on a Jacobian matrix

$$\mathbf{J}' = \begin{bmatrix} \mathbf{J}_a \\ \mathbf{J}_c \end{bmatrix}. \quad (117)$$

When a constraint task is specified independently of the end-effector task, there is no guarantee that the matrix \mathbf{J}' remains full-rank along the entire task path. Even if $\text{rank}(\mathbf{J}_a) = m$ and $\text{rank}(\mathbf{J}_c) = p$, then $\text{rank}(\mathbf{J}') = m + p$ if and only if $\mathcal{R}(\mathbf{J}_a^T) \cap \mathcal{R}(\mathbf{J}_c^T) = \{\emptyset\}$; singularities of \mathbf{J}' are termed *artificial singularities* and it can be shown that those are given by singularities of the matrix $\mathbf{J}_c(\mathbf{I} - \mathbf{J}_a^\dagger \mathbf{J}_a)$.

The above discussion suggests that, when solving for joint velocities, a *task priority strategy* is advisable so as to avoid conflicting situations between the end-effector task and the constraint task. Substituting (109) into (116) gives

$$\dot{\mathbf{x}}_c = \mathbf{J}_c(\mathbf{q})\mathbf{J}_a^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}) + \mathbf{J}_c(\mathbf{q})(\mathbf{I} - \mathbf{J}_a^\dagger(\mathbf{q})\mathbf{J}_a(\mathbf{q}))\dot{\mathbf{q}}_0 \quad (118)$$

which could be solved for $\dot{\mathbf{q}}_0$ provided that artificial singularities —those of the matrix $\mathbf{J}_c(\mathbf{I} - \mathbf{J}_a^\dagger \mathbf{J}_a)$ — are avoided. Observing that equality (118) can be achieved only for the components of $\dot{\mathbf{x}}_c$ belonging to $\mathcal{R}(\mathbf{J}_c)$, it is sufficient to consider the equation

$$\mathbf{J}_c^\dagger(\mathbf{q})\dot{\mathbf{x}}_c = \mathbf{J}_a^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}) + (\mathbf{I} - \mathbf{J}_a^\dagger(\mathbf{q})\mathbf{J}_a(\mathbf{q}))\dot{\mathbf{q}}_0 \quad (119)$$

that can be solved for $\dot{\mathbf{q}}_0$ giving

$$\dot{\mathbf{q}}_0 = (\mathbf{I} - \mathbf{J}_a^\dagger(\mathbf{q})\mathbf{J}_a(\mathbf{q}))^\dagger (\mathbf{J}_c^\dagger(\mathbf{q})\dot{\mathbf{x}}_c - \mathbf{J}_a^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e})). \quad (120)$$

By recalling that $(\mathbf{I} - \mathbf{J}_a^\dagger \mathbf{J}_a)^\dagger = (\mathbf{I} - \mathbf{J}_a^\dagger \mathbf{J}_a)$, solution (120) reduced to the simple form

$$\dot{\mathbf{q}}_0 = (\mathbf{I} - \mathbf{J}_a^\dagger(\mathbf{q})\mathbf{J}_a(\mathbf{q}))\mathbf{J}_c^\dagger(\mathbf{q})\dot{\mathbf{x}}_c. \quad (121)$$

Folding (121) back into (109) and exploiting the idempotence of $(\mathbf{I} - \mathbf{J}_a^\dagger \mathbf{J}_a)$ gives

$$\dot{\mathbf{q}} = \mathbf{J}_a^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}) + (\mathbf{I} - \mathbf{J}_a^\dagger(\mathbf{q})\mathbf{J}_a(\mathbf{q}))\mathbf{J}_c^\dagger(\mathbf{q})(\dot{\mathbf{x}}_{cd} + \mathbf{K}_c\mathbf{e}_c) \quad (122)$$

where $\mathbf{e}_c = \mathbf{x}_{cd} - \mathbf{x}_c$, being \mathbf{x}_{cd} the desired value of the constraint task, and \mathbf{K}_c is a positive definite matrix. The operator $(\mathbf{I} - \mathbf{J}_a^\dagger \mathbf{J}_a)$ projects the secondary velocity contribution $\dot{\mathbf{q}}_0$ on the null space $\mathcal{N}(\mathbf{J}_a)$, guaranteeing correct execution of the primary end-effector task while the secondary constraint task is correctly executed as long as it does not interfere with the end-effector task. Obviously, if desired, the order of priority can be switched, e.g. in an obstacle avoidance task when an obstacle comes to be along the end-effector path.

