• Inverse Kinematics
  + Analytical Methods
  + Iterative (Differential) Methods
    ♦ Geometric and Analytical Jacobian
    ♦ Jacobian Transpose Method
    ♦ Pseudo-Inverse
    ♦ Pseudo-Inverse with Optimization
    ♦ Extended Jacobian Method

• Reading Assignment for Next Class
  ♦ See http://www-slab.usc.edu/courses/CS545
The Inverse Kinematics Problem

• Direct Kinematics

\[ x = f(q) \]

• Inverse Kinematics

\[ q = f^{-1}(x) \]

• Possible Problems of Inverse Kinematics
  + Multiple solutions
  + Infinitely many solutions
  + No solutions
  + No closed-form (analytical solution)
Analytical (Algebraic) Solutions

• Analytically invert the direct kinematics equations and enumerate all solution branches
  – Note: this only works if the number of constraints is the same as the number of degrees-of-freedom of the robot
  – What if not?
    + Iterative solutions
    + Invent artificial constraints

• Examples
  – 2DOF arm
  – See S&S textbook 2.11 ff
Analytical Inverse Kinematics of a 2 DOF Arm

- Inverse Kinematics:

\[
x = l_1 \cos q_1 + l_2 \cos (q_1 + q_2)
\]

\[
y = l_1 \sin q_1 + l_2 \sin (q_1 + q_2)
\]

\[
l = \sqrt{x^2 + y^2}
\]

\[
l_2^2 = l_1^2 + l_2^2 - 2l_1l \cos g
\]

\[
g = \arccos \frac{l^2 + l_1^2 - l_2^2}{2l_1l}
\]

\[
\frac{y}{x} = \tan \epsilon \quad q_1 = \arctan \frac{y}{x} - g
\]

\[
q_2 = \arctan \frac{y - l_1 \sin q_1}{x - l_1 \cos q_1} - q_1
\]
Iterative Solutions of Inverse Kinematics

• Resolved Motion Rate Control

\[ \dot{x} = J(q) \dot{q} \]
\[ \dot{q} = J(q)^\# \dot{x} \]

• Properties

+ Only holds for high sampling rates or low Cartesian velocities
+ “a local solution” that may be “globally” inappropriate
+ Problems with singular postures
+ Can be used in two ways:
  - As an instantaneous solutions of “which way to take “
  - As an “batch” iteration method to find the correct configuration at a target
Essential in Resolved Motion Rate Methods: The Jacobian

• Jacobian of direct kinematics:

\[
\mathbf{x} = f(\mathbf{q})
\]

\[
\frac{\partial \mathbf{x}}{\partial \mathbf{q}} = \frac{\partial f(\mathbf{q})}{\partial \mathbf{q}} = J(\mathbf{q})
\]

• In general, the Jacobian (for Cartesian positions and orientations) has the following form (geometrical Jacobian):

\[
J(\theta) = \begin{pmatrix}
    j_{p1} & j_{p2} & \cdots & j_{pn} \\
    \vdots & \ddots & \ddots & \vdots \\
    j_{\theta 1} & j_{\theta 2} & \cdots & j_{\theta n}
\end{pmatrix}
\]

where \[
\begin{pmatrix}
    j_{pi} \\
    j_{\theta i}
\end{pmatrix} = \begin{cases}
    \begin{bmatrix}
        z_{i-1} \\
        0
    \end{bmatrix} & \text{for a prismatic joint} \\
    z_{i-1} \times (\mathbf{p} - \mathbf{p}_{i-1}) & \text{for a revolute joint}
\end{cases}
\]

\(\mathbf{p}_i\) is the vector from the origin of the world coordinate system to the origin of the i-th link coordinate system, \(\mathbf{p}\) is the vector from the origin to the endeffector end, and \(z\) is the i-th joint axis (p.72 S&S)
The Jacobian Transpose Method

\[ Dq = a J^T(q) Dx \]

• Operating Principle:
  – Project difference vector \( \Delta x \) on those dimensions \( \Theta \) which can reduce it the most

• Advantages:
  – Simple computation (numerically robust)
  – No matrix inversions

• Disadvantages:
  – Needs many iterations until convergence in certain configurations (e.g., Jacobian has very small coefficients)
  – Unpredictable joint configurations
  – Non conservative
Jacobian Transpose
Derivation

Minimize cost function

\[ F = \frac{1}{2} (x_{\text{target}} - x)^T (x_{\text{target}} - x) \]

\[ = \frac{1}{2} (x_{\text{target}} - f(q))^T (x_{\text{target}} - f(q)) \]

with respect to \( q \) by gradient descent:

\[ Dq = -a \frac{\nabla F}{\nabla q} \]

\[ = a \frac{\nabla (x_{\text{target}} - x)^T \nabla f(q)}{\nabla q} \]

\[ = a J^T(q) \nabla x \]

\[ = a J^T(q)Dx \]
Jacobian Transpose
Geometric Intuition

\[ \begin{align*}
&x \\
&\theta_1 \\
&\theta_2 \\
&\theta_3 \\
\end{align*} \]
The Pseudo Inverse Method

\[ \Delta \theta = \alpha J^T(\theta)(J(\theta)J^T(\theta))^{-1} \Delta x = J^\# \Delta x \]

- Operating Principle:
  - Shortest path in \( \theta \)-space

- Advantages:
  - Computationally fast (second order method)

- Disadvantages:
  - Matrix inversion necessary (numerical problems)
  - Unpredictable joint configurations
  - Non conservative
Pseudo Inverse Method
Derivation

For a small step $\Delta x$, minimize with respect to $\Delta q$ the cost function:

$$F = \frac{1}{2} \Delta q^T \mathbf{D} \Delta q + 1^T (\Delta x - J(\mathbf{q}) \mathbf{D} \mathbf{q})$$

where $1^T$ is a vector of Lagrange multipliers.

Solution:

(1) $\frac{\partial F}{\partial 1} = 0 \quad \Delta x = J \Delta q$

(2) $\frac{\partial F}{\partial \mathbf{D} \mathbf{q}} = 0 \quad \mathbf{D} \mathbf{q} = J^T 1 \quad J \mathbf{D} \mathbf{q} = JJ^T 1$

$$1 = (JJ^T)^{-1} J \mathbf{D} \mathbf{q}$$

insert (1) into (2):

(3) $1 = (JJ^T)^{-1} \Delta x$

insertion of (3) into (2) gives the final result:

$\mathbf{D} \mathbf{q} = J^T 1 = J^T (JJ^T)^{-1} \Delta x$
Pseudo Inverse
Geometric Intuition

start posture = desired posture for optimization
Pseudo Inverse with explicit Optimization Criterion

\[ Dq = aJ^\#Dx + (I - J^\#J)(q_o - q) \]

• Operating Principle:
  - Optimization in null-space of Jacobian using a kinematic cost function
  \[ F = g(\theta), \quad e.g., \quad F = \sum_{i=1}^{d} (\theta_i - \theta_{i,0})^2 \]

• Advantages:
  - Computationally fast
  - Explicit optimization criterion provides control over arm configurations

• Disadvantages:
  - Numerical problems at singularities
  - Non conservative
Pseudo Inverse Method & Optimization Derivation

For a small step $\Delta x$, minimize with respect to $\Delta q$ the cost function:

$$ F = \frac{1}{2} (\Delta q + q - q_o)^T (\Delta q + q - q_o) + 1^T \Delta x - J(q) \Delta q $$

where $1^T$ is a vector of Lagrange multipliers.

