Chapter 6

Inverse kinematic model of serial robots

6.1. Introduction

The inverse kinematic model gives the joint velocities $\dot{\mathbf{q}}$ for a desired end-effector velocity $\dot{\mathbf{X}}$. This model is equivalent to the inverse differential model, which determines the differential variation of the joint variables $\mathbf{dq}$ corresponding to a given differential displacement of the end-effector coordinates $\mathbf{dX}$. We obtain the inverse kinematic model by solving a system of linear equations analytically or numerically. The analytical solutions, whenever they exist, offer much lower computational complexity than the numerical solutions, but all the singular cases must be considered separately on a case by case basis [Chevallereau 87]. Thus, the computational complexity of numerical methods is compensated by its generality in handling the regular, singular and redundant cases in a unified way.

In this chapter, we present the techniques used to develop an inverse kinematic model for the regular, singular and redundant cases. The analytical solution is developed for the regular case. The numerical methods presented for the other cases are based essentially on the pseudoinverse of the Jacobian matrix. Finally, we show how to take advantage of redundancy in the inverse kinematic problem using a minimum description of tasks. We assume that the reader is familiar with the techniques of solving linear equations, which are exposed in Appendix 4.

6.2. General form of the kinematic model

From equations [5.22] and [5.50], whatever the method used to describe the end-effector coordinates, the direct kinematic model can be expressed as:

$$\dot{\mathbf{X}} = \begin{bmatrix} \boldsymbol{\Omega}_p & \mathbf{0}_3 \\ \mathbf{0}_3 & \boldsymbol{\Omega}_r \end{bmatrix} \begin{bmatrix} {}^0\mathbf{A}_i & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^0\mathbf{A}_i \end{bmatrix} \begin{bmatrix} \mathbf{I}_3 & -{}^i\hat{\mathbf{L}}_{j,n} \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} {}^i\mathbf{J}_{n,j}\, \dot{\mathbf{q}} \qquad [6.1]$$

or in compact form as:

$$\dot{\mathbf{X}} = {}^0\mathbf{J}_x\, \dot{\mathbf{q}} \qquad [6.2]$$

Equation [6.1] can be written as:

$${}^i\dot{\mathbf{X}}_{n,j} = {}^i\mathbf{J}_{n,j}\, \dot{\mathbf{q}} \qquad [6.3]$$

with:

$${}^i\dot{\mathbf{X}}_{n,j} = \begin{bmatrix} \mathbf{I}_3 & {}^i\hat{\mathbf{L}}_{j,n} \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} {}^i\mathbf{A}_0 & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^i\mathbf{A}_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Omega}_p^{-1} & \mathbf{0}_3 \\ \mathbf{0}_3 & \boldsymbol{\Omega}_r^{+} \end{bmatrix} \dot{\mathbf{X}} \qquad [6.4]$$

We find in § 5.11 the expression of the pseudoinverse $\boldsymbol{\Omega}_r^+$ for different representations of the orientation, while $\boldsymbol{\Omega}_p^{-1} = \mathbf{I}_3$ if the Cartesian coordinates are used to describe the position.

Since the elements of ${}^i\mathbf{J}_{n,j}$ are simpler than those of ${}^0\mathbf{J}_x$, equation [6.3] is more appropriate for developing an analytical solution to the inverse kinematic problem. To simplify the notation, we will use the following form for both equations [6.2] and [6.3]:

$$\dot{\mathbf{X}} = \mathbf{J}\, \dot{\mathbf{q}} \qquad [6.5]$$

NOTE.– If $n < 6$, we cannot use the Jacobian matrix ${}^i\mathbf{J}_{n,j}$ systematically. The singularities of this matrix do not take into account the corresponding particular choice of the task coordinates [Borrel 86].

6.3. Inverse kinematic model for a regular case

In this case, the Jacobian matrix $\mathbf{J}$ is square and of full rank. Thus, it is possible to move the end-effector with finite velocity in any desired direction of the task space. The joint velocities can be evaluated using one of the following methods.

6.3.1. *First method*

We compute $\mathbf{J}^{-1}$, the inverse of $\mathbf{J}$, either numerically or analytically. Then, the joint velocity vector $\dot{\mathbf{q}}$ is obtained as:

$$\dot{\mathbf{q}} = \mathbf{J}^{-1}\,\dot{\mathbf{X}} \qquad [6.6]$$

If the matrix $\mathbf{J}$ has the following form:

$$\mathbf{J} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{B} & \mathbf{C} \end{bmatrix} \qquad [6.7]$$

the matrices $\mathbf{A}$ and $\mathbf{C}$ being square and invertible, it is easy to show that:

$$\mathbf{J}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} & \mathbf{0} \\ -\mathbf{C}^{-1}\mathbf{B}\mathbf{A}^{-1} & \mathbf{C}^{-1} \end{bmatrix} \qquad [6.8]$$

Consequently, the inverse of $\mathbf{J}$ reduces to the inverse of two matrices of smaller dimension. For a six degree-of-freedom robot with a spherical wrist, the general form of $\mathbf{J}$ is given by equation [6.7] where $\mathbf{A}$ and $\mathbf{C}$ are (3x3) matrices [Gorla 84].

6.3.2. *Second method*

In this method, instead of solving a linear system of n equations in n unknowns, the problem is reduced to solving two linear systems of equations of lower dimensions. In general, this technique requires less computational complexity. Let us take for example a six degree-of-freedom robot with a spherical wrist whose Jacobian matrix (see Example 5.3) can be written as:

$$\begin{bmatrix} \dot{\mathbf{X}}_a \\ \dot{\mathbf{X}}_b \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0}_3 \\ \mathbf{B} & \mathbf{C} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_a \\ \dot{\mathbf{q}}_b \end{bmatrix} \qquad [6.9]$$

$\mathbf{A}$ and $\mathbf{C}$ being (3x3) regular square matrices.

The solution $\dot{\mathbf{q}}$ is given by:

$$\begin{cases} \dot{\mathbf{q}}_a = \mathbf{A}^{-1}\,\dot{\mathbf{X}}_a \\ \dot{\mathbf{q}}_b = \mathbf{C}^{-1}\,[\dot{\mathbf{X}}_b - \mathbf{B}\,\dot{\mathbf{q}}_a] \end{cases} \qquad [6.10]$$

which, *a priori*, is simpler than that obtained by the first method.

• **Example 6.1.** Calculate the inverse kinematic model of the Stäubli RX-90 robot. The Jacobian ${}^3\mathbf{J}_6$ has been computed in Example 5.3. We develop the solutions according to equations [6.8] and [6.10].

i) first method. The inverses of **A** and **C** are respectively:

$$\mathbf{A}^{-1} = \begin{bmatrix} 0 & 0 & V1 \\ 0 & V3 & 0 \\ -1/RL4 & V2V3/RL4 & 0 \end{bmatrix}$$

$$\mathbf{C}^{-1} = \begin{bmatrix} V4 & 1 & -V5 \\ S4 & 0 & C4 \\ -C4/S5 & 0 & S4/S5 \end{bmatrix}$$

with:

$$V1 = \frac{1}{S23RL4 - C2D3}$$
$$V2 = -RL4 + S3D3$$
$$V3 = \frac{1}{C3D3}$$
$$V4 = C4\,\mathrm{cotg}5$$
$$V5 = S4\,\mathrm{cotg}5$$