In the case when \mathbf{J}_c becomes singular, a damped least-squares inverse of \mathbf{J}_c in lieu of the pseudoinverse in (121) can be used. Otherwise, by recalling the

Jacobian transpose solution for the end-effector task (110), the null space joint velocity vector can be conveniently chosen as

$$\dot{\mathbf{q}}_0 = \mathbf{J}_c^T(\mathbf{q})\mathbf{K}_c(\mathbf{x}_{cd} - \mathbf{x}_c). \quad (123)$$

which allows the algorithm to work at a singularity of \mathbf{J}_c and even at an artificial singularity. A tracking error arises for the constraint task but, observing that the desired constraint task is often constant over time ($\dot{\mathbf{x}}_{cd} = \mathbf{0}$), it can be concluded that the solution based on (123) performs equally well.

7.4 Orientation Errors

The above inverse kinematics algorithms make use of the analytical Jacobian since they operate on error variables (position and orientation) which are defined in the task space. More insight about the implications of different end-effector orientation descriptions can be gained by separating the position from the orientation components. With reference to the pseudoinverse algorithm based on (107), using the geometric Jacobian in lieu of the analytical Jacobian, the solution can be rewritten as

$$\dot{\mathbf{q}} = \mathbf{J}^\dagger(\mathbf{q}) \begin{bmatrix} \mathbf{v}_p \\ \mathbf{v}_o \end{bmatrix} \quad (124)$$

where $\mathbf{v}_p, \mathbf{v}_o$ represent two resolved velocities that shall be chosen so as to ensure tracking of the desired end-effector motion. Substituting (124) into (56) gives

$$\dot{\mathbf{p}}_e = \mathbf{v}_p \quad (125)$$

$$\boldsymbol{\omega}_e = \mathbf{v}_o \quad (126)$$

where the explicit end-effector linear and angular velocities have been evidenced.

For what concerns position, the choice is rather straightforward, i.e.

$$\mathbf{v}_p = \dot{\mathbf{p}}_d + \mathbf{K}_p \mathbf{e}_p \quad (127)$$

where the position error

$$\mathbf{e}_p = \mathbf{p}_d - \mathbf{p}_e(\mathbf{q}) \quad (128)$$

between the desired and actual end-effector positions has been defined. Substituting (127) into (125) gives

$$\dot{\mathbf{e}}_p + \mathbf{K}_p \mathbf{e}_p = \mathbf{0} \quad (129)$$

and the choice of a positive definite matrix \mathbf{K}_p guarantees asymptotic stability of the error system which in turn implies tracking of \mathbf{p}_d .

On the other hand, for what concerns the orientation error, some considerations are in order depending on the type of description adopted. If *Euler angles* are adopted, the resolved angular velocity in (124) is chosen as

$$\mathbf{v}_o = \mathbf{T}(\boldsymbol{\varphi}_e)(\dot{\boldsymbol{\varphi}}_d + \mathbf{K}_o \mathbf{e}_{o, \text{Eul}}) \quad (130)$$

where

$$\mathbf{e}_{o,\text{Eul}} = \boldsymbol{\varphi}_d - \boldsymbol{\varphi}_e(\mathbf{q}) \quad (131)$$

is the orientation error. Substituting (130) into (126) gives

$$\dot{\mathbf{e}}_{o,\text{Eul}} + \mathbf{K}_o \mathbf{e}_{o,\text{Eul}} = \mathbf{0} \quad (132)$$

provided that the matrix $\mathbf{T}(\boldsymbol{\varphi}_e)$ is nonsingular. The system (132) is asymptotically stable for a positive definite \mathbf{K}_o , which in turn implies tracking of $\boldsymbol{\varphi}_d$.