Solution:

(1) $\frac{\partial F}{\partial 1} = 0 \quad \Delta x = JDq$

(2) $\frac{\partial F}{\partial Dq} = 0 \quad Dq = J^T 1 - (q - q_o) \quad JDq = J J^T 1 - J (q - q_o)$

$$ 1 = (JJ^T)^{-1} JDq + (JJ^T)^{-1} J (q - q_o) $$

Insert (1) into (2):

(3) $1 = (JJ^T)^{-1} \Delta x + (JJ^T)^{-1} J (q - q_o)$

Insertion of (3) into (2) gives the final result:

$$ Dq = J^T 1 - (q - q_o) = J^T (JJ^T)^{-1} \Delta x + J^T (JJ^T)^{-1} J (q - q_o) - (q - q_o) $$

$$ = J^\# \Delta x + (I - J^\# J) (q_o - q) $$
The Extended Jacobian Method

\[ \Delta \theta = \alpha (J^{\text{ext}}(\theta))^{-1} \Delta x^{\text{ext}}. \]

- **Operating Principle:**

  - Optimization in null-space of Jacobian using a kinematic cost function

  \[ F = g(\theta), \quad \text{e.g.,} \quad F = \sum_{i=1}^{d} (\theta_i - \theta_{i,0})^2 \]

- **Advantages:**

  - Computationally fast (second order method)

  - Explicit optimization criterion provides control over arm configurations

  - Numerically robust

  - Conservative

- **Disadvantages:**

  - Computationally expensive matrix inversion necessary (singular value decomposition)

  - Note: new and better ext. Jac. algorithms exist
Extended Jacobian Method
Derivation

The forward kinematics $x = f(q)$ is a mapping from a $n$-dimensional joint space to a $m$-dimensional Cartesian space. The singular value decomposition of the Jacobian of this mapping is:

$$J(q) = USV^T$$

The rows $[V_i]$ whose corresponding entry in the diagonal matrix $S$ is zero are the vectors which span the Null space of $J(q)$. There must be (at least) $n - m$ such vectors ($n + m$). Denote these vectors $n_i, i \in [1, n - m]$.

The goal of the extended Jacobian method is to augment the rank deficient Jacobian such that it becomes properly invertible. In order to do this, a cost function $F = g(q)$ has to be defined which is to be minimized with respect to $q$ in the Null space. Minimization of $F$ must always yield:

$$\frac{\partial F}{\partial q} = \frac{\partial g}{\partial q} = 0$$

Since we are only interested in zeroing the gradient in Null space, we project this gradient onto the Null space basis vectors:

$$G_i = \frac{\partial g}{\partial q} n_i$$

If all $G_i$ equal zero, the cost function $F$ is minimized in Null space. Thus we obtain the following set of equations which are to be fulfilled by the inverse kinematics solution:

$$f(q) \quad x$$

$$G_1 \quad 0$$

$$... \quad 0$$

$$z G_{n-m} \quad z 0 \quad ...$$

For an incremental step $dx$, this system can be linearized:

$$\frac{J(q)}{\frac{\partial G_i}{\partial q}} \quad \frac{\partial x}{0} \quad or \quad J^{ext} \frac{\partial q}{\partial q} = dx^{ext}.$$
Extended Jacobian
Geometric Intuition

start posture

X

Target

desired posture
for optimization
CHAPTER 3

DIFFERENTIAL KINEMATICS AND STATICS

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

3.1 GEOMETRIC JACOBIAN

Consider an \( n \)-degree-of-freedom manipulator. The direct kinematics equation can be written in the form

\[
T(q) = \begin{bmatrix}
R(q) & p(q) \\
0^T & 1
\end{bmatrix}
\]

where \( q = [q_1 \ldots q_n]^T \) is the vector of joint variables. Both end-effector position and orientation vary as \( q \) varies.

The goal of differential kinematics is to find the relationship between the joint velocities and the end-effector linear and angular velocities. In other words, it is desired to express the end-effector linear velocity \( \dot{p} \) and angular velocity \( \omega \) as a function of the
joint velocities $\dot{q}$ by means of the following relations:
\begin{align}
\dot{p} &= J_P(q)\dot{q} \\
\omega &= J_O(q)\dot{q};
\end{align}
(3.1)\hspace{1cm} (3.2)

notice that $v$ and $\omega$ are free vectors since their directions in space are prescribed but their points of application and lines of application are not prescribed.

In (3.1) $J_P$ is the $(3 \times n)$ matrix relative to the contribution of the joint velocities $\dot{q}$ to the end-effector linear velocity $\dot{p}$, while in (3.2) $J_O$ is the $(3 \times n)$ matrix relative to the contribution of the joint velocities $\dot{q}$ to the end-effector angular velocity $\omega$. In compact form, Eqs. (3.1) and (3.2) can be written as
\[ v = \begin{bmatrix} \dot{p} \\ \omega \end{bmatrix} = J(q)\dot{q} \]
(3.3)

which represents the manipulator differential kinematics equation. The $(6 \times n)$ matrix $J$ is the manipulator geometric Jacobian
\[ J = \begin{bmatrix} J_P \\ J_O \end{bmatrix}, \]
(3.4)

which in general is a function of the joint variables.

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

### 3.1.1 Derivative of a Rotation Matrix

The manipulator direct kinematics equation (2.40) describes the end-effector position and orientation, as a function of the joint variables, in terms of a position vector and a rotation matrix. Since the aim is to characterize the end-effector linear and angular velocity, it is worth considering first the derivative of a rotation matrix with respect to time.

Consider a time-varying rotation matrix $R = R(t)$. In view of the orthogonality of $R$, one has the relation
\[ R(t)R^T(t) = I \]
which, differentiated with respect to time, gives the identity
\[ \dot{R}(t)R^T(t) + R(t)\dot{R}^T(t) = 0. \]

Set
\[ S(t) = \dot{R}(t)R^T(t); \]
(3.5)

the $(3 \times 3)$ matrix $S$ is skew-symmetric since
\[ S(t) + S^T(t) = 0. \]
(3.6)
Postmultiplying both sides of (3.5) by $R(t)$ gives

$$\dot{R}(t) = S(t)R(t)$$

that allows expressing the time derivative of $R(t)$ as a function of $R(t)$ itself.

Eq. (3.7) relates the rotation matrix $R$ to its derivative by means of the skew-symmetric operator $S$ and has a meaningful physical interpretation. Consider a constant vector $p'$ and the vector $p(t) = R(t)p'$. The time derivative of $p(t)$ is

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

which, in view of (3.7), can be written as

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

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

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

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

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

which justifies the expression $S(t) = S(\omega(t))$.

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

$$RS(\omega)R^T = S(R\omega)$$

which will be useful later.

