

JACOBIAN-BASED ALGORITHMS: A BRIDGE BETWEEN KINEMATICS AND CONTROL

The PRISMA Lab

www.prisma.unina.it

Bruno Siciliano

Dipartimento di Ingegneria dell'Informazione e Ingegneria Elettrica

Università di Salerno

Via Ponte Don Melillo, 84084 Fisciano (SA), Italy

bsiciliano@unisa.it

Lorenzo Sciavicco

Dipartimento di Informatica e Automazione

Università di Roma Tre

Via della Vasca Navale 79, 00146 Roma, Italy

sciavicco@dia.uniroma3.it

Stefano Chiaverini

Dipartimento di Automazione, Elettromagnetismo,

Ingegneria dell'Informazione e Matematica Industriale

Università di Cassino

Via G. Di Biasio 43, 03043 Cassino (FR), Italy

chiaverini@unicas.it

Pasquale Chiacchio, Luigi Villani

Dipartimento di Informatica e Sistemistica

Università di Napoli Federico II

Via Claudio 21, 80125 Napoli, Italy

pasquale.chiacchio | luigi.villani@unina.it

Fabrizio Caccavale

Dipartimento di Ingegneria e Fisica dell'Ambiente

Università della Basilicata

Contrada Macchia Romana, 85100 Potenza, Italy

caccavale@unibas.it

Abstract Whenever it comes to the point of controlling complex robotic systems, all goes back to fundamental kinematic issues which should be thoroughly mastered to design effective control systems. This has been the experience of many scholars working in the field whose research has been influenced, directly or indirectly, by the seminal work of bright designers and kinematicians like Professor Bernard Roth. The present paper aims at surveying a body of literature on Jacobian-based algorithms developed by the PRISMA Lab, e.g. a group of researchers with control background who have constantly found inspiration in the fascinating world of kinematics.

Keywords: Robot manipulators, kinematics, Jacobian, kinematic control, serial manipulators, redundant manipulators, parallel manipulators, cooperative multiarm systems, underwater vehicle-manipulator systems, spacecraft-manipulator systems, flexible manipulators

1. Kinematic Control

No doubt, the *Jacobian* is the most important tool for characterizing articulated mechanical systems, such as robot manipulators. It is useful for finding singular configuration, analyzing redundancy, developing inverse kinematics algorithms, describing the statics, deriving dynamic equations of motion and, last but not least, designing control schemes.

Robot *kinematic control* consists of solving the motion control problem into two stages, i.e. the desired end-effector trajectory is transformed via inverse kinematics into the corresponding joint trajectories, which then constitute the reference inputs to some joint space control scheme [Siciliano, 1990a]. This approach differs from operational space control [Khatib, 1987] in the sense that manipulator kinematics is handled outside the control loop thus allowing the problem of kinematic singularities and/or redundancy to be solved separately from the motion control problem. The key point of kinematic control is the solution to the inverse kinematics problem.

In all general terms, the kinematics of a robotic systems is expressed in terms of a nonlinear relationship between the vector $\mathbf{q}$ of configuration variables and the vector $\mathbf{x}$ of task variables, i.e.

$$\mathbf{x} = \mathbf{k}(\mathbf{q}). \quad (1)$$

Well-assessed methods exist to calculate the function $\mathbf{k}$ in a closed form [Bottema and Roth, 1979]. On the other hand, finding solutions to (1) has attracted a wide number of kinematicians who have spent considerable efforts on investigating the existence of admissible solutions, e.g. [Pieper and Roth, 1969] and, in the affirmative case, computing all possible solutions [Raghavan and Roth, 1995].

An effective numerical approach to solve (1) can be pursued by resorting to the differential mapping

$$\dot{\mathbf{x}} = \mathbf{J}_A(\mathbf{q})\dot{\mathbf{q}}, \quad (2)$$

where $\mathbf{J}_A(\mathbf{q}) = \partial \mathbf{k} / \partial \mathbf{q}$ is the robot *analytical Jacobian*. The term “analytical” merely reflects the analytical derivation of the Jacobian, as opposed to the “geometric” Jacobian which would relate configuration velocities to task velocities [Sciavicco and Siciliano, 2000]; this issue is dealt with in further detail in the next section.

The main advantage of (2) over (1) is its linearity in the configuration velocities. This allows solving the differential kinematics by a suitable inversion of the Jacobian matrix to compute [Whitney, 1969]

$$\dot{\mathbf{q}} = \mathbf{J}_A^{-1}(\mathbf{q})\dot{\mathbf{x}}, \quad (3)$$

which then can be integrated over time to give $\mathbf{q}$. Of course, if $\mathbf{J}_A$ is square matrix, a simple inverse suffices; yet, if $\mathbf{J}_A$ is not full-rank, numerical robustness could be achieved by resorting to a damped least-squares inverse [Chiaverini *et al*, 1994], where the damping factor can be suitably tuned as a function of proximity to singularities, e.g. on the basis of the estimates of the smallest singular values of the Jacobian matrix [Chiaverini, 1993].