In order to overcome the drawback of representation singularities in (130), an algorithm based on an *alternative Euler angles* description can be conceived which makes use of the rotation matrix describing the mutual orientation between the desired and the actual end-effector frame, i.e.

$${}^e \mathbf{R}_d = \mathbf{R}_e^{\text{T}}(\mathbf{q}) \mathbf{R}_d. \quad (133)$$

Differentiating (133) with respect to time and accounting for (53) gives

$${}^e \dot{\mathbf{R}}_d = \mathbf{S}({}^e \boldsymbol{\omega}_{de}) {}^e \mathbf{R}_d \quad (134)$$

where $\boldsymbol{\omega}_{de} = \boldsymbol{\omega}_d - \boldsymbol{\omega}_e(\mathbf{q})$ is the end-effector angular velocity error.

Let $\boldsymbol{\varphi}_{de}$ denote the set of Euler angles that can be extracted from ${}^e \mathbf{R}_d$. Then, in view of (55) and (53), the angular velocity ${}^e \boldsymbol{\omega}_{de}$ in (134) is related to the time derivative of $\boldsymbol{\varphi}_{de}$ as

$${}^e \boldsymbol{\omega}_{de} = \mathbf{T}(\boldsymbol{\varphi}_{de}) \dot{\boldsymbol{\varphi}}_{de}. \quad (135)$$

At this point, the resolved angular velocity in (124) can be chosen as

$$\mathbf{v}_o = \boldsymbol{\omega}_d + \mathbf{R}_e \mathbf{T}(\boldsymbol{\varphi}_{de}) \mathbf{K}_o \mathbf{e}_{o,\text{EulAlt}} \quad (136)$$

where

$$\mathbf{e}_{o,\text{EulAlt}} = \boldsymbol{\varphi}_{de}. \quad (137)$$

Substituting (136) into (126) gives

$$\dot{\mathbf{e}}_{o,\text{EulAlt}} + \mathbf{K}_o \mathbf{e}_{o,\text{EulAlt}} = \mathbf{0} \quad (138)$$

provided that the matrix $\mathbf{T}(\boldsymbol{\varphi}_{de})$ is nonsingular.

The clear advantage of the alternative over the classical Euler angles algorithm based on (130) is that, by adopting a representation $\boldsymbol{\phi}_{de}$ for which $\mathbf{T}(\mathbf{0})$ is nonsingular, representation singularities occur only for large orientation errors, e.g. when $\beta_{de} = \pm\pi/2$ for the *XYZ* representation. In other words, the ill-conditioning of matrix \mathbf{T} is not influenced by the desired or actual end-effector orientation but only by the orientation error; hence, as long as the error parameter $|\beta_{de}| < \pi/2$, the behavior of system (138) is not affected by representation singularities. In this respect, the choice of a particular Euler angles description

among the twelve possible should be carefully made, i.e. in the sense of avoiding a representation singularity for the second angle of the type $\beta = 0$.

In order to overcome the problem of representation singularities, an inverse kinematics algorithm based on the *angle/axis* description of orientation can be devised. From (133), the rotation ϑ_{de} and the unit vector \mathbf{r}_{de} can be extracted using the formulæ (12). Then, the orientation error can be defined as

$$\mathbf{e}_{o, \text{AnAx}} = \sin \vartheta_{de} \mathbf{r}_{de}. \quad (139)$$

Notice that (139) gives a unique solution for $-\pi/2 < \vartheta < \pi/2$, but this interval is not limiting for a convergent inverse kinematics algorithm. It can be shown that a computational expression of the orientation error in (139) is given by

$$\mathbf{e}_{o, \text{AnAx}} = \frac{1}{2} (\mathbf{S}(\mathbf{n}_e(\mathbf{q})) \mathbf{n}_d + \mathbf{S}(\mathbf{s}_e(\mathbf{q})) \mathbf{s}_d + \mathbf{S}(\mathbf{a}_e(\mathbf{q})) \mathbf{a}_d), \quad (140)$$

where the triplet of unit vectors has been used for both the desired and the actual end-effector rotation matrix. Note that the above limitation on ϑ sets the conditions $\mathbf{n}_e^T \mathbf{n}_d \geq 0$, $\mathbf{s}_e^T \mathbf{s}_d \geq 0$, $\mathbf{a}_e^T \mathbf{a}_d \geq 0$.