**Example 3.1.** Consider the elementary rotation matrix about axis $z$ given in (2.6). If $\alpha$ is a function of time, by computing the time derivative of $R_z(\alpha(t))$, Eq. (3.5) becomes

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

According to (3.8), it is

$$\omega = \begin{bmatrix} 0 & 0 & \dot{\alpha} \end{bmatrix}^T$$

that expresses the angular velocity of the frame about axis $z$. 
FIGURE 3.1
Characterization of generic link \( i \) of a manipulator.

With reference to Fig. 2.11, consider the coordinate transformation of a point \( P \) from frame 1 to frame 0; in view of (2.27), this is given by

\[
p^0 = o_1^0 + R_1^0 p^1 .
\]  
(3.10)

Differentiating (3.10) with respect to time gives

\[
\dot{p}^0 = \dot{o}_1^0 + \dot{R}_1^0 p^1 + \dot{R}_1^0 p^1 .
\]  
(3.11)

utilizing the expression of the derivative of a rotation matrix (3.7) and specifying the dependence on the angular velocity gives

\[
\dot{p}^0 = \dot{o}_1^0 + \dot{R}_1^0 p^1 + S(\omega_1^0)R_1^0 p^1 .
\]

Further, denoting the vector \( R_1^0 p^1 \) by \( r_1^0 \), it is

\[
\dot{p}^0 = \dot{o}_1^0 + \dot{R}_1^0 p^1 + \omega_1^0 \times r_1^0
\]  
(3.12)

which is the known form of the velocity composition rule.

Notice that, if \( p^1 \) is fixed in frame 1, it is

\[
\dot{p}^0 = \dot{o}_1^0 + \omega_1^0 \times r_1^0
\]  
(3.13)

since \( \dot{p}^1 = 0 \).

3.1.2 Link Velocity

Consider the generic link \( i \) of a manipulator with an open kinematic chain. According to the Denavit-Hartenberg convention adopted in the previous chapter, link \( i \) connects joints \( i \) and \( i + 1 \); frame \( i \) is attached to link \( i \) and has origin along joint \( i + 1 \) axis, while frame \( i - 1 \) has origin along joint \( i \) axis (Fig. 3.1).
Let \( p_{i-1} \) and \( p_i \) be the position vectors of the origins of frames \( i - 1 \) and \( i \), respectively. Also, let \( r_{i-1,i}^{i-1} \) denote the position of the origin of frame \( i \) with respect to frame \( i - 1 \) expressed in frame \( i - 1 \). According to the coordinate transformation (3.10), one can write:

\[
p_i = p_{i-1} + R_{i-1} r_{i-1,i}^{i-1}.
\]

Then, by virtue of (3.12), it is

\[
\dot{p}_i = \dot{p}_{i-1} + R_{i-1} \dot{r}_{i-1,i}^{i-1} + \omega_{i-1} \times R_{i-1} r_{i-1,i}^{i-1}
\]

\[
= \dot{p}_{i-1} + v_{i-1,i} + \omega_{i-1} \times r_{i-1,i}
\]

which gives the expression of the linear velocity of link \( i \) as a function of the translational and rotational velocities of link \( i - 1 \). Note that \( v_{i-1,i} \) denotes the velocity of the origin of frame \( i \) with respect to the origin of frame \( i - 1 \), expressed in the base frame.

Concerning link angular velocity, it is worth starting from the rotation composition

\[
R_i = R_{i-1} R_{i}^{i-1};
\]

from (3.7), its time derivative can be written as

\[
S(\omega_i) R_i = S(\omega_{i-1}) R_i + R_{i-1} S(\omega_{i-1,i}^{i-1}) R_i^{i-1}
\]

(3.15)

where \( \omega_{i-1,i}^{i-1} \) denotes the angular velocity of frame \( i \) with respect to frame \( i - 1 \) expressed in frame \( i - 1 \). From (2.4), the second term on the right-hand side of (3.15) can be rewritten as

\[
R_{i-1} S(\omega_{i-1,i}^{i-1}) R_i^{i-1} = R_{i-1} S(\omega_{i-1,i}^{i-1}) R_{i-1}^T R_{i-1} R_i^{i-1};
\]

in view of property (3.9), it is

\[
R_{i-1} S(\omega_{i-1,i}^{i-1}) R_i^{i-1} = S(R_{i-1} \omega_{i-1,i}^{i-1}) R_i;
\]

Then, Eq. (3.15) becomes

\[
S(\omega_i) R_i = S(\omega_{i-1}) R_i + S(R_{i-1} \omega_{i-1,i}^{i-1}) R_i
\]

leading to the result

\[
\omega_i = \omega_{i-1} + R_{i-1} \omega_{i-1,i}^{i-1}
\]

\[
= \omega_{i-1} + \omega_{i-1,i},
\]

(3.16)

which gives the expression of the angular velocity of link \( i \) as a function of the angular velocities of link \( i - 1 \) and of link \( i \) with respect to link \( i - 1 \).

---

1 In the following, the indication of superscript '0' is omitted for quantities expressed in the base frame.
Eqs. (3.14) and (3.16) attain different expressions depending on the type of joint (prismatic or revolute).

**Prismatic Joint.** Since orientation of frame \(i\) with respect to frame \(i-1\) does not vary by moving joint \(i\), it is
\[
\mathbf{\omega}_{i-1,i} = 0.
\] (3.17)
Further, the linear velocity is
\[
v_{i-1,i} = \dot{d}_i z_{i-1}
\] (3.18)
where \(z_{i-1}\) is the unit vector of joint \(i\) axis. Hence, the expressions of angular velocity (3.16) and linear velocity (3.14) respectively become
\[
\mathbf{\omega}_i = \mathbf{\omega}_{i-1}
\] (3.19)
\[
\mathbf{\dot{p}}_i = \mathbf{\dot{p}}_{i-1} + \dot{d}_i z_{i-1} + \mathbf{\omega}_i \times \mathbf{r}_{i-1,i}
\] (3.20)
where the relation \(\mathbf{\omega}_i = \mathbf{\omega}_{i-1}\) has been exploited to derive (3.20).

**Revolute Joint.** For the angular velocity it is obviously
\[
\mathbf{\omega}_{i-1,i} = \mathbf{\dot{\theta}}_i z_{i-1},
\] (3.21)
while for the linear velocity it is
\[
v_{i-1,i} = \omega_{i-1,i} \times \mathbf{r}_{i-1,i}
\] (3.22)
due to the rotation of frame \(i\) with respect to frame \(i-1\) induced by the motion of joint \(i\). Hence, the expressions of angular velocity (3.16) and linear velocity (3.14) respectively become
\[
\mathbf{\omega}_i = \mathbf{\omega}_{i-1} + \mathbf{\dot{\theta}}_i z_{i-1}
\] (3.23)
\[
\mathbf{\dot{p}}_i = \mathbf{\dot{p}}_{i-1} + \mathbf{\omega}_i \times \mathbf{r}_{i-1,i}
\] (3.24)
where (3.16) has been exploited to derive (3.24).