Using equation [6.8], we obtain:

$${}^3\mathbf{J}_6^{-1} = \begin{bmatrix} 0 & 0 & V1 & 0 & 0 & 0 \\ 0 & V3 & 0 & 0 & 0 & 0 \\ -1/RL4 & V2V3/RL4 & 0 & 0 & 0 & 0 \\ -S4C5V7 & V5V6 & V8 & V4 & 1 & -V5 \\ C4/RL4 & -C4V6 & -S23S4V1 & S4 & 0 & C4 \\ S4V7 & -S4V6/S5 & S23C4V1/S5 & -C4/S5 & 0 & S4/S5 \end{bmatrix}$$

with:

$$V6 = \frac{S3}{C3RL4}$$
$$V7 = \frac{1}{S5RL4}$$
$$V8 = (-S23V4 - C23)V1$$

The computation of $\dot{\mathbf{q}}$ by equation [6.8] needs 18 additions, 47 multiplications/divisions and 8 sine/cosine functions;

ii) second method. We calculate successively $\dot{\mathbf{q}}_a$ and $\dot{\mathbf{q}}_b$:

$$\dot{\mathbf{q}}_a = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \begin{bmatrix} V1\dot{X}_3 \\ V3\dot{X}_2 \\ (-\dot{X}_1 + V2V3\dot{X}_2) / RL4 \end{bmatrix}$$

$$\dot{\mathbf{X}}_b - \mathbf{B}\,\dot{\mathbf{q}}_a = \begin{bmatrix} \dot{X}_{4'} \\ \dot{X}_{5'} \\ \dot{X}_{6'} \end{bmatrix} = \begin{bmatrix} \dot{X}_4 - S23\,\dot{q}_1 \\ \dot{X}_5 - C23\,\dot{q}_1 \\ \dot{X}_6 - \dot{q}_2 - \dot{q}_3 \end{bmatrix}$$

$$\dot{\mathbf{q}}_b = \begin{bmatrix} \dot{q}_4 \\ \dot{q}_5 \\ \dot{q}_6 \end{bmatrix} = \mathbf{C}^{-1} \begin{bmatrix} \dot{X}_{4'} \\ \dot{X}_{5'} \\ \dot{X}_{6'} \end{bmatrix} = \begin{bmatrix} C4\,\mathrm{cotg}5\,\dot{X}_{4'} + \dot{X}_{5'} - S4\,\mathrm{cotg}5\,\dot{X}_{6'} \\ S4\,\dot{X}_{4'} + C4\,\dot{X}_{6'} \\ (-C4\,\dot{X}_{4'} + S4\,\dot{X}_{6'}) / S5 \end{bmatrix}$$

This solution requires 12 additions, 22 multiplications/divisions and 8 sine/cosine functions.

6.4. Solution in the neighborhood of singularities

When the robot is non-redundant, the singular configurations are the roots of $\det(\mathbf{J}) = 0$. In the redundant case, they are given by the roots of $\det(\mathbf{J}\mathbf{J}^T) = 0$. Thus, singularities are identified by the rank deficiency of the matrix $\mathbf{J}$, which physically represents the inability of the robot to generate an arbitrary velocity in the task space. The neighborhood of a singular position is more precisely detected by using the singular values. In fact, the decrease of one or several singular values is generally more significant to indicate the vicinity of a singular configuration than that of examining the value of the determinant. In the neighborhood of these configurations, the use of the classical inverse of the Jacobian matrix will give excessive joint velocities. Since such high velocities are physically unrealizable, we cannot obtain an accurate motion.

The redundancy can be exploited to design robots that avoid singularities [Hollerbach 84b], [Luh 85a]. However, robots with revolute joints will have unavoidable singularities [Baillieul 84]. In § 6.5, we will see that redundancy may be exploited to go away from avoidable singularities [Baillieul 84]. An avoidable singularity is a singular configuration where the corresponding tool location can be reached with a different non-singular configuration.

6.4.1. *Use of the pseudoinverse*

The most widely proposed methods for solving the inverse kinematic problem near singularities involve the use of the pseudoinverse $\mathbf{J}^+$ of the matrix $\mathbf{J}$ (Appendix 4):

$$\dot{\mathbf{q}} = \mathbf{J}^+ \dot{\mathbf{X}} \qquad [6.11]$$

This solution, proposed by Whitney [Whitney 69], minimizes $\|\dot{\mathbf{q}}\|^2$ and $\|\dot{\mathbf{X}} - \mathbf{J}\dot{\mathbf{q}}\|^2$. Depending on $\dot{\mathbf{X}}$, the following cases are distinguished:

- $\dot{\mathbf{X}}$ belongs to $\mathcal{R}(\mathbf{J})$, representing the range space of $\mathbf{J}$: equation [6.11] gives an exact solution with zero error even though the inverse Jacobian $\mathbf{J}^{-1}$ is not defined;
- $\dot{\mathbf{X}}$ belongs to the subspace of the degenerated directions $\mathcal{R}(\mathbf{J})^{\perp}$: there are no joint velocities that can generate this velocity. In this case, the solution [6.11] gives $\dot{\mathbf{q}} = \mathbf{0}$. If the next desired velocity is also defined along this direction, the robot is blocked and it is necessary to define strategies to release it [Chevallereau 88];
- $\dot{\mathbf{X}}$ belongs to both $\mathcal{R}(\mathbf{J})$ and $\mathcal{R}(\mathbf{J})^{\perp}$: the solution [6.11] gives $\dot{\mathbf{q}}$, which only realizes the components belonging to $\mathcal{R}(\mathbf{J})$.

A major shortcoming of this method is that it produces discontinuous joint velocities near singularities [Wampler 86]. This can be seen by expressing the joint velocity solution in terms of singular value decomposition (§ 5.8.1). In fact, far from singularities, the joint velocities are given by:

$$\dot{\mathbf{q}} = \sum_{i=1}^{m} \frac{1}{\sigma_i} \mathbf{V}_i \, \mathbf{U}_i^T \dot{\mathbf{X}} \qquad [6.12]$$

While approaching a singularity, σ_{min} becomes small, leading to high joint velocities. At singularity, the smallest singular value σ_{min} becomes zero, consequently, it is not taken into account any more. The summation in equation [6.12] is carried out up to $m-1$, and the joint velocity $\dot{\mathbf{q}}$ decreases significantly.

NOTE.– Both $\|\dot{q}\|^2$ and $\|\dot{X}-J\dot{q}\|^2$ may contain elements with different units. However, using radians for the angles and meters for the distances gives good results for industrial robots of common size (1 to 2 meters reach).

6.4.2. *Use of the damped pseudoinverse*

A general approach to solving the problem of discontinuity of the pseudoinverse solution at a singular configuration is to use the damped least-squares method, which is known as the Levenberg-Marquardt stabilization method [Wampler 86], [Nakamura 87]. This solution minimizes the following expression:

$$\|\dot{X} - J\dot{q}\|^2 + \alpha^2 \|\dot{q}\|^2 \qquad [6.13]$$

where α is a constant.

This new criterion means that the end-effector tracking error is weighted against the norm of joint velocity by using the factor α, also known as the *damping factor*. This solution is typically obtained as the least-squares solution of the following system:

$$\begin{bmatrix} J \\ \alpha I_n \end{bmatrix} \dot{q} = \begin{bmatrix} \dot{X} \\ 0_{nx1} \end{bmatrix} \qquad [6.14]$$

which is given as:

$$\dot{q}_a = [J^T J + \alpha^2 I_n]^{-1} J^T \dot{X} \qquad [6.15]$$

When n > m, the following equivalent relation is easier to compute [Maciejewski 88]:

$$\dot{q}_a = J^T [J J^T + \alpha^2 I_m]^{-1} \dot{X} \qquad [6.16]$$

Using the singular value decomposition, the solution is written as:

$$\dot{q}_a = \sum_{i=1}^{m} \frac{\sigma_i}{\sigma_i^2+\alpha^2} V_i U_i^T \dot{X} \qquad [6.17]$$

If $\sigma_i >> \alpha$, then $\frac{\sigma_i}{\sigma_i^2+\alpha^2} \approx \frac{1}{\sigma_i}$. If $\sigma_i << \alpha$, then $\frac{\sigma_i}{\sigma_i^2+\alpha^2} \approx \frac{\sigma_i}{\alpha^2}$. The error due to the damping factor α in the joint coordinates is expressed as:

$$\dot{e}_q = \dot{q} - \dot{q}_a = \sum_{i=1}^{m} \frac{\alpha^2}{(\sigma_i^2+\alpha^2)\sigma_i} V_i \, U_i^T \dot{X} \qquad [6.18]$$

The error in $\dot{X}$ is obtained as:

$$\dot{e}_x = J \, \dot{e}_q = \sum_{i=1}^{m} \frac{\alpha^2}{\sigma_i^2+\alpha^2} U_i \, U_i^T \dot{X} \qquad [6.19]$$

The damping factor α limits the norm of the solution. However, at positions far away from singularities, no damping is needed. Thus, a trade-off must be found between the precision of the solution and the possibility of its realization.

Wampler [Wampler 86] proposes to use a fixed damping factor $\alpha = 0.003$, while Nakamura [Nakamura 86] suggests the computation of the damping factor as a function of the manipulability w (equation [5.38]) as follows:

$$\begin{cases} \alpha = \alpha_0 \left(1 - \frac{w}{w_0}\right)^2 & \text{if } w < w_0 \\ \alpha = 0 & \text{if } w \geq w_0 \end{cases} \qquad [6.20]$$

where α_0 is a positive constant and w_0 is a threshold, which defines the boundary of the neighborhood of singular points.