Differentiation of (140) with respect to time gives

$$\dot{\mathbf{e}}_{o, \text{AnAx}} = \mathbf{L}^T \boldsymbol{\omega}_d - \mathbf{L} \boldsymbol{\omega} \quad (141)$$

where

$$\mathbf{L} = -\frac{1}{2} (\mathbf{S}(\mathbf{n}_d) \mathbf{S}(\mathbf{n}_e) + \mathbf{S}(\mathbf{s}_d) \mathbf{S}(\mathbf{s}_e) + \mathbf{S}(\mathbf{a}_d) \mathbf{S}(\mathbf{a}_e)). \quad (142)$$

At this point, the resolved angular velocity in (124) can be chosen as

$$\mathbf{v}_o = \mathbf{L}^{-1} (\mathbf{L}^T \boldsymbol{\omega}_d + \mathbf{K}_o \mathbf{e}_{o, \text{AnAx}}). \quad (143)$$

Substituting (143) into (126) gives

$$\dot{\mathbf{e}}_{o, \text{AnAx}} + \mathbf{K}_o \mathbf{e}_{o, \text{AnAx}} = \mathbf{0} \quad (144)$$

provided that the matrix \mathbf{L} is nonsingular. In this respect, if the angle ϑ_{de} is extended to the interval $(-\pi, \pi)$, then a singularity occurs at $\vartheta_{de} = \pm\pi/2$ for the matrix \mathbf{L} which does not allow the computation of \mathbf{v}_o as in (143).

The final inverse kinematics algorithm is based on the *unit quaternion* description of orientation. Let $\mathcal{Q}_d = \{\eta_d, \boldsymbol{\varepsilon}_d\}$ and $\mathcal{Q}_e = \{\eta_e, \boldsymbol{\varepsilon}_e\}$ represent the unit quaternions associated with \mathbf{R}_d and \mathbf{R}_e , respectively. The mutual orientation can be expressed in terms of the unit quaternion $\mathcal{Q}_{de} = \{\eta_{de}, \boldsymbol{\varepsilon}_{de}\}$ where

$$\begin{aligned} \eta_{de} &= \eta_e(\mathbf{q}) \eta_d + \boldsymbol{\varepsilon}_e^T(\mathbf{q}) \boldsymbol{\varepsilon}_d \\ \boldsymbol{\varepsilon}_{de} &= \eta_e(\mathbf{q}) \boldsymbol{\varepsilon}_d - \eta_d \boldsymbol{\varepsilon}_e(\mathbf{q}) - \mathbf{S}(\boldsymbol{\varepsilon}_d) \boldsymbol{\varepsilon}_e(\mathbf{q}). \end{aligned} \quad (145)$$

It can be recognized that $\mathcal{Q}_{de} = \{1, \mathbf{0}\}$ if and only if \mathbf{R}_e and \mathbf{R}_d are aligned, and thus it is sufficient to consider $\boldsymbol{\varepsilon}_{de}$ to express an end-effector orientation error, i.e.

$$\mathbf{e}_{o, \text{Quat}} = \boldsymbol{\varepsilon}_{de}. \quad (146)$$

Notice that the explicit computation of $\eta_e(\mathbf{q})$ and $\varepsilon_e(\mathbf{q})$ is not possible, but it requires the intermediate computation of the rotation matrix $\mathbf{R}_e(\mathbf{q})$ that is available from the robot direct kinematics; then, the unit quaternion can be extracted using the formulæ (17).