### 3.1.3 Jacobian Computation

Let the Jacobian in (3.4) be partitioned into the \((3 \times 1)\) column vectors as:
\[
\mathbf{J} = \begin{bmatrix}
\mathbf{J}_{P1} & \cdots & \mathbf{J}_{Pn} \\
\mathbf{J}_{O1} & \cdots & \mathbf{J}_{On}
\end{bmatrix}
\] (3.25)
The term \(\mathbf{\dot{q}}_i \mathbf{J}_{Pi}\) represents the contribution of single joint \(i\) to the end-effector linear velocity, while the term \(\mathbf{\dot{q}}_i \mathbf{J}_{Oi}\) represents the contribution of single joint \(i\) to the end-effector angular velocity. In order to compute the Jacobian it is convenient to compute
the single contributions by distinguishing the case of a prismatic joint \( q_i = \dot{d}_i \) from the case of a revolute joint \( q_i = \dot{\theta}_i \).

For the contribution to the angular velocity:

- If joint \( i \) is prismatic, from (3.17) it is

\[
\dot{q}_i J_\Omega_i = 0
\]

and then

\[
J_\Omega_i = 0.
\]

- If joint \( i \) is revolute, from (3.21) it is

\[
\dot{q}_i J_\Omega_i = \dot{\theta}_i z_{i-1}
\]

and then

\[
J_\Omega_i = z_{i-1}.
\]

For the contribution to the linear velocity:

- If joint \( i \) is prismatic, from (3.18) it is

\[
\dot{q}_i J_p_i = \dot{d}_i z_{i-1}
\]

and then

\[
J_p_i = z_{i-1}.
\]

- If joint \( i \) is revolute, observing that the contribution to the linear velocity is to be computed with reference to the origin of the end-effector frame (Fig. 3.2), it is

\[
\dot{q}_i J_p_i = \omega_{i-1,i} \times r_{i-1,n}
= \dot{\theta}_i z_{i-1} \times (p - p_{i-1})
\]

and then

\[
J_p_i = z_{i-1} \times (p - p_{i-1}).
\]

In sum, it is:

\[
\begin{bmatrix}
J_p_i \\
J_\Omega_i
\end{bmatrix} = \begin{bmatrix}
z_{i-1} \\
0 \\
z_{i-1} \times (p - p_{i-1}) \\
z_{i-1}
\end{bmatrix}
\begin{cases}
\text{for a prismatic joint} \\
\text{for a revolute joint.}
\end{cases}
\]
Eqs. (3.26) allow Jacobian computation in a simple, systematic way on the basis of direct kinematics relations. In fact, the vectors $z_{i-1}$, $p$, and $p_{i-1}$ are all functions of the joint variables. In particular:

- $z_{i-1}$ is given by the third column of the rotation matrix $R_{i-1}^0$, i.e.,
  \[ z_{i-1} = R_{i-1}^0(q_1) \cdots R_{i-1}^{i-2}(q_{i-1})z_0 \]  
  \[ (3.27) \]
  where $z_0 = [0 \ 0 \ 1]^T$ allows selecting the third column.

- $p$ is given by the first three elements of the fourth column of the transformation matrix $T_n^0$, i.e., by expressing $\tilde{p}$ in the $(4 \times 1)$ homogeneous form
  \[ \tilde{p} = A_1^0(q_1) \cdots A_{n-1}^{n-1}(q_n)\tilde{p}_0 \]  
  \[ (3.28) \]
  where $\tilde{p}_0 = [0 \ 0 \ 0 \ 1]^T$ allows selecting the fourth column.

- $p_{i-1}$ is given by the first three elements of the fourth column of the transformation matrix $T_{i-1}^0$, i.e., it can be extracted from
  \[ \tilde{p}_{i-1} = A_1^0(q_1) \cdots A_{i-1}^{i-2}(q_{i-1})\tilde{p}_0. \]  
  \[ (3.29) \]

Remarkably, the above equations can be conveniently used to compute the translational and rotational velocities of any point along the manipulator structure, as long as the direct kinematics functions relative to that point are known.

Finally, notice that the Jacobian matrix depends on the frame in which the end-effector velocity is expressed. The above equations allow computation of the geometric Jacobian with respect to the base frame. If it is desired to represent the Jacobian
in a different frame \( u \), it is sufficient to know the relative rotation matrix \( R^u \). The relationship between velocities in the two frames is

\[
\begin{bmatrix}
\dot{p}^u \\
\omega^u
\end{bmatrix} =
\begin{bmatrix}
R^u & O \\
O & R^u
\end{bmatrix}
\begin{bmatrix}
\dot{p} \\
\omega
\end{bmatrix},
\]

which, substituted in (3.3), gives

\[
\begin{bmatrix}
\dot{p}^u \\
\omega^u
\end{bmatrix} =
\begin{bmatrix}
R^u & O \\
O & R^u
\end{bmatrix} J \dot{q}.
\]

On the assumption of a time-invariant frame \( u \), it is

\[
J^u =
\begin{bmatrix}
R^u & O \\
O & R^u
\end{bmatrix} J,
\]

(3.30)

where \( J^u \) denotes the geometric Jacobian in frame \( u \).

### 3.2 JACOBIAN OF TYPICAL MANIPULATOR STRUCTURES

In the following, the Jacobian is computed for some of the typical manipulator structures of the previous chapter.

#### 3.2.1 Three-Link Planar Arm

In this case, from (3.26) the Jacobian is

\[
J(q) =
\begin{bmatrix}
z_0 \times (p - p_0) & z_1 \times (p - p_1) & z_2 \times (p - p_2)
\end{bmatrix}
\]

Computation of the position vectors of the various links gives

\[
p_0 =
\begin{bmatrix}
0 \\
0 \\
0
\end{bmatrix}
\quad p_1 =
\begin{bmatrix}
a_1 c_1 \\
a_1 s_1 \\
0
\end{bmatrix}
\quad p_2 =
\begin{bmatrix}
a_1 c_1 + a_2 c_{12} \\
a_1 s_1 + a_2 s_{12} \\
0
\end{bmatrix}
\]

\[
p =
\begin{bmatrix}
a_1 c_1 + a_2 c_{12} + a_3 c_{123} \\
a_1 s_1 + a_2 s_{12} + a_3 s_{123} \\
0
\end{bmatrix},
\]

while computation of the unit vectors of revolute joint axes gives

\[
z_0 = z_1 = z_2 =
\begin{bmatrix}
0 \\
0 \\
0
\end{bmatrix}
\]
since they are all parallel to axis \( z_0 \). From (3.25) it is

\[
J = \begin{bmatrix}
-a_1 s_1 - a_2 s_{12} - a_3 s_{123} & -a_2 s_{12} - a_3 s_{123} & -a_3 s_{123} \\
\frac{a_1 c_1 + a_2 c_{12} + a_3 c_{123}}{} & \frac{a_2 c_{12} + a_3 c_{123}}{} & \frac{a_3 c_{123}}{}
\end{bmatrix}.
\]  

(3.31)