A more appropriate solution can be obtained by adjusting the value of α as a function of the smallest singular value σ_{min}, which is the exact measure of the neighborhood of a singular position. Maciejewski and Klein [Maciejewski 88] propose to compute the damping factor as follows:

$$\begin{cases} \alpha = \varepsilon^2 - \sigma_{min}^2 & \text{if } \sigma_{min} \leq \varepsilon \\ \alpha = 0 & \text{if } \sigma_{min} > \varepsilon \end{cases} \qquad [6.21]$$

where ε is a constant.

In [Maciejewski 88], we find an efficient method to estimate σ_{min}. In the damping least-squares method, the robot can stay blocked in a singular configuration if the desired velocity is along the degenerated directions, i.e. when (equations [5.30] and [5.31]):

$$\dot{\mathbf{X}} = \sum_{i=r+1}^{m} \mathbf{U}_i (\mathbf{U}_i^T \dot{\mathbf{X}}) \qquad [6.22]$$

where $r < m$ gives the rank of $\mathbf{J}$.

6.4.3. *Other approaches for controlling motion near singularities*

The kinematic model, which is a first order linearization, does not give an exact solution respecting the actuator constraints in the neighborhood of singularities. Some authors [Nielsen 91], [Chevallereau 98] have used the IGM or a kinematic model of higher order to determine the joint variables corresponding to a Cartesian motion passing through a singularity. Recently, it has been shown [Lloyd 96] that the end-effector could move along any specified path using a suitable time law.

To show the efficiency of such techniques, let us consider the case of a two degree-of-freedom planar robot in the singular configuration "extended arm". Let us suppose that we want to move the terminal point towards the origin along the x-axis (Figure 6.1a) (which is a degenerated direction for the kinematic model). It is easy to deduce from the kinematic model that a constant velocity motion along this direction is not feasible. However, a motion with a constant end-effector acceleration and a zero initial velocity can be proved realizable (Figure 6.1b) by developing the IGM up to the second order [Nielsen 91] or by using the second-order kinematic model [Chevallereau 98].

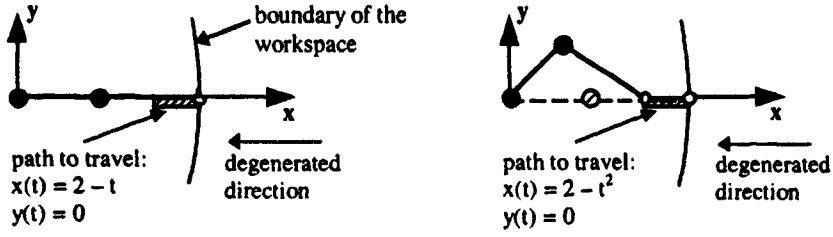
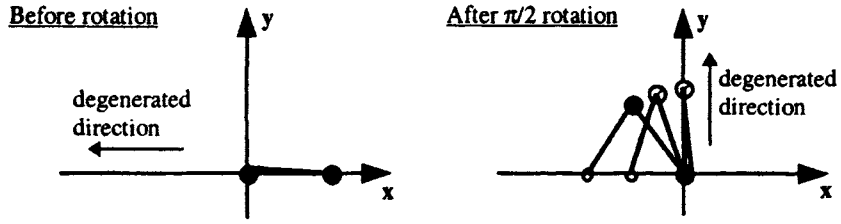


Figure 6.1. *Displacement along a degenerated direction*

In addition, Egeland and Spangelo [Egeland 91] showed that, in certain cases, a non-feasible path could become realizable after carrying out a specific motion in the null space of $\mathbf{J}$. This motion does not modify the end-effector coordinates but it modifies the degenerated direction. Let us illustrate this method for the two degree-of-freedom planar robot with identical link lengths. From the initial configuration "folded arm" of Figure 6.2a, it is not possible to track a trajectory along the x

direction. However, after a $\pi/2$ rotation of the first joint, which does not modify the terminal point coordinates but modifies the degenerated direction (Figure 6.2b), we can produce a velocity along the x-axis by using the kinematic model.



a) non feasible trajectory *b) feasible trajectory*

Figure 6.2. *Motion in the null space of J*

6.5. Inverse kinematic model of redundant robots

A robot manipulator is redundant when its number of degrees of freedom N is greater than the dimension of the workspace M. The difference (N – M) represents the degree of redundancy. In this case, the inverse kinematic model gives an infinite number of solutions. Consequently, secondary performance criteria can be optimized, such as:

- minimizing the norm of the joint velocities [Whitney 69];
- avoiding obstacles [Maciejewski 85], [Baillieul 86];
- avoiding singular configurations [Yoshikawa 84a];
- avoiding joint limits [Fournier 80], [Klein 84];
- minimizing driving joint torques [Baillieul 84], [Hollerbach 85].

When the end-effector coordinates are independent, we have $n = N$ and $m = M$. For a redundant mechanism, the Jacobian **J** is represented by an (mxn) matrix, with $n > m$. In the following sections, we present several approaches to solving the inverse kinematic problem of redundant robots.

6.5.1. *Extended Jacobian*

In this approach, we add n – m secondary linearly independent equations to the end-effector coordinates **X** [Baillieul 85], [Chang 86], [Nenchev 92]. These equations can represent either physical constraints on the robot or constraints related to the environment. They are written in the following general form:

$$\mathbf{X}_c = \mathbf{h}(\mathbf{q}) \qquad [6.23]$$

In this expression, $\mathbf{X}_c$ is an $((n-m)\,x1)$ vector whose elements are functions of $\mathbf{q}$. Differentiating equation [6.23] with respect to time gives:

$$\dot{\mathbf{X}}_c = \mathbf{J}_h\,\dot{\mathbf{q}} \qquad [6.24]$$

where $\mathbf{J}_h = \partial\mathbf{h}(\mathbf{q})/\partial\mathbf{q}$ is the $((n-m)xn)$ Jacobian matrix of $\mathbf{h}(\mathbf{q})$. Combining this equation with the kinematic model, we obtain an (nxn) extended Jacobian matrix $\mathbf{J}_a$ and a new velocity vector $\dot{\mathbf{X}}_a$ such that:

$$\dot{\mathbf{X}}_a = \mathbf{J}_a\,\dot{\mathbf{q}} \qquad [6.25]$$

with $\dot{\mathbf{X}}_a = \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\mathbf{X}}_c \end{bmatrix}$ and $\mathbf{J}_a = \begin{bmatrix} \mathbf{J} \\ \mathbf{J}_h \end{bmatrix}$.

If the extended Jacobian $\mathbf{J}_a$ is not singular, a unique solution for the joint velocity $\dot{\mathbf{q}}$ is obtained by inverting $\mathbf{J}_a$. We can use this technique to optimize the desired objective function $\phi(\mathbf{q})$ by taking $\mathbf{h}(\mathbf{q})$ such that:

$$h_i(\mathbf{q}) = 0 = (\eta_i)^T \nabla\phi \quad \text{for } i = 1, \ldots, n-m \qquad [6.26]$$

where the $(nx1)$ vectors η_i, for $i = 1, \ldots, n-m$, form a basis for the null space of $\mathbf{J}$, and $\nabla\phi$ is the gradient of ϕ.

Since the calculation of the basis of the null space of the Jacobian matrix must be carried out analytically, this method can be used only for systems with a small degree of redundancy. A solution to this problem can be found in [Klein 95].

The extended Jacobian method presents the following disadvantages:

- the choice of the $(n-m)$ additional relationships is not a trivial matter;
- the extended Jacobian $\mathbf{J}_a$ may be singular even though the end-effector Jacobian is of full rank. These configurations are called *artificial singularities* or *algorithmic singularities*.

A desirable property of this method is that it yields cyclic behavior, meaning that a closed path in the task space is always tracked by a closed path in the joint space. This is important because it allows one to judge the suitability of a trajectory after executing one cycle.

6.5.2. *Jacobian pseudoinverse*

The vast majority of research in the control of redundant robots has involved the resolution through the use of the pseudoinverse $\mathbf{J}^+$ of the Jacobian matrix:

$$\dot{\mathbf{q}} = \mathbf{J}^+ \dot{\mathbf{X}} \qquad [6.27]$$

This solution minimizes $\|\dot{\mathbf{q}}\|^2$. Because of this minimization property, the early hope of researchers [Whitney 69] was that singularities would automatically be avoided. It has been proved that, without modification, this approach does not avoid singularity [Baillieul 85]. Moreover, Klein and Huang [Klein 83] have pointed out that it does not produce cyclic behavior, which is a serious practical problem.