At this point, the resolved angular velocity in (124) can be chosen as

$$\mathbf{v}_o = \boldsymbol{\omega}_d + \mathbf{K}_o \mathbf{e}_{o, \text{Quat}} \quad (147)$$

Substituting (147) into (126) gives

$$\boldsymbol{\omega}_{de} + \mathbf{K}_o \mathbf{e}_{o, \text{Quat}} = \mathbf{0}. \quad (148)$$

It should be observed that now the orientation error equation is not homogeneous in $\mathbf{e}_{o, \text{Quat}}$ since it contains the end-effector angular velocity error instead of the time derivative of the orientation error. To study stability of system (148), consider the positive definite Lyapunov function

$$V = (\eta_d - \eta_e)^2 + (\boldsymbol{\varepsilon}_d - \boldsymbol{\varepsilon}_e)^T (\boldsymbol{\varepsilon}_d - \boldsymbol{\varepsilon}_e). \quad (149)$$

In view of the quaternion propagation (54), the time derivative of V along the trajectories of system (148) is given by

$$\dot{V} = -\mathbf{e}_{o, \text{Quat}}^T \mathbf{K}_o \mathbf{e}_{o, \text{Quat}} \quad (150)$$

which is negative definite, implying that $\mathbf{e}_{o, \text{Quat}}$ converges to zero.

8 Further Reading

Kinematic modelling of rigid robot manipulators can be found in any classical robotics textbook, e.g. [43, 17, 19, 54, 56, 51]. Precious reference sources on kinematics are also [25, 3, 1, 57, 38]. Symbolic software packages have been developed to derive robot kinematic models, e.g. [26].

The Denavit-Hartenberg notation dates back to the original work of [18], which was recently modified in [17, 28]. One advantage of the so-called modified Denavit-Hartenberg notation over the classical one is that it can be used also for tree-structured and closed-chain robots [28]. The homogeneous transformation representation for direct kinematics of open-chain robots was first proposed in [45].

Sufficient conditions for the inverse kinematics problem to have closed-form solutions were given in [45]. These ensure the existence of solutions to 6-degree-of-freedom robots provided that there are three revolute joints with intersecting axes or three prismatic joints; in the former case, at most 8 admissible solutions exist, while the number reduces to 2 in the latter case. The kinematic decoupling resulting for spherical-wrist robots was developed in [22, 24, 44, 27]. An algebraic approach to the inverse kinematics problem for robots having closed-form

solutions was presented in [43], which consists of successively post- (or pre-) multiplying both sides of the direct kinematics equation by partial transformation matrices so as to isolate the joint variables one after another; the types of equations that can be obtained with this approach were formalized in [19]. Recent methods [32, 46] have found the inverse kinematics solution to general six-revolute-joint robots in the form of a polynomial equation of degree 16, i.e. the maximum number of admissible solutions is 16. On the other hand, numerical solution techniques based on iterative algorithms have been proposed, e.g. [55, 23].

The geometric Jacobian of the differential kinematics equation was originally proposed in [59]. The decomposition of the Jacobian into the product of three matrices is due to [47]. The problem of efficient Jacobian computation was addressed in [42]. The analytical Jacobian concept was introduced in [29] in connection with the operational space control problem. A treatment of differential kinematics mapping properties can be found in [51]; the reader is referred to [31] for SVD decomposition.

The inversion of differential kinematics dates back to [59] under the name of resolved motion rate control. The adoption of the pseudoinverse of the Jacobian is due to [30]. More on the properties of the pseudoinverse can be found in [4]. The use of null-space joint velocities for redundancy resolution was proposed in [33], and further refined in [60, 36] as concerns the choice of objective functions. The reader is referred to [39] for a complete treatment of redundant robots.

The adoption of the damped least-squares inverse was independently presented in [40] and [58]. More about kinematic control in the neighborhood of kinematic singularities can be found in [9]. The technique for estimating the smallest singular value of the Jacobian is due to [37], and its modification to include the second smallest singular value was achieved by [10]. The use of the damped least-squares inverse for redundant robots was presented in [21]. The user-defined accuracy strategy was proposed in [12] and further refined in [13]. A review of the damped least-squares inverse kinematics with experiments on an industrial robot was recently presented [16].