On the other hand, if the Jacobian is a non-square matrix, then a suitable right pseudo-inverse can be adopted, i.e.

$$\dot{\mathbf{q}} = \mathbf{J}_A^\dagger(\mathbf{q})\dot{\mathbf{x}}. \quad (4)$$

From an algorithmic viewpoint, it is advisable to introduce a feedback correction term aimed at eliminating any numerical drift of the solution, i.e.

$$\dot{\mathbf{q}} = \mathbf{J}_A^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}), \quad (5)$$

where $\mathbf{e} = \mathbf{x}_d - \mathbf{x}$ is the error between the desired and the actual set of task variables, and $\mathbf{K}$ is a suitable positive definite matrix gain. The resulting scheme of the *Jacobian inverse* kinematics algorithm is illustrated in Fig. 1. This scheme solves the so-called *kinematic control* problem, i.e. to find suitable joint trajectories $\mathbf{q}(t)$ corresponding to a desired task trajectory $\mathbf{x}_d(t)$.

An effective alternative to the above algorithm based on (5) can be adopted by resorting to a *Jacobian transpose* in lieu of its inverse, i.e. [Sciavicco and Siciliano, 1986]

$$\dot{\mathbf{q}} = \mathbf{J}_A^T(\mathbf{q})\mathbf{K}\mathbf{e}, \quad (6)$$

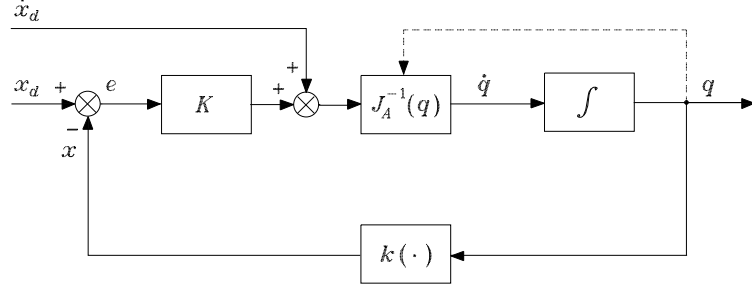


Figure 1. The Jacobian Inverse Kinematics Algorithm

which can be proved to guarantee limited tracking errors and null steady-state errors [Chiacchio and Siciliano, 1989]. The resulting scheme is illustrated in Fig. 2.

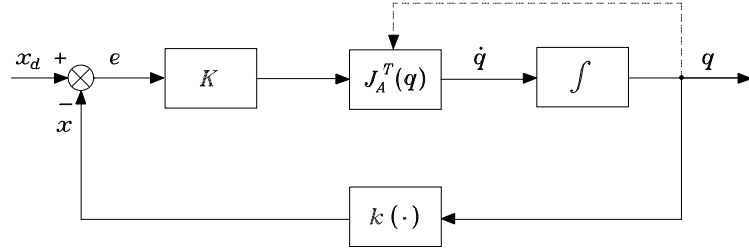


Figure 2. The Jacobian Transpose Kinematics Algorithm

The Jacobian-based algorithms hereby formulated can be used to solve a number of kinematic control problems, as illustrated in the following sections; namely, for serial manipulators, redundant manipulators, parallel manipulators, cooperative multiarm systems, underwater vehicle-manipulator systems, spacecraft-manipulator systems, and flexible manipulators.

2. Serial Manipulators

The kinematics of a *serial* open-chain manipulator can be described in the form (1) where $\mathbf{q}$ is the set of joint variables, and $\mathbf{x}_e$ is the set of end-effector variables

$$\mathbf{x}_e = \begin{bmatrix} \mathbf{p}_e \\ \boldsymbol{\varphi}_e \end{bmatrix}, \quad (7)$$

where $\mathbf{p}_e$ denotes the end-effector position and $\boldsymbol{\varphi}_e$ denotes a minimal representation of orientation in terms of three Euler angles, with respect to some base frame (Fig. 3).

Notice that the vector $\mathbf{x}_e$ is defined in the task space, i.e. in which the robot task is specified. The dimension of this space is at most $m = 6$, since 3 coordinates specify position and 3 coordinates specify orientation. Nevertheless, depending on the geometry of the task, a reduced number of task space variables may be specified; for instance, for a planar robot it is $m = 3$, since two coordinates specify position and one coordinate specifies orientation.