For these reasons, we generally add to the pseudoinverse solution another component belonging to the null space of the Jacobian, in order to realize the secondary objective function.

6.5.3. *Weighted pseudoinverse*

Since each joint has different limits and even different units, it may be interesting to weight the contribution of each joint in the objective function differently. This can be achieved by the use of the weighted pseudoinverse, which minimizes a criteria C such that:

$$C = \dot{\mathbf{q}}^T \mathbf{E} \dot{\mathbf{q}} \qquad [6.28]$$

When $\mathbf{J}$ is of full rank, the solution is given by:

$$\dot{\mathbf{q}} = \mathbf{J}_E^+ \dot{\mathbf{X}} \qquad [6.29]$$

with:

$$\mathbf{J}_E^+ = \mathbf{E}^{-1} \mathbf{J}^T (\mathbf{J} \mathbf{E}^{-1} \mathbf{J}^T)^{-1} \qquad [6.30]$$

Benoit et al. [Benoit 75] propose to take for $\mathbf{E}$ the inertia matrix of the robot (Chapter 9) in order to minimize the kinetic energy. Konstantinov et al. [Konstantinov 81] have used the weighted pseudoinverse to avoid the joint limits.

6.5.4. *Jacobian pseudoinverse with an optimization term*

One of the advantages of the pseudoinverse solution is the possibility to utilize the null space to optimize another objective function (beside that of $\|\dot{\mathbf{q}}\|^2$). In fact, the general solution of the linear system [6.5] is written as (Appendix 4):

$$\dot{\mathbf{q}} = \mathbf{J}^+ \dot{\mathbf{X}} + (\mathbf{I}_n - \mathbf{J}^+ \mathbf{J})\, \mathbf{Z} \qquad [6.31]$$

where $\mathbf{Z}$ is an arbitrary (nx1) vector in the $\dot{\mathbf{q}}$ space.

The second term on the right belongs to the null space of $\mathbf{J}$. It corresponds to a self-motion of the joints that does not move the end-effector. This term, which is called *homogeneous solution* or *optimization term*, can be used to optimize a desired function $\phi(\mathbf{q})$. In fact, taking $\mathbf{Z} = \alpha \nabla \phi$ where $\nabla \phi$ is the gradient of this function with respect to $\mathbf{q}$, minimizes the function $\phi(\mathbf{q})$ when $\alpha < 0$ and maximizes it when $\alpha > 0$. Equation [6.31] is rewritten as:

$$\dot{\mathbf{q}} = \mathbf{J}^+ \dot{\mathbf{X}} + \alpha(\mathbf{I}_n - \mathbf{J}^+ \mathbf{J})\, \nabla \phi \qquad [6.32]$$

with:

$$\nabla \phi = \left[\frac{\partial \phi}{\partial q_1} \; \cdots \; \frac{\partial \phi}{\partial q_n}\right]^T \qquad [6.33]$$

The value of α allows us to realize a trade-off between the minimization of $\|\dot{\mathbf{q}}\|^2$ and the optimization of $\phi(\mathbf{q})$. In the following sections, we present two examples of desired objective functions.

6.5.4.1. *Avoiding joint limits*

A practical solution to control a redundant robot is to keep the joint variables away from their limits $\mathbf{q}_{max}$ and $\mathbf{q}_{min}$. Let:

$$\mathbf{q}_{moy} = \frac{1}{2}(\mathbf{q}_{max} + \mathbf{q}_{min}) \qquad [6.34]$$

where $\mathbf{q}_{moy}$ is the mean value of the joint positions, and:

$$\Delta \mathbf{q} = \mathbf{q}_{max} - \mathbf{q}_{min} \qquad [6.35]$$

A possible scalar function, whose minimization generates a motion away from the joint limits, can be expressed in the following quadratic form [Fournier 80]:

$$\phi(\mathbf{q}) = \sum_{i=1}^{n} \left[\frac{q_i - q_{imoy}}{\Delta q_i}\right]^2 \qquad [6.36]$$

The division by Δq_i allows us to weight the contribution of each joint in $\phi(\mathbf{q})$ such that it varies between 0 and 1. The i^{th} element of the vector $\mathbf{Z}$ is written as (with $\alpha < 0$):

$$Z_i = \frac{\alpha \, \partial\phi(\mathbf{q})}{\partial q_i} = \frac{2\alpha \, (q_i - q_{imoy})}{\Delta q_i^2} \qquad [6.37]$$

NOTE.– If the mean position of a joint corresponds to a singular configuration, it is recommended to replace the corresponding value of q_{imoy} by another value.

About the criterion [6.36], Klein [Klein 84] pointed out that the quadratic form, used generally to solve optimization problems, does not always give the best solution to the desired objectives. To avoid joint limits in particular, the following form is more suitable:

$$\phi = \max \frac{|q_i - q_{imoy}|}{|\Delta qi|} \quad \text{for } i = 1, \ldots, n \qquad [6.38]$$

Introducing this criterion in equation [6.32] is however not as easy as the quadratic criterion. A solution consists of approximating the criterion [6.38] by a p-norm function defined as [Klein 83]:

$$\|\mathbf{q} - \mathbf{q}_{moy}\|_p = \left[\sum_{i=1}^{n} |q_i - q_{imoy}|^p\right]^{1/p} \qquad [6.39]$$

When p tends towards infinity, the corresponding p-norm meets the criterion [6.38]. However, sufficient approximation can be achieved by taking $p = 6$.

6.5.4.2. *Increasing the manipulability*

In § 5.8.2, we showed that the manipulability $w(\mathbf{q})$ of a robot manipulator (equation [5.38]) could be used as a measure of the ability of the mechanism to move its end-effector. At a singular point, w is minimum and is zero. In order to improve the manipulability of a structure, we can choose to maximize a scalar function ϕ such that:

$$\phi(\mathbf{q}) = \det [\mathbf{J}(\mathbf{q}) \mathbf{J}^T(\mathbf{q})] \qquad [6.40]$$

We calculate $\mathbf{Z}$ as indicated previously with $\alpha > 0$. Maximizing ϕ moves the robot away from the singular configurations.

NOTE.- Certain singular configurations are unavoidable [Baillieul 84]. This is the case if there is no other configuration that can yield the same end-effector location. For the three degree-of-freedom planar robot of Example 6.1, the unavoidable singularities correspond to the configurations where it is fully stretched out or folded up (Figure 6.3). The other singularities are avoidable and the robot can find other configurations to achieve them (Figure 6.4).

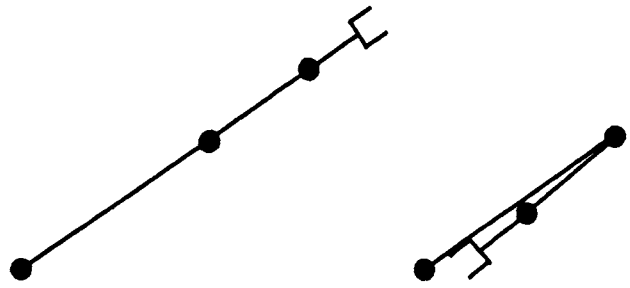


Figure 6.3. *Unavoidable singularities of a three degree-of-freedom planar robot*