Closed-loop inverse kinematics algorithms are discussed in [51]. The original Jacobian transpose inverse kinematics algorithm was proposed in [49]; the choice of suitable gains for achieving robustness to singularities was discussed in [7]. Singular value decomposition of the Jacobian transpose is due to [14]. Combination of the Jacobian transpose solution with the pseudoinverse solution was proposed in [8]. References on the augmented task space approach are [20, 50, 52, 48]. The occurrence of artificial singularities was pointed out in [2], and their properties were studied in [6]. The task priority strategy was originally proposed in [41] and has recently been refined in [11] concerning robustness to artificial singularities. The use of the Jacobian transpose for the constraint task was presented in [15, 53]. The expression of the end-effector orientation error based on an angle/axis description of orientation is due to [35],

and its properties were studied in [34]. The use of a quaternion-based orientation error is due to [61]. More about the possible definitions of the orientation error can be found in [5].

References

- [1] Angeles, J., *Spatial Kinematic Chains: Analysis, Synthesis, Optimization*, Springer-Verlag, Berlin, D, 1982.
- [2] Baillieul, J., Kinematic programming alternatives for redundant manipulators, in *Proc. 1985 IEEE Int. Conf. on Robotics and Automation*, St. Louis, MO, 1985, 722.
- [3] Bottema, O. and Roth, B., *Theoretical Kinematics*, North Holland, Amsterdam, NL, 1979.
- [4] Boullion, T. L. and Odell, P. L., *Generalized Inverse Matrices*, Wiley, New York, NY, 1971.
- [5] Caccavale, F., Natale, C., Siciliano, B., and Villani, L., Resolved-acceleration control of robot manipulators: A critical review with experiments, *Robotica*, 16, 565, 1998.
- [6] Chiacchio, P., Chiaverini, S., Sciavicco, L., and Siciliano, B., Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy, *Int. J. of Robotics Research*, 10, 410, 1991.
- [7] Chiacchio, P., and Siciliano, B., Achieving singularity robustness: An inverse kinematic solution algorithm for robot control, in *Robot Control: Theory and Applications*, IEE Control Engineering Series 36, Warwick, K. and Pugh, A., Eds., Peter Peregrinus, Herts, UK, 1988, 149.
- [8] Chiacchio, P. and Siciliano, B., A closed-loop Jacobian transpose scheme for solving the inverse kinematics of nonredundant and redundant robot wrists, *J. of Robotic Systems*, 6, 601, 1989.
- [9] Chiaverini, S., Inverse differential kinematics of robotic manipulators at singular and near-singular configurations, in *Prepr. 1992 IEEE Int. Conf. on Robotics and Automation — Tutorial on ‘Redundancy: Performance Indices, Singularities Avoidance, and Algorithmic Implementations’*, Nice, F, 1992.
- [10] Chiaverini, S., Estimate of the two smallest singular values of the Jacobian matrix: Application to damped least-squares inverse kinematics, *J. of Robotic Systems*, 10, 991, 1993.