In the Jacobian (3.31), only the three nonnull rows are relevant (the rank of the matrix is at most 3); these refer to the two components of linear velocity along axes \( x_0, y_0 \) and the component of angular velocity about axis \( z_0 \). This result can be derived by observing that three degrees of mobility allow specification of at most three end-effector variables; \( v_z, \omega_z \), and \( \omega_y \) are always null for this kinematic structure. If orientation is of no concern, the \((2 \times 3)\) Jacobian for the positional part can be derived by considering just the first two rows, i.e.,

\[
J_P = \begin{bmatrix}
-a_1 s_1 - a_2 s_{12} - a_3 s_{123} & -a_2 s_{12} - a_3 s_{123} & -a_3 s_{123} \\
\frac{a_1 c_1 + a_2 c_{12} + a_3 c_{123}}{} & \frac{a_2 c_{12} + a_3 c_{123}}{} & \frac{a_3 c_{123}}{}
\end{bmatrix}.
\]  

(3.32)

### 3.2.2 Anthropomorphic Arm

In this case, from (3.26) the Jacobian is

\[
J = \begin{bmatrix}
z_0 \times (p - p_0) & z_1 \times (p - p_1) & z_2 \times (p - p_2)
z_0 & z_1 & z_2
\end{bmatrix}.
\]

Computation of the position vectors of the various links gives

\[
p_0 = p_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad p_2 = \begin{bmatrix} a_2 c_1 c_2 \\ a_2 s_1 c_2 \\ a_2 s_2 \end{bmatrix}
\]

\[
p = \begin{bmatrix} c_1 (a_2 c_2 + a_3 c_{23}) \\ s_1 (a_2 c_2 + a_3 c_{23}) \\ a_2 s_2 + a_3 s_{23} \end{bmatrix},
\]

while computation of the unit vectors of revolute joint axes gives

\[
z_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad z_1 = z_2 = \begin{bmatrix} s_1 \\ -c_1 \\ 0 \end{bmatrix}.
\]

From (3.25) it is

\[
J = \begin{bmatrix}
-s_1 (a_2 c_2 + a_3 c_{23}) & -c_1 (a_2 s_2 + a_3 s_{23}) & -a_3 c_1 s_{23} \\
c_1 (a_2 c_2 + a_3 c_{23}) & -s_1 (a_2 s_2 + a_3 s_{23}) & -a_3 s_1 s_{23} \\
0 & a_2 c_2 + a_3 c_{23} & a_3 c_{23} \\
0 & s_1 & s_1 \\
0 & -c_1 & -c_1 \\
1 & 0 & 0
\end{bmatrix}.
\]  

(3.33)
Only three of the six rows of the Jacobian (3.33) are linearly independent. Having three degrees of mobility only, it is worth considering the upper (3 × 3) block of the Jacobian

\[
J_P = \begin{bmatrix}
-s_1(a_2c_2 + a_2c_{23}) & -c_1(a_2s_2 + a_3s_{23}) & -a_3c_1s_{23} \\
-1(a_2c_2 + a_3c_{23}) & -s_1(a_2s_2 + a_3s_{23}) & -a_3s_1s_{23} \\
0 & a_2c_2 + a_3c_{23} & a_3c_{23}
\end{bmatrix}
\] (3.34)

that describes the relationship between the joint velocities and the end-effector linear velocity. This structure does not allow obtaining arbitrary angular velocity \(\omega\); in fact, the two components \(\omega_x\) and \(\omega_y\) are not independent \((s_1\omega_y = -c_1\omega_x)\).

### 3.2.3 Stanford Manipulator

In this case, from (3.26) it is

\[
J = \begin{bmatrix}
 z_0 \times (p - p_0) & z_1 \times (p - p_1) & z_2 \\
 z_0 & z_1 & 0 \\
 z_3 \times (p - p_3) & z_4 \times (p - p_4) & z_5 \times (p - p_5)
\end{bmatrix}
\]

Computation of the position vectors of the various links gives

\[
p_0 = p_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad p_3 = p_4 = p_5 = \begin{bmatrix} c_1s_2d_3 - s_1d_2 \\ s_1s_2d_3 + c_1d_2 \\ c_2d_3 \end{bmatrix}
\]

\[
p = \begin{bmatrix} c_1s_2d_3 - s_1d_2 + d_6(c_1c_2c_4s_5 + c_1c_5s_2 - s_1s_4s_5) \\ s_1s_2d_3 + c_1d_2 + d_6(c_1s_4s_5 + c_2c_4s_1s_5 + c_5s_1s_2) \\ c_2d_3 + d_6(c_2c_5 - c_4s_2s_5) \end{bmatrix}
\]

while computation of the unit vectors of joint axes gives

\[
z_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad z_1 = \begin{bmatrix} -s_1 \\ c_1 \\ 0 \end{bmatrix}, \quad z_2 = z_3 = \begin{bmatrix} c_1s_2 \\ s_1s_2 \\ c_2 \end{bmatrix}
\]

\[
z_4 = \begin{bmatrix} -c_1c_2s_4 - s_1c_4 \\ -s_1c_2s_4 + c_1c_4 \\ s_2s_4 \end{bmatrix}, \quad z_5 = \begin{bmatrix} c_1c_2c_4s_5 - s_1s_4s_5 + c_1s_2c_5 \\ s_1c_2c_4s_5 + c_1s_4s_5 + s_1s_2c_5 \\ -s_2c_4s_5 + c_2c_5 \end{bmatrix}
\]

The sought Jacobian can be obtained by developing the computations as in (3.25), leading to expressing end-effector linear and angular velocity as a function of joint velocities.

### 3.3 ANTONYICAL JACOBIAN

The above sections have shown the way to compute the end-effector velocity in terms of the velocity of the end-effector frame. The Jacobian is computed by following a
**geometric technique** in which the contributions of each joint velocity to the components of end-effector linear and angular velocity are determined.

If the end-effector position and orientation are specified in terms of a minimal number of parameters in the operational space as in (2.49), it is natural to ask whether it is possible to compute the Jacobian via differentiation of the direct kinematics function with respect to the joint variables. To this purpose, below an **analytical technique** is presented to compute the Jacobian, and the existing relationship between the two Jacobians is found.

The translational velocity of the end-effector frame can be expressed as the time derivative of vector \( \mathbf{p} \), representing the origin of the end-effector frame with respect to the base frame, i.e.,

\[
\dot{\mathbf{p}} = \frac{\partial \mathbf{p}}{\partial \mathbf{q}} \dot{\mathbf{q}} = \mathbf{J}_P(q) \dot{\mathbf{q}}.
\]  

(3.35)

For what concerns the rotational velocity of the end-effector frame, the minimal representation of orientation in terms of three variables \( \phi \) can be considered. Its time derivative \( \dot{\phi} \) in general differs from the angular velocity vector defined above. In any case, once the function \( \phi(q) \) is known, it is formally correct to consider the Jacobian obtained as

\[
\dot{\phi} = \frac{\partial \phi}{\partial \mathbf{q}} \dot{\mathbf{q}} = \mathbf{J}_\phi(q) \dot{\mathbf{q}}.
\]  

(3.36)

Computing the Jacobian \( \mathbf{J}_\phi(q) \) as \( \partial \phi/\partial \mathbf{q} \) is not straightforward, since the function \( \phi(q) \) is not usually available in direct form, but requires computation of the elements of the relative rotation matrix.

Upon these premises, the differential kinematics equation can be obtained as the time derivative of the direct kinematics equation (2.51), i.e.,

\[
\dot{x} = \begin{bmatrix} \dot{p} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_P(q) \\ \mathbf{J}_\phi(q) \end{bmatrix} \dot{\mathbf{q}} = \mathbf{J}_A(q) \dot{\mathbf{q}}
\]  

(3.37)

where the **analytical Jacobian**

\[
\mathbf{J}_A(q) = \frac{\partial k(q)}{\partial \mathbf{q}}
\]  

(3.38)

is different from the geometric Jacobian \( \mathbf{J} \), since the end-effector angular velocity \( \omega \) with respect to the base frame is not given by \( \dot{\phi} \).

It is possible to find the relationship between the angular velocity \( \omega \) and the rotational velocity \( \dot{\phi} \) for a given set of orientation angles. For instance, consider the Euler angles \( \text{ZYZ} \) defined in Section 2.5; in Fig. 3.3, the vectors corresponding to the rotational velocities \( \dot{\phi}, \dot{\theta}, \) and \( \psi \) have been represented with reference to the current frame. Fig. 3.4 illustrates how to compute the contributions of each rotational velocity to the components of angular velocity about the axes of the reference frame:

- as a result of \( \dot{\phi} \):
  \[
  \begin{bmatrix} \omega_z \omega_y \omega_x \end{bmatrix}^T = \dot{\phi} \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T
  \]

- as a result of \( \dot{\theta} \):
  \[
  \begin{bmatrix} \omega_x \omega_y \omega_z \end{bmatrix}^T = \dot{\theta} \begin{bmatrix} -s_\psi & c_\psi & 0 \end{bmatrix}^T
  \]
FIGURE 3.3
Rotational velocities of Euler angles ZYZ in current frame.

FIGURE 3.4
Composition of elementary rotational velocities for computing angular velocity.

- as a result of \( \dot{\psi} \):
  
  \[
  [\omega_x \ \omega_y \ \omega_z]^T = \dot{\psi} \begin{bmatrix}
  c_\phi s_\theta & s_\phi s_\theta & c_\theta \n  0 & c_\phi & s_\phi \n  1 & 0 & c_\theta
  \end{bmatrix}^T,
  \]

  and then the equation relating the angular velocity \( \omega \) to the time derivative of the Euler angles \( \dot{\phi} \) is

  \[
  \omega = \begin{bmatrix}
  0 & -s_\phi & c_\phi s_\theta \\
  0 & c_\phi & s_\phi s_\theta \\
  1 & 0 & c_\theta
  \end{bmatrix} \dot{\phi} = T(\phi) \dot{\phi}.
  \quad (3.39)
  \]

The determinant of matrix \( T \) is \(-s_\phi\), which implies that the relationship cannot be inverted for \( \dot{\psi} = 0, \pi \). This means that, even though all rotational velocities of the end-effector frame can be expressed by means of a suitable angular velocity vector \( \omega \), there exist angular velocities which cannot be expressed by means of \( \dot{\phi} \) when the orientation
of the end-effector frame causes $s_\theta = 0^2$. In fact, in this situation, the angular velocities that can be described by $\dot{\phi}$ shall have linearly dependent components in the directions orthogonal to axis $z$ ($\omega_x^0 + \omega_y^0 = \dot{\phi})$. An orientation for which the determinant of the transformation matrix vanishes is termed representation singularity of $\phi$.

From a physical viewpoint, the meaning of $\omega$ is more intuitive than that of $\dot{\phi}$. The three components of $\omega$ represent the components of angular velocity with respect to the base frame. Instead, the three elements of $\dot{\phi}$ represent nonorthogonal components of angular velocity defined with respect to the axes of a frame that varies as the end-effector orientation varies. On the other hand, while the integral of $\phi$ over time gives $\phi$, the integral of $\omega$ does not admit a clear physical interpretation, as can be seen in the following example.

**Example 3.2.** Consider an object whose orientation with respect to the base frame is known at time $t = 0$. Assign the following time profiles to $\omega$:

$$
\omega = \begin{bmatrix} \pi/2 & 0 & 0 \end{bmatrix}^T \quad 0 \leq t \leq 1 \quad \omega = \begin{bmatrix} 0 & \pi/2 & 0 \end{bmatrix}^T \quad 1 < t \leq 2.
$$

$$
\omega = \begin{bmatrix} 0 & \pi/2 & 0 \end{bmatrix}^T \quad 0 \leq t \leq 1 \quad \omega = \begin{bmatrix} \pi/2 & 0 & 0 \end{bmatrix}^T \quad 1 < t \leq 2.
$$

---

2 In Section 2.5, it was shown that for this orientation the inverse solution of the Euler angles degenerates.
The integral of \( \omega \) gives the same result in the two cases

\[
\int_0^2 \omega dt = [\pi/2 \quad \pi/2 \quad 0]^T
\]

but the final object orientation corresponding to the second time law is clearly different from the one obtained with the first time law (Fig. 3.5).

Once the transformation \( T \) between \( \omega \) and \( \dot{\phi} \) is given, the analytical Jacobian can be related to the geometric Jacobian as

\[
v = \begin{bmatrix} I & O \\ O & T(\phi) \end{bmatrix} \dot{x} = T_A(\phi) \dot{x}
\] (3.40)

which, in view of (3.3) and (3.37), yields

\[
J = T_A(\phi) J_A.
\] (3.41)

This relationship shows that \( J \) and \( J_A \), in general, differ. Regarding the use of either one or the other in all those problems where the influence of the Jacobian matters, it is anticipated that the geometric Jacobian will be adopted whenever it is necessary to refer to quantities of clear physical meaning, while the analytical Jacobian will be adopted whenever it is necessary to refer to differential quantities of variables defined in the operational space.

For certain manipulator geometries, it is possible to establish a substantial equivalence between \( J \) and \( J_A \). In fact, when the degrees of mobility cause rotations of the end effector all about the same fixed axis in space, the two Jacobians are essentially the same. This is the case of the above three-link planar arm. Its geometric Jacobian (3.31) reveals that only rotations about axis \( z_0 \) are permitted. The \((3 \times 3)\) analytical Jacobian that can be derived by considering the end-effector position components in the plane of the structure and defining the end-effector orientation as \( \phi = \vartheta_1 + \vartheta_2 + \vartheta_3 \) coincides with the matrix that is obtained by eliminating the three null rows of the geometric Jacobian.

### 3.4 KINEMATIC SINGULARITIES

The Jacobian in the differential kinematics equation of a manipulator defines a linear mapping

\[
v = J(q) \dot{q}
\] (3.42)

between the vector \( \dot{q} \) of joint velocities and the vector \( v = [\dot{p}^T \quad \omega^T]^T \) of end-effector velocity. The Jacobian is, in general, a function of the configuration \( q \); those configurations at which \( J \) is rank-deficient are termed kinematic singularities. To find the singularities of a manipulator is of great interest for the following reasons:

- Singularities represent configurations at which mobility of the structure is reduced, i.e., it is not possible to impose an arbitrary motion to the end effector.
null space increases, since the following relation holds

$$\dim(\mathcal{R}(J)) + \dim(\mathcal{N}(J)) = n$$

independently of the rank of the matrix $J$.

The existence of a subspace $\mathcal{N}(J) \neq \emptyset$ for a redundant manipulator allows determination of systematic techniques for handling redundant degrees of freedom. To this purpose, if $\dot{q}^*$ denotes a solution to (3.48) and $P$ is an $(n \times n)$ matrix so that

$$\mathcal{R}(P) \equiv \mathcal{N}(J),$$

also the joint velocity vector

$$\dot{q} = \dot{q}^* + P\dot{q}_a,$$

(3.49)

with arbitrary $\dot{q}_a$, is a solution to (3.48). In fact, premultiplying both sides of (3.49) by $J$ yields

$$J\dot{q} = J\dot{q}^* + JP\dot{q}_a = J\dot{q}^* = v$$

since $JP\dot{q}_a = 0$ for any $\dot{q}_a$. This result is of fundamental importance for redundancy resolution; a solution of the kind (3.49) points out the possibility of choosing the vector of arbitrary joint velocities $\dot{q}_a$ so as to advantageously exploit the redundant degrees of freedom. In fact, the effect of $\dot{q}_a$ is to generate internal motions of the structure that do not change the end-effector position and orientation but may allow, for instance, manipulator reconfiguration into more dexterous postures for execution of a given task.

### 3.6 DIFFERENTIAL KINEMATICS INVERSION

In Section 2.11 it was shown how the inverse kinematics problem admits closed-form solutions only for manipulators having a simple kinematic structure. Problems arise whenever the end effector attains a particular position and/or orientation in the operational space, or the structure is complex and it is not possible to relate end-effector position and orientation to different sets of joint variables, or else the manipulator is redundant. These limitations are caused by the highly nonlinear relationship between joint space variables and operational space variables.

On the other hand, the differential kinematics equation, either in the form (3.37) or in the form (3.48), represents a linear mapping between the joint velocity space and the operational velocity space, although it varies with the current configuration. This fact suggests the possibility to utilize the differential kinematics equation to tackle the inverse kinematics problem.

Suppose that a motion trajectory is assigned to the end effector in terms of $v$ and the initial conditions on position and orientation. The aim is to determine a feasible joint trajectory $(q(t), \dot{q}(t))$ that reproduces the given trajectory.

By considering Eq. (3.48) with $n = r$, the joint velocities can be obtained via simple inversion of the Jacobian matrix

$$\dot{q} = J^{-1}(q)v.$$  

(3.50)
If the initial manipulator posture \( q(0) \) is known, joint positions can be computed by integrating velocities over time, i.e.,

\[
q(t) = \int_0^t \dot{q}(\zeta) d\zeta + q(0).
\]

The integration can be performed in discrete time by resorting to numerical techniques. The simplest technique is based on the Euler integration method; given an integration interval \( \Delta t \), if the joint positions and velocities at time \( t_k \) are known, the joint positions at time \( t_{k+1} = t_k + \Delta t \) can be computed as

\[
q(t_{k+1}) = q(t_k) + \dot{q}(t_k) \Delta t.
\]  \hspace{1cm} (3.51)

This technique for inverting kinematics is independent of the solvability of the kinematic structure. Nonetheless, it is necessary that the Jacobian be square and of full rank; this demands further insight into the cases of redundant manipulators and kinematic singularity occurrence.

### 3.6.1 Redundant Manipulators

When the manipulator is redundant \((r < n)\), the Jacobian matrix has more columns than rows and infinite solutions exist to (3.48). A viable solution method is to formulate the problem as a constrained linear optimization problem.

In detail, once the end-effector velocity \( v \) and Jacobian \( J \) are given (for a given configuration \( q \)), it is desired to find the solutions \( \dot{q} \) that satisfy the linear equation (3.48) and minimize the quadratic cost functional of joint velocities

\[
g(\dot{q}) = \frac{1}{2} \dot{q}^T W \dot{q}
\]

where \( W \) is a suitable \((n \times n)\) symmetric positive definite weighting matrix.

This problem can be solved with the method of Lagrangian multipliers. Consider the modified cost functional

\[
g(\dot{q}, \lambda) = \frac{1}{2} \dot{q}^T W \dot{q} + \lambda^T (v - J \dot{q})
\]

where \( \lambda \) is an \((r \times 1)\) vector of unknown multipliers that allows incorporating the constraint (3.48) in the functional to minimize. The requested solution has to satisfy the necessary conditions:

\[
\left( \frac{\partial g}{\partial \dot{q}} \right)^T = 0 \quad \left( \frac{\partial g}{\partial \lambda} \right)^T = 0.
\]

From the first one, it is \( W \dot{q} - J^T \lambda = 0 \) and thus

\[
\dot{q} = W^{-1} J^T \lambda
\]  \hspace{1cm} (3.52)
where the inverse of $W$ exists. Notice that the solution (3.52) is a minimum, since $\partial^2 g/\partial \dot{q}^2 = W$ is positive definite. From the second condition above, the constraint

$$v = J\dot{q}$$

is recovered. Combining the two conditions gives

$$v = JW^{-1}J^T \lambda;$$

on the assumption that $J$ has full rank, $JW^{-1}J^T$ is an $(r \times r)$ square matrix of rank $r$ and thus can be inverted. Solving for $\lambda$ yields

$$\lambda = (JW^{-1}J^T)^{-1}v$$

which, substituted into (3.52), gives the sought optimal solution

$$\ddot{q} = W^{-1}J^T(JW^{-1}J^T)^{-1}v.$$ (3.53)

Premultiplying both sides of (3.53) by $J$, it is easy to verify that this solution satisfies the differential kinematics equation (3.48).

A particular case occurs when the weighting matrix $W$ is the identity matrix $I$ and the solution simplifies into

$$\ddot{q} = J\dot{v};$$ (3.54)

the matrix

$$J\dot{v} = J^T(JJ^T)^{-1}$$ (3.55)

is the right pseudo-inverse of $J$. The obtained solution locally minimizes the norm of joint velocities.

It was pointed out above that if $\ddot{q}_a$ is a solution to (3.48), also $\ddot{q}_a + P\ddot{q}_a$ is a solution, where $\ddot{q}_a$ is a vector of arbitrary joint velocities and $P$ is a projector in the null space of $J$. Therefore, thanks to the presence of redundant degrees of freedom, the solution (3.54) can be modified by the introduction of another term of the kind $P\ddot{q}_a$. In particular, $\ddot{q}_a$ can be specified so as to satisfy an additional constraint to the problem.

In that case, it is necessary to consider a new cost functional in the form

$$g'(\ddot{q}) = \frac{1}{2}(\ddot{q} - \ddot{q}_a)^T(\ddot{q} - \ddot{q}_a);$$

this choice is aimed at minimizing the norm of vector $\ddot{q} - \ddot{q}_a$; in other words, solutions are sought which satisfy the constraint (3.48) and are as close as possible to $\ddot{q}_a$. In this way, the objective specified through $\ddot{q}_a$ becomes unavoidably a secondary objective to satisfy with respect to the primary objective specified by the constraint (3.48).

Proceeding in a way similar to the above yields

$$g'(\ddot{q}, \lambda) = \frac{1}{2}(\ddot{q} - \ddot{q}_a)^T(\ddot{q} - \ddot{q}_a) + \lambda^T(v - J\dot{q});$$
from the first necessary condition it is
\[ \dot{q} = J^T \lambda + \dot{q}_a \]  
(3.56)
which, substituted into (3.48), gives
\[ \lambda = (JJ^T)^{-1}(v - J\dot{q}_a). \]
Finally, substituting \( \lambda \) back in (3.56) gives
\[ \dot{q} = J^\dagger v + (I - J^\dagger J)\dot{q}_a. \]  
(3.57)
As can be easily recognized, the obtained solution is composed of two terms. The first one is relative to minimum norm joint velocities. The second one, termed homogeneous solution, attempts to satisfy the additional constraint to specify via \( \dot{q}_a \); the matrix \((I - J^\dagger J)\) is one of those matrices \( P \) introduced in (3.49) which allows projecting the vector \( \dot{q}_a \) in the null space of \( J \), so as not to violate the constraint (3.48). A direct consequence is that, in the case \( v = 0 \), is is possible to generate internal motions described by \((I - J^\dagger J)\dot{q}_a\) that reconfigure the manipulator structure without changing the end-effector position and orientation.

Finally, it is worth discussing the way to specify the vector \( \dot{q}_a \) for a convenient utilization of redundant degrees of freedom. A typical choice is
\[ \dot{q}_a = k_a \left( \frac{\partial w(q)}{\partial q} \right)^T \]  
(3.58)
where \( k_a > 0 \) and \( w(q) \) is a (secondary) objective function of the joint variables. Since the solution moves along the direction of the gradient of the objective function, it attempts to locally maximize it compatible to the primary objective (kinematic constraint). Typical objective functions are:

- The manipulability measure, defined as
\[ w(q) = \sqrt{\det(J(q)J^T(q))} \]  
(3.59)
which vanishes at a singular configuration; thus, by maximizing this measure, redundancy is exploited to move away from singularities.

- The distance from mechanical joint limits, defined as
\[ w(q) = -\frac{1}{2n} \sum_{i=1}^{n} \left( \frac{q_i - \dot{q}_i}{q_{i,M} - q_{i,m}} \right)^2 \]  
(3.60)

\footnote{It should be recalled that the additional constraint has secondary priority with respect to the primary kinematic constraint.}
where $q_{lM}$ ($q_{im}$) denotes the maximum (minimum) joint limit and $\bar{q}$, the middle value of the joint range; thus, by maximizing this distance, redundancy is exploited to keep the joint variables as close as possible to the center of their ranges.

- The **distance from an obstacle**, defined as

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

(3.61)

where $\mathbf{o}$ is the position vector of a suitable point on the obstacle (its center, for instance, if the obstacle is modeled as a sphere) and $\mathbf{p}$ is the position vector of a generic point along the structure; thus, by maximizing this distance, redundancy is exploited to avoid collision of the manipulator with an obstacle\(^5\).

### 3.6.2 Kinematic Singularities

Both solutions (3.50) and (3.54) can be computed only when the Jacobian has full rank. Hence, they become meaningless when the manipulator is at a singular configuration; in such a case, the system $\mathbf{v} = J\dot{q}$ contains linearly dependent equations.

It is possible to find a solution $\dot{q}$ by extracting all the linearly independent equations only if $\mathbf{v} \in \mathcal{R}(J)$. The occurrence of this situation means that the assigned path is physically executable by the manipulator, even though it is at a singular configuration. If instead $\mathbf{v} \notin \mathcal{R}(J)$, the system of equations has no solution; this means that the operational space path cannot be executed by the manipulator at the given posture.

It is important to underline that the inversion of the Jacobian can represent a serious inconvenience not only at a singularity but also in the neighborhood of a singularity. For instance, for the Jacobian inverse it is well known that its computation requires the computation of the determinant; in the neighborhood of a singularity, the determinant takes on a relatively small value which can cause large joint velocities (see point (c) in Section 3.4). Consider again the above example of the shoulder singularity for the anthropomorphic arm. If a path is assigned to the end effector which passes nearby the base rotation axis (geometric locus of singular configurations), the base joint is forced to make a rotation of about $\pi$ in a relatively short time to allow the end effector to keep tracking the imposed trajectory.

A more rigorous analysis of the solution features in the neighborhood of singular configurations can be developed by resorting to the singular value decomposition (SVD) of matrix $J$.

An alternative solution overcoming the problem of inverting differential kinematics in the neighborhood of a singularity is provided by the so-called damped least-

---

\(^5\) If an obstacle occurs along the end-effector path, it is opportune to invert the order of priority between the kinematic constraint and the additional constraint; in this way the obstacle may be avoided, but one gives up tracking the desired path.
squares (DLS) inverse

\[ \mathbf{J}^* = \mathbf{J}^T (\mathbf{J} \mathbf{J}^T + k^2 \mathbf{I})^{-1} \quad (3.62) \]

where \( k \) is a damping factor that renders the inversion better conditioned from a numerical viewpoint. It can be shown that such a solution can be obtained by reformulating the problem in terms of the minimization of the cost functional

\[ g''(\dot{q}) = \frac{1}{2} (v - \mathbf{J} \dot{q})^T (v - \mathbf{J} \dot{q}) + \frac{1}{2} k^2 \dot{q}^T \dot{q}, \]

where the introduction of the first term allows tolerating a finite inversion error with the advantage of norm-bounded velocities. The factor \( k \) establishes the relative weight between the two objectives, and there exist techniques for selecting optimal values for the damping factor.

### 3.7 INVERSE KINEMATICS ALGORITHMS

In the previous section it was shown how to invert kinematics by using the differential kinematics equation. In the numerical implementation of (3.51), computation of joint velocities is obtained by using the inverse of the Jacobian evaluated with the joint variables at the previous instant of time

\[ \mathbf{q}(t_{k+1}) = \mathbf{q}(t_k) + \mathbf{J}^{-1} (\mathbf{q}(t_k)) \mathbf{v}(t_k) \Delta t. \]

It follows that the computed joint velocities \( \dot{q} \) do not coincide with those satisfying (3.50) in the continuous time. Therefore, reconstruction of joint variables \( q \) is entrusted to a numerical integration which involves drift phenomena of the solution; as a consequence, the end-effector location corresponding to the computed joint variables differs from the desired one.

This inconvenience can be overcome by resorting to a solution scheme that accounts for the operational space error between the desired and the actual end-effector position and orientation. Let

\[ e = \mathbf{x}_d - \mathbf{x} \quad (3.63) \]

be the expression of such error.

Consider the time derivative of (3.63)

\[ \dot{e} = \dot{x}_d - \dot{x} \quad (3.64) \]

which, according to differential kinematics (3.37), can be written as

\[ \dot{e} = \dot{x}_d - \mathbf{J}_A(q) \dot{q}. \quad (3.65) \]

For this equation to lead to an inverse kinematics algorithm, it is worth relating the computed joint velocity vector \( \dot{q} \) to the error \( e \) so that (3.65) gives a differential equation describing error evolution over time. Nonetheless, it is necessary to choose a relationship between \( \dot{q} \) and \( e \) that ensures convergence of the error to zero.