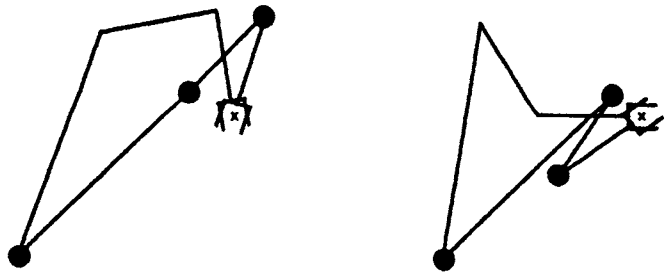


Figure 6.4. *Avoidable singularities of a three degree-of-freedom planar robot*

6.5.5. *Task-priority concept*

To solve the inverse kinematic model of redundant robots, Nakamura [Nakamura 87] introduced the concept of task priority, where a required task is divided into a primary task $\mathbf{X}_1$ of higher priority and a secondary task $\mathbf{X}_2$ of lower priority. These tasks are described by the following relationships:

$$\mathbf{X}_1 = \mathbf{f}_1(\mathbf{q}) \qquad [6.41]$$

$$\mathbf{X}_2 = \mathbf{f}_2(\mathbf{q}) \qquad [6.42]$$

Let m_1 and m_2 be the dimensions of $\mathbf{X}_1$ and $\mathbf{X}_2$ respectively. Differentiating equations [6.41] and [6.42] with respect to time gives:

$$\dot{\mathbf{X}}_1 = \mathbf{J}_1 \dot{\mathbf{q}} \qquad [6.43]$$

$$\dot{\mathbf{X}}_2 = \mathbf{J}_2 \dot{\mathbf{q}} \qquad [6.44]$$

where $\mathbf{J}_i = \partial \mathbf{f}_i(\mathbf{q})/\partial \mathbf{q}$ is the $(m_i x n)$ Jacobian matrix of the task $\mathbf{X}_i$. Using the pseudoinverse, the general solution of equation [6.43] is given by:

$$\dot{\mathbf{q}} = \mathbf{J}_1^{+} \dot{\mathbf{X}}_1 + (\mathbf{I}_n - \mathbf{J}_1^{+} \mathbf{J}_1)\, \mathbf{Z}_1 \qquad [6.45]$$

Substituting equation [6.45] into equation [6.44] yields:

$$\mathbf{J}_2 (\mathbf{I}_n - \mathbf{J}_1^{+} \mathbf{J}_1)\, \mathbf{Z}_1 = \dot{\mathbf{X}}_2 - \mathbf{J}_2 \mathbf{J}_1^{+} \dot{\mathbf{X}}_1 \qquad [6.46]$$

From this equation, the vector $\mathbf{Z}_1$ can be determined by using the pseudoinverse:

$$\mathbf{Z}_1 = \mathbf{J}_3^{+} [\dot{\mathbf{X}}_2 - \mathbf{J}_2 \mathbf{J}_1^{+} \dot{\mathbf{X}}_1] + (\mathbf{I}_n - \mathbf{J}_3^{+} \mathbf{J}_3)\, \mathbf{Z}_2 \qquad [6.47]$$

where $\mathbf{J}_3 = \mathbf{J}_2 (\mathbf{I}_n - \mathbf{J}_1^{+} \mathbf{J}_1)$ is an $(m_2 x n)$ matrix and $\mathbf{Z}_2$ is an arbitrary $(n x 1)$ vector chosen to satisfy the optimization criterion.

The joint velocity $\dot{\mathbf{q}}$ of the robot is obtained from equations [6.45] and [6.47]:

$$\dot{\mathbf{q}} = \mathbf{J}_1^{+} \dot{\mathbf{X}}_1 + (\mathbf{I}_n - \mathbf{J}_1^{+} \mathbf{J}_1)\, \{\mathbf{J}_3^{+} [\dot{\mathbf{X}}_2 - \mathbf{J}_2 \mathbf{J}_1^{+} \dot{\mathbf{X}}_1] + (\mathbf{I}_n - \mathbf{J}_3^{+} \mathbf{J}_3)\, \mathbf{Z}_2\} \qquad [6.48]$$

The interpretation of this method is illustrated in Figure 6.5.

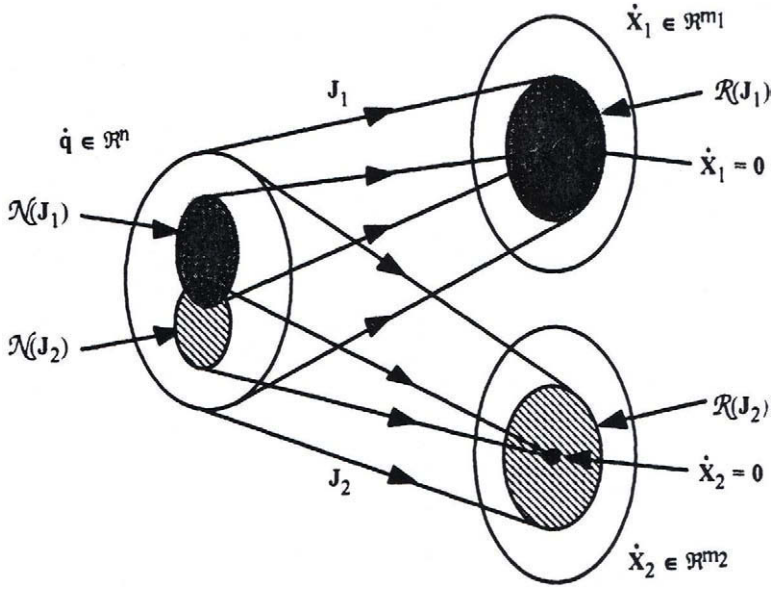


Figure 6.5. *Null space and range space of tasks X_1 and X_2 (from [Nakamura 87])*

6.6. Numerical calculation of the inverse geometric problem

When it is not possible to find a closed-form solution to the inverse geometric problem, we can use the differential model to compute an iterative numerical solution. To obtain the joint positions $\mathbf{q}^d$ corresponding to a desired location ${}^0\mathbf{T}_n^d$ of the terminal link, we proceed as follows:

- initialize $\mathbf{q}^c$ by the current joint configuration or by any random value within the joint domain of the robot;
- calculate the location of the terminal frame ${}^0\mathbf{T}_n^c$ corresponding to $\mathbf{q}^c$ using the direct geometric model;
- calculate the vectors of position error $\mathbf{dX}_p$ and rotation error $\mathbf{dX}_r$, representing the difference between the desired location ${}^0\mathbf{T}_n^d$ and the current location ${}^0\mathbf{T}_n^c$. Note that $\mathbf{dX}_p = \mathbf{dP}_n = \mathbf{P}_n^d - \mathbf{P}_n^c$ and $\mathbf{dX}_r = \mathbf{u}\,\alpha$, where the angle α and the unit vector $\mathbf{u}$ are obtained by solving the equation (§ 2.3.8): ${}^0\mathbf{A}_n^d = \mathbf{rot}(\mathbf{u}, \alpha)\,{}^0\mathbf{A}_n^c$, which can be written as ${}^0\mathbf{A}_n^d\,({}^0\mathbf{A}_n^c)^T = \mathbf{rot}(\mathbf{u}, \alpha)$;
- if $\mathbf{dX}_p$ and $\mathbf{dX}_r$ are sufficiently small, then $\mathbf{q}^d = \mathbf{q}^c$ and stop the calculation;

- to remain in the validity domain of the differential model, which is a first order expansion, we must introduce thresholds S_p and S_r on $\mathbf{dX}_p$ and $\mathbf{dX}_r$ respectively such that:

  - if $\|\mathbf{dX}_p\| > S_p$, then $\mathbf{dX}_p = \dfrac{\mathbf{dX}_p}{\|\mathbf{dX}_p\|} S_p$

  - if $\|\mathbf{dX}_r\| > S_r$, then $\mathbf{dX}_r = \dfrac{\mathbf{dX}_r}{\|\mathbf{dX}_r\|} S_r$

  The values 0.2 meter and 0.2 radian for these thresholds are acceptable for most of the industrial robots in view of their dimensions;
- calculate the Jacobian matrix ${}^0\mathbf{J}_n(\mathbf{q}^c)$ denoted as $\mathbf{J}$;
- calculate the joint variation $\mathbf{dq} = \mathbf{J}^+ \mathbf{dX}$. An optimization term in the null space of $\mathbf{J}$ can also be taken into account;
- update the current joint configuration: $\mathbf{q}^c = \mathbf{q}^c + \mathbf{dq}$;
- return to the second step.

This algorithm converges rapidly and can be executed in real time. If it does not converge within a relatively large number of iterations, we have to restart the calculation using a new random value $\mathbf{q}^c$; if no convergence occurs for many different values of $\mathbf{q}^c$, it can be stated that there is no solution.