- [11] Chiaverini, S., Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators, *IEEE Trans. on Robotics and Automation*, 13, 398, 1997.
- [12] Chiaverini, S., Egeland, O., and Kanestrøm, R. K., Achieving user-defined accuracy with damped least-squares inverse kinematics, in *Proc. 5th Int. Conf. on Advanced Robotics*, Pisa, I, 1991, 672.
- [13] Chiaverini, S., Egeland, O., Sagli, J. R., and Siciliano, B., User-defined accuracy in the augmented task space approach for redundant manipulators, *Laboratory Robotics and Automation*, 4, 59, 1992.
- [14] Chiaverini, S., Sciavicco, L., and Siciliano, B., Control of robotic systems through singularities, in *Advanced Robot Control*, Lecture Notes in Control and Information Science 162, Canudas de Wit, C., Ed., Springer-Verlag, Berlin, D, 1991, 285.
- [15] Chiaverini, S., Siciliano, B., and Egeland, O., Redundancy resolution for the human-arm-like manipulator, *Robotics and Autonomous Systems*, 8, 239, 1991.
- [16] Chiaverini, S., Siciliano, B., and Egeland, O., Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator, *IEEE Trans. on Control Systems Technology*, 2, 123, 1994.
- [17] Craig, J. J., *Introduction to Robotics: Mechanics and Control*, 2nd ed., Addison-Wesley, Reading, MA, 1989.
- [18] Denavit, J. and Hartenberg, R. S., A kinematic notation for lower-pair mechanisms based on matrices, *ASME J. of Applied Mechanics*, 22, 215, 1955.
- [19] Dombre, E. and Khalil, W., *Modélisation et Commande des Robots*, Hermès, Paris, F, 1988.
- [20] Egeland, O., Task-space tracking with redundant manipulators, *IEEE J. of Robotics and Automation*, 3, 471, 1987.
- [21] Egeland, O., Sagli, J. R., Spangelo, I., and Chiaverini, S., A damped least-squares solution to redundancy resolution, in *Proc. 1991 IEEE Int. Conf. on Robotics and Automation*, Sacramento, CA, 1991, 945.
- [22] Featherstone, R., Position and velocity transformations between robot end-effector coordinates and joint angles, *Int. J. of Robotics Research*, 2(2), 35, 1983.
- [23] Goldenberg, A. A., Benhabib, B., and Fenton, R. G., A complete generalized solution to the inverse kinematics of robots, *IEEE J. of Robotics and Automation*, 1, 14, 1985.

- [24] Hollerbach, J. M., Wrist-partitioned inverse kinematic accelerations and manipulator dynamics, *Int J. of Robotics Research*, 2(4), 61, 1983.
- [25] Hunt, K. H., *Kinematic Geometry of Mechanisms*, Clarendon Press, Oxford, UK, 1978.
- [26] Khalil, W., A system for generating the symbolic models of robots, in *Postpr. 4th IFAC Symp. on Robot Control*, Capri, I, 1994, 416.
- [27] Khalil, W. and Bennis, F., Automatic generation of the inverse geometric model of robots, *Robotics and Autonomous Systems*, 7, 1, 1991.
- [28] Khalil, W. and Kleinfinger, J. F., A new geometric notation for open and closed-loop robots, in *Proc. 1986 IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, 1986, 1174.
- [29] Khatib, O., A unified approach for motion and force control of robot manipulators: The operational space formulation, *IEEE J. of Robotics and Automation*, 3, 43, 1987.
- [30] Klein, C. A. and Huang, C. H., Review of pseudoinverse control for use with kinematically redundant manipulators, *IEEE Trans. on Systems, Man, and Cybernetics*, 13, 245, 1983.
- [31] Klema, V. C. and Laub, A. J., The singular value decomposition: Its computation and some applications, *IEEE Trans. on Automatic Control*, 25, 164, 1980.
- [32] Lee, H. Y. and Liang, C. G., Displacement analysis of the general 7-link 7R mechanism, *Mechanism and Machine Theory*, 23, 219, 1988.
- [33] Liégeois, A., Automatic supervisory control of the configuration and behavior of multibody mechanisms, *IEEE Trans. on Systems, Man, and Cybernetics*, 7, 868, 1977.
- [34] Lin, S. K., Singularity of a nonlinear feedback control scheme for robots, *IEEE Trans. on Systems, Man, and Cybernetics*, 19, 134, 1989.
- [35] Luh, J. Y. S., Walker, M. W., and Paul, R. P. C., Resolved-acceleration control of mechanical manipulators, *IEEE Trans. on Automatic Control*, 25, 468, 1980.
- [36] Maciejewski, A. A. and Klein, C. A., Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments, *Int. J. of Robotics Research*, 4(3), 109, 1985.
- [37] Maciejewski, A. A. and Klein, C. A., Numerical filtering for the operation of robotic manipulators through kinematically singular configurations, *J. of Robotic Systems*, 5, 527, 1988.