6.7. Minimum description of tasks [Fournier 80], [Dombre 81]

In current robot controllers, the desired trajectory of the end-effector is described by a sequence of frames. However, in many industrial applications, it is not necessary to completely specify the location of the end-effector frame and the task could be described by a reduced number of coordinates. For example:

- when the manipulated object is symmetric: for a spherical object, it is not necessary to specify the orientation; likewise, the rotation of a cylindrical object about its axis can be left free;
- releasing an object into a container: if the end-effector is already above the container, only an approach distance has to be specified; the task is thus described by a translational component;
- transferring objects from one point to another with arbitrary orientation; the task can be described by three translational components;
- placing a cylindrical object on a conveyor: the only orientation constraint is that the principal axis of the cylinder is horizontal; if the end-effector is already above the conveyor, the task could be described by two components (one vertical translation and one rotation).

When the number of components of a task is less than the number of degrees of freedom of the robot, the robot is redundant with respect to the task. Consequently, an infinite number of solutions can be obtained to realize such tasks. This redundancy can be exploited to satisfy secondary optimization criteria (§ 6.5).

6.7.1. *Principle of the description*

The proposed description of task is minimal in the sense that it only constrains the degrees of freedom of the task that have a functional role. The formulation is based on the use of the contact conditions between usual surfaces (plane, cylinder, sphere) that describe usual mechanical joints (or pairing) (Table 6.1 and Figure 6.6). To these six joints, we add the composite revolute and prismatic joints, which have one degree of mobility (Figure 6.7), and the fixed rigid pairing, which has no degree of freedom.

The description of a task is realized by a sequence of virtual mechanical joints. The choice of a type of joint is dictated by the local constraints associated with the task.

Table 6.1. *Simple mechanical joints*

	Plane	**Cylinder**	**Sphere**
Plane	Plane contact	Line contact	Point contact
Cylinder		Cylindrical joint	Cylindrical groove joint
Sphere			Spherical joint

A practical description of the mechanical joint formulation consists of specifying the task in terms of contact between two simple geometric entities (point, line, plane), one belonging to the robot, the other to the environment [Dombre 85]. A spherical joint, for example, is specified by matching two points. In the same way, the revolute and prismatic joints will be specified with two simultaneous combinations of geometric elements. The choice is not unique: a revolute joint for example can be achieved either by a line-to-line contact and a point-to-plane contact simultaneously or by a line-to-line contact and a point-to-point contact.

This geometric description is particularly convenient for graphic programming of tasks. Figure 6.8 shows the example of a peg-in-hole assembly, realized with the CAD/CAM software package CATIA [Catia] in which this formulation was implemented for robotic application. The different steps are as follows:

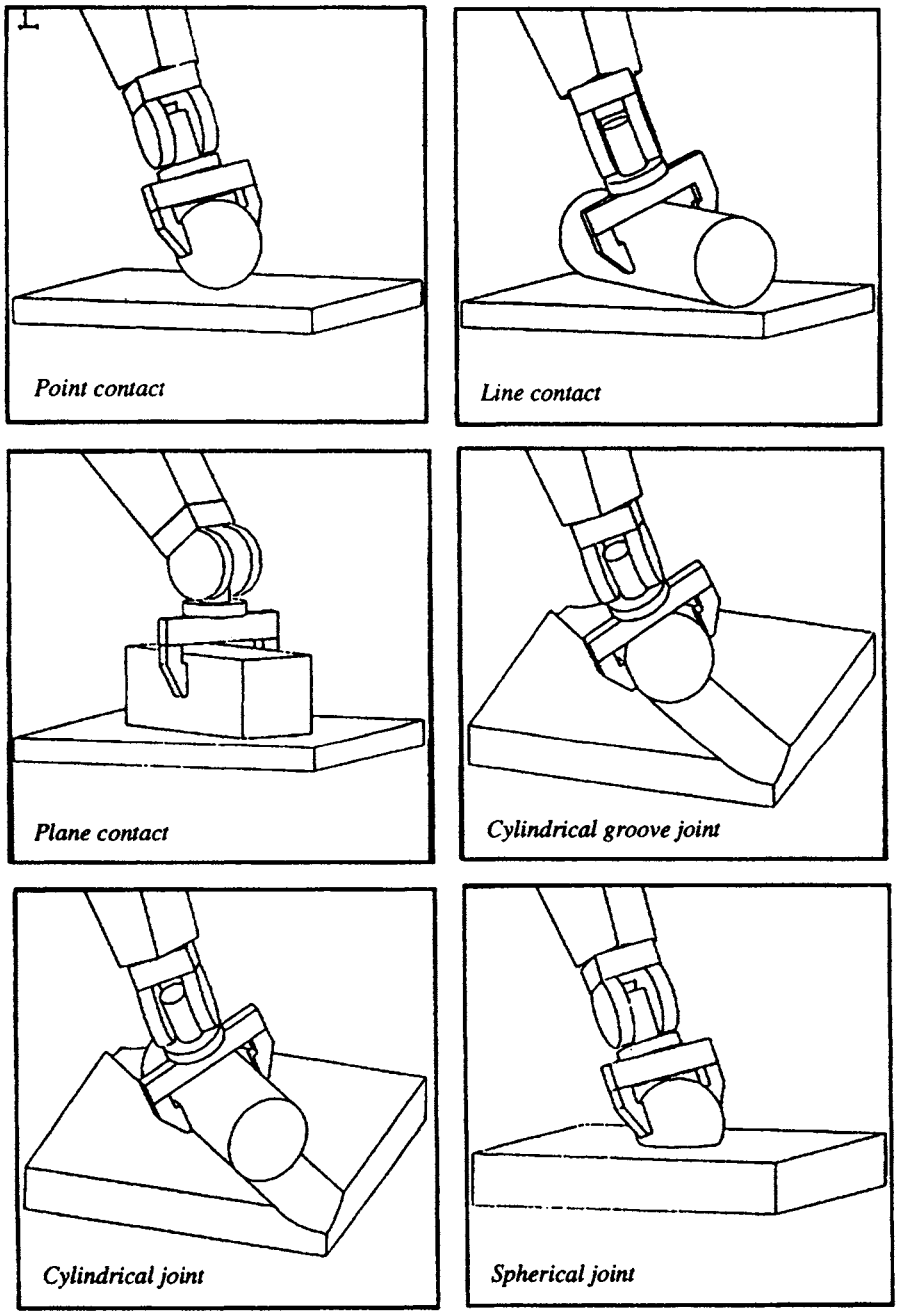


Figure 6.6. *Simple mechanical joints*

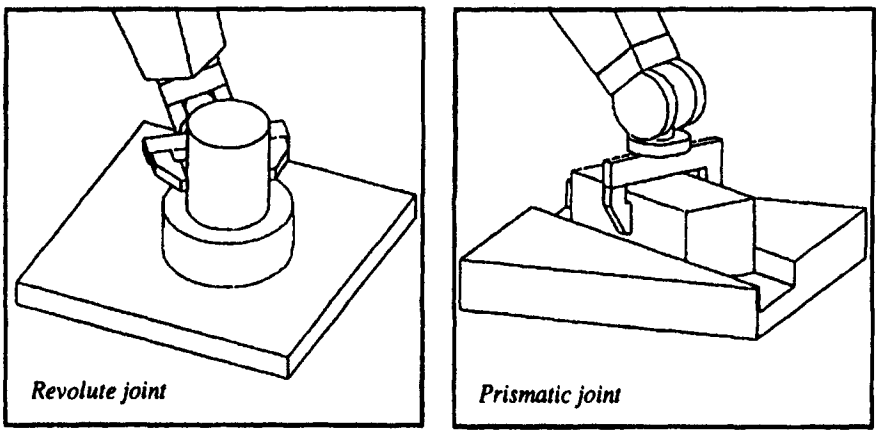


Figure 6.7. *Revolute and prismatic joints*

1) definition of a point-to-point contact (spherical joint) by selecting a point of the robot and a point of the environment (Figure 6.8a); after execution, the cylinder is positioned with an arbitrary orientation above the assembly site (Figure 6.8b);
2) definition of a line-to-line contact (cylindrical contact) by selecting a line of the robot and a line of the environment (Figure 6.8b); after execution, the axes of the hole and the peg are aligned (Figure 6.8c);
3) definition of a revolute joint by selecting a point and a line of the robot, and a point and a line of the environment (Figure 6.8c); after execution, the assembly task is completed (Figure 6.8d).

6.7.2. *Differential models associated with the minimum description of tasks*

To implement these types of tasks, we write the differential model of the location of frame R_E in the following form:

$$\begin{bmatrix} {}^0\mathbf{dP_E} \\ {}^0\boldsymbol{\delta}_E \end{bmatrix} = \begin{bmatrix} {}^0\mathbf{A}_n & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^0\mathbf{A}_n \end{bmatrix} \begin{bmatrix} {}^n\mathbf{dP_E} \\ {}^n\boldsymbol{\delta}_E \end{bmatrix} = \begin{bmatrix} {}^0\mathbf{A}_n & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^0\mathbf{A}_n \end{bmatrix} \begin{bmatrix} \mathbf{I}_3 & -{}^n\hat{\mathbf{P}}_E \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} {}^n\mathbf{dP}_n \\ {}^n\boldsymbol{\delta}_n \end{bmatrix}$$

$$= \begin{bmatrix} {}^0\mathbf{A}_n & -{}^0\mathbf{A}_n{}^n\hat{\mathbf{P}}_E \\ \mathbf{0}_3 & {}^0\mathbf{A}_n \end{bmatrix} {}^n\mathbf{J}_n\,\mathbf{dq} \qquad [6.49]$$

where ${}^n\mathbf{P}_E$ defines the origin of frame R_E referred to frame R_n.

The differential model of a virtual joint can be written as:

$$\mathbf{dX} = \mathbf{H}\,{}^{n}\mathbf{J}_{n}\,\mathbf{dq} \qquad [6.50]$$

where ${}^{n}\mathbf{J}_{n}$ and $\mathbf{H}$ are (6xn) and (cx6) matrices respectively, and c indicates the number of constraint equations of the task.

We will show in the following section how to determine $\mathbf{H}$ for the virtual joints [Dombre 81].

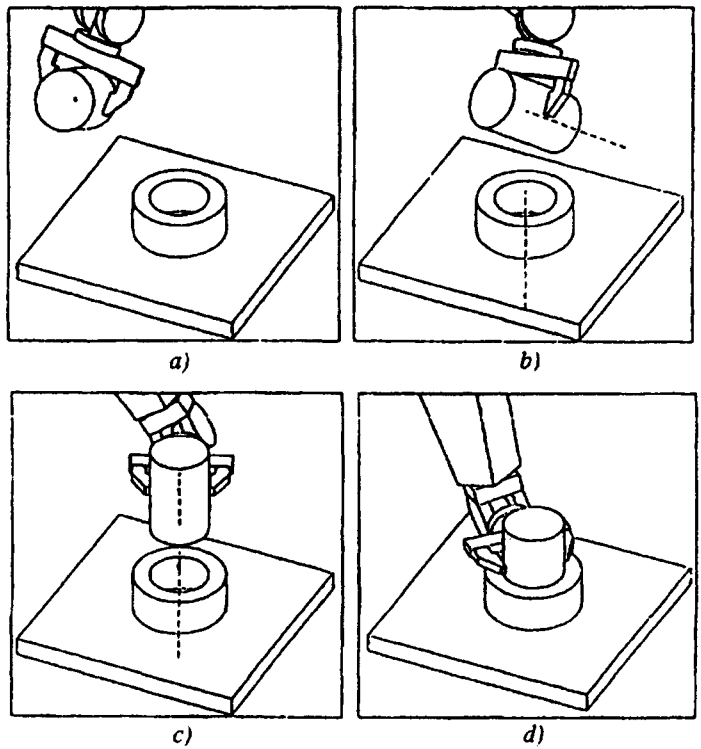


Figure 6.8. *Graphic programming of an assembly task with a minimum description*

6.7.2.1. *Point contact (point on plane)*

This joint drives a point O_E of the tool on any position on a plane Q (Figure 6.9). Let N be the unit vector normal to the plane Q and let O_D be an arbitrary point of Q. The necessary global displacement to realize the point contact is expressed in frame R_0 by:

$$r = {}^0N^T\,[{}^0P_D - {}^0P_E] \qquad [6.51]$$

where ${}^0P_D$ and ${}^0P_E$ define the coordinates of the points O_D and O_E in frame R_0.

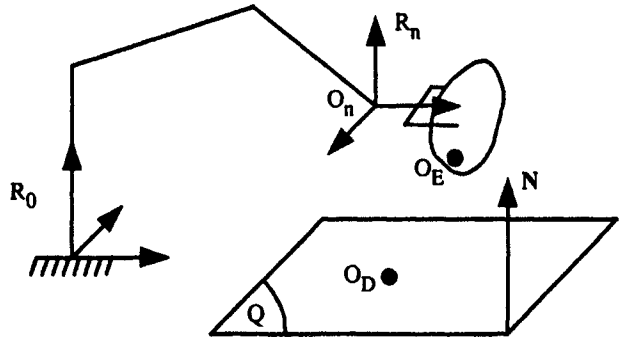


Figure 6.9. *Realization of point contact*

The displacement r is realized by a sequence of elementary displacements along a single direction such that (equation [6.49]):

$$\mathbf{dX} = dr = {}^0N^T\,{}^0dP_E = [\ {}^0N^T\,{}^0A_n \quad -{}^0N^T\,{}^0A_n\,{}^n\hat{P}_E\ ]\ {}^nJ_n\,dq$$

$$= {}^0N^T\,{}^0A_n\,[\ I_3 \quad -{}^n\hat{P}_E\ ]\ {}^nJ_n\,dq \qquad [6.52]$$

Expression [6.52] constitutes the differential model of the point contact. The matrix **H** is given by the row vector ${}^0N^T\,{}^0A_n\,[\ I_3 \quad -{}^n\hat{P}_E\ ]$.

6.7.2.2. *Line contact (line on plane)*

The equations of a line contact are derived from Figure 6.10. The line U_E is driven on plane Q without constraining its orientation in the plane. We can realize this joint by simultaneously carrying out a rotation and a translation [Dombre 88a]. However, it is more judicious to avoid the calculation of an angle by defining the task as driving two points O_{E1} and O_{E2} of U_E on plane Q. The joint is thus equivalent to two point contact. The corresponding differential model is written as:

$$\mathbf{dX} = \begin{bmatrix} dr_1 \\ dr_2 \end{bmatrix} = \begin{bmatrix} {}^0N^T\,{}^0A_n & -{}^0N^T\,{}^0A_n\,{}^n\hat{P}_{E1} \\ {}^0N^T\,{}^0A_n & -{}^0N^T\,{}^0A_n\,{}^n\hat{P}_{E2} \end{bmatrix} {}^nJ_n\,dq \qquad [6.53]$$

where **H** is a (2xn) matrix.

We can generalize this approach for the other joints where the j^{th} row of **H** takes the following general form:

$$\mathbf{H}_j = {}^0\mathbf{N}_j^T \, {}^0\mathbf{A}_n \, [\, \mathbf{I}_3 \quad -{}^n\hat{\mathbf{P}}_{Ej} \,] \qquad [6.54]$$

where ${}^0\mathbf{N}_j$ denotes the unit vector of the normal to the plane of the j^{th} point contact, and ${}^n\mathbf{P}_{E_j}$ is the vector of the coordinates of the tool point O_{Ej} with respect to frame R_n.

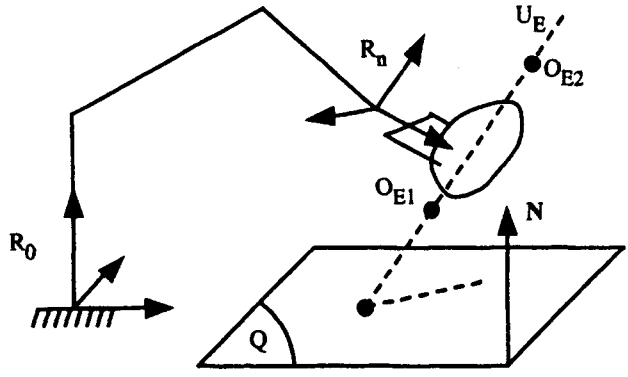


Figure 6.10. *Realization of line contact*

6.7.2.3. *Planar contact (plane on plane)*

This joint drives a plane Q_E attached to the tool on a plane Q_D of the environment, without orientation or position constraints (Figure 6.11). We select three non-aligned points O_{E1}, O_{E2} and O_{E3} in Q_E, then we carry out three simultaneous point contacts.

6.7.2.4. *Cylindrical groove joint (point on line)*

The cylindrical groove joint drives a point O_E of the tool on a line U_D of the environment. This is done by simultaneously realizing two point contacts of O_E on two arbitrary orthogonal planes Q_{D1} and Q_{D2} whose intersection is the line U_D (Figure 6.12).

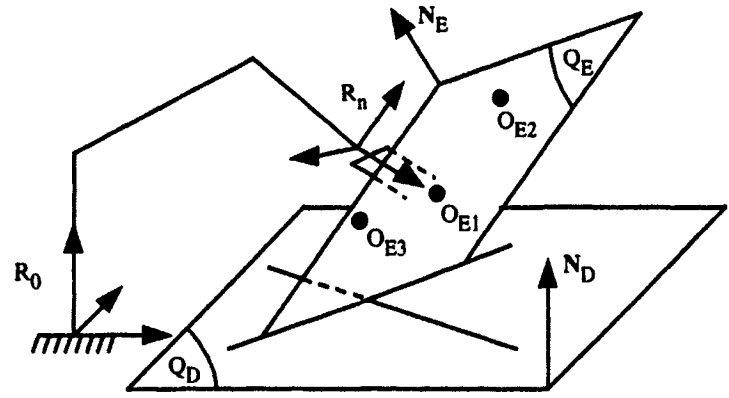


Figure 6.11. *Realization of a plane contact*

6.7.2.5. *Cylindrical joint (line on line)*

The task consists of aligning two lines U_E and U_D without position or orientation constraints along and about these lines (Figure 6.12). We define two arbitrary orthogonal planes Q_{D1} and Q_{D2} whose intersection is the line U_D and whose normals are N_{D1} and N_{D2} respectively. To realize a cylindrical joint, any two distinct points O_{E1} and O_{E2} of the line U_E are driven simultaneously on the planes Q_{D1} and Q_{D2}. In other words, the cylindrical joint corresponds to four point contacts.

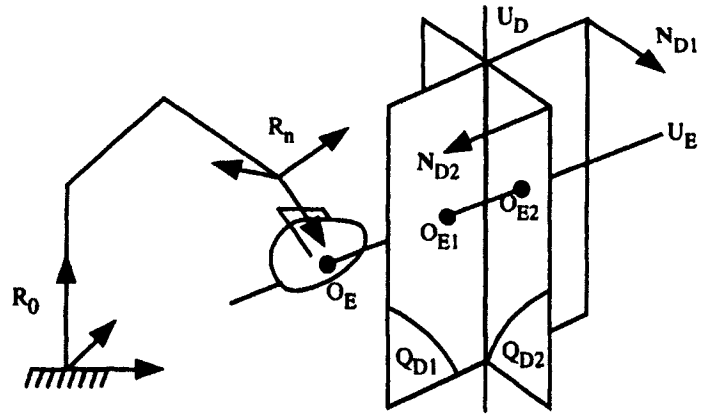


Figure 6.12. *Realization of cylindrical groove joint, cylindrical joint and revolute joint*

6.7.2.6. *Spherical joint (point on point)*

A spherical joint drives a point O_E of the tool on a point O_D of the environment without constraining the orientation of the tool. The task can be realized by three point contacts that drive O_E simultaneously on the planes Q_{D1}, Q_{D2} and Q_{D3}, which are parallel to the planes (y_0, z_0), (x_0, z_0), (x_0, y_0) and pass through the point O_D. The required displacements r_1, r_2 and r_3 are the components of the vector $\mathbf{O_E O_D}$ along the axes of frame R_0. The task is defined as:

$$\mathbf{dX} = \begin{bmatrix} dr_1 \\ dr_2 \\ dr_3 \end{bmatrix} = \begin{bmatrix} {}^0\mathbf{A}_n & -{}^0\mathbf{A}_n {}^n\hat{\mathbf{P}}_E \end{bmatrix} {}^n\mathbf{J}_n \, \mathbf{dq} \qquad [6.55]$$

6.7.2.7. *Revolute joint (line-point on line-point)*

A revolute joint (Figure 6.12) consists of aligning a line U_E of the tool with a line U_D of the environment while simultaneously driving a point O_E of U_E on a plane Q_D normal to U_D (not represented in the figure). Let O_{E1} and O_{E2} be any two distinct points on U_E. Similar to the cylindrical groove joint, let us consider that Q_{D1} and Q_{D2} are two arbitrary orthogonal planes whose intersection is the line U_D. The joint is thus equivalent to the simultaneous realization of five point contacts:

- driving the point O_{E1} on the planes Q_{D1} and Q_{D2};
- driving the point O_{E2} on the planes Q_{D1} and Q_{D2};
- driving the point O_E on the plane Q_D.

In practice, it is more convenient to describe the revolute joint by a line-to-line contact and a point-to-point contact. This choice leads to seven equations, and the rank of the matrix $\mathbf{H\,J}$ is five.

6.7.2.8. *Prismatic joint (plane-plane on plane-plane)*

A prismatic joint consists of aligning two lines of the tool with two geometrically compatible lines of the environment, and making a translation along an arbitrary axis. To simplify, we consider that the two lines are perpendicular and the displacement is carried out along one of these lines.

Let U_{E1} and U_{E2} be the two lines of the tool and let U_{D1} and U_{D2} be two compatible lines of the environment (Figure 6.13). Let us suppose that the free translation is along the line U_{D1}. Let Q_{D1a} and Q_{D1b} be two arbitrary orthogonal

planes whose intersection is U_{D1}, and O_{E1a} and O_{E1b} be any two distinct points on the line U_{E1}. We realize the prismatic joint by five point contacts:

- driving the point O_{E1a} on the planes Q_{D1a} and Q_{D2b};
- driving the point O_{E1b} on the planes Q_{D1a} and Q_{D2b};
- driving any point of U_{E2}, that is not the intersection of U_{E1} and U_{E2}, on the plane formed by the lines U_{D1} and U_{D2}.

Similar to the revolute joint case, it may be more convenient for the user to specify a prismatic joint using two plane-to-plane contacts. In this case, the number of equations is six.

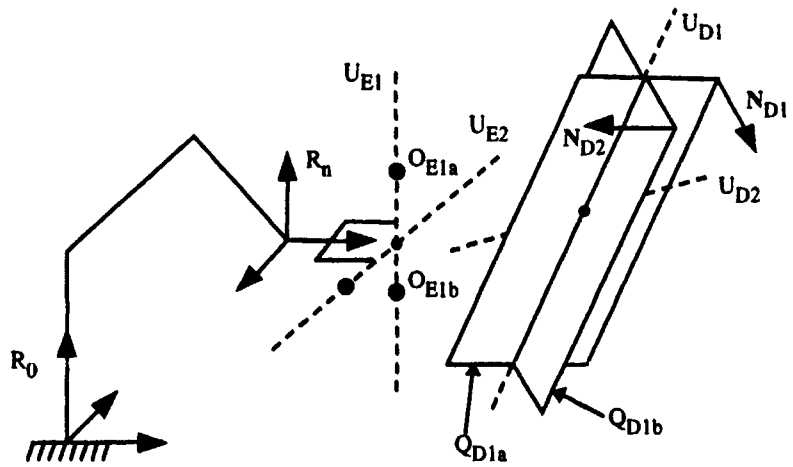


Figure 6.13. *Realization of a prismatic joint*

NOTES.–

- for the fixed rigid pairing, we use the complete description of $\mathbf{dX} = \mathbf{J}\,\mathbf{dq}$;
- Table 6.2 summarizes the specification of each virtual mechanical joint as well as the number of necessary equations.

Table 6.2. *Equivalence between virtual mechanical joints and geometric specification*

Type of joint	Elements of the tool	Elements of the environment	Number of independent equations	Total number of equations
Point contact	Point	Plane	1	1
Line contact	Line	Plane	2	2
Plane contact	Plane	Plane	3	3
Cylindrical groove	Point	Line	2	2
Spherical	Point	Point	3	3
Cylindrical	Line	Line	4	4
Revolute	Line-Point	Line-Point	5	7
Prismatic	Plane-Plane	Plane-Plane	5	6

6.8. Conclusion

In this chapter, we have studied the inverse kinematic model by considering the regular, singular and redundant cases. The solution may be obtained either analytically or numerically. The analytical solution can be used for simple robots in regular configurations, whereas the numerical methods are more general.

We have also shown how to reduce the functional degrees of freedom of the task using a description method based on the virtual mechanical joints formulation.

The redundancy, whether it is a built-in feature of the robot or the consequence of a minimum description of the task, can be used to optimize the trajectory generation of the mechanism. In this respect, the solution based on the pseudoinverse method proves to be very powerful. It allows us to realize secondary optimization functions such as keeping the joints away from their limits or improving the manipulability.