- [38] McCarthy, J. M., *An Introduction to Theoretical Kinematics*, MIT Press, Cambridge, MA, 1990.
- [39] Nakamura, Y., *Advanced Robotics: Redundancy and Optimization*, Addison-Wesley, Reading, MA, 1991.
- [40] Nakamura, Y. and Hanafusa, H., Inverse kinematic solutions with singularity robustness for robot manipulator control, *ASME J. of Dynamic Systems, Measurement, and Control*, 108, 163, 1986.
- [41] Nakamura, Y., Hanafusa, H., and Yoshikawa, T., Task-priority based redundancy control of robot manipulators, *Int. J. of Robotics Research*, 6(2), 3, 1987.
- [42] Orin, D. E. and Schrader, W. W., Efficient computation of the Jacobian for robot manipulators, *Int. J. of Robotics Research*, 3(4), 66, 1984.
- [43] Paul, R. P., *Robot Manipulators: Mathematics, Programming, and Control*, MIT Press, Cambridge, MA, 1981.
- [44] Paul, R. P. and Zhang, H., Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation, *Int. J. of Robotics Research*, 5(2), 32, 1986.
- [45] Pieper, D. L., *The Kinematics of Manipulators Under Computer Control*, memo. AIM 72, Stanford Artificial Intelligence Laboratory, 1968.
- [46] Raghavan, M. and Roth, B., Inverse kinematics of the general 6R manipulator and related linkages, *ASME J. of Mechanical Design*, 115, 502, 1990.
- [47] Renaud, M., Calcul de la matrice jacobienne necessaire à la commande coordonnee d'un manipulateur, *Mechanism and Machine Theory*, 15, 81, 1980.
- [48] Samson, C., Le Borgne, M., and Espiau, B., *Robot Control: The Task Function Approach*, Oxford Engineering Science Series 22, Clarendon Press, Oxford, UK, 1991.
- [49] Sciavicco, L. and Siciliano, B., Coordinate transformation: A solution algorithm for one class of robots, *IEEE Trans. on Systems, Man, and Cybernetics*, 16, 550, 1986.
- [50] Sciavicco, L. and Siciliano, B., A solution algorithm to the inverse kinematic problem for redundant manipulators, *IEEE J. of Robotics and Automation*, 4, 403, 1988.
- [51] Sciavicco, L. and Siciliano, B., *Modeling and Control of Robot Manipulators*, McGraw-Hill, New York, NY, 1996.

- [52] Seraji, H., Configuration control of redundant manipulators: Theory and implementation, *IEEE Trans. on Robotics and Automation*, 5, 472, 1989.
- [53] Siciliano, B., Solving manipulator redundancy with the augmented task space method using the constraint Jacobian transpose, in *Prepr. 1992 IEEE Int. Conf. on Robotics and Automation — Tutorial on ‘Redundancy: Performance Indices, Singularities Avoidance, and Algorithmic Implementations’*, Nice, F, 1992.
- [54] Spong, M. W. and Vidyasagar, M., *Robot Dynamics and Control*, Wiley, New York, NY, 1989.
- [55] Tsai, L. W. and Morgan, A. P., Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods, *ASME J. of Mechanisms, Transmission, and Automation in Design*, 107, 189, 1985.
- [56] Vukobratović, M., *Introduction to Robotics*, Springer, Berlin, D, 1989.
- [57] Vukobratović, M. and Kirčanski, M., *Kinematics and Trajectory Synthesis of Manipulation Robots*, Scientific Fundamentals of Robotics 3, Springer, Berlin, D, 1986.
- [58] Wampler, C. W., Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods, *IEEE Trans. on Systems, Man, and Cybernetics*, 16, 93, 1986.
- [59] Whitney, D. E., Resolved motion rate control of manipulators and human prostheses, *IEEE Trans. on Man-Machine Systems*, 10, 47, 1969.
- [60] Yoshikawa, T., Manipulability of robotic mechanisms, *Int. J. of Robotics Research*, 4(2), 3, 1985.
- [61] Yuan, J. S.-C., Closed-loop manipulator control using quaternion feedback, *IEEE J. of Robotics and Automation*, 4, 434, 1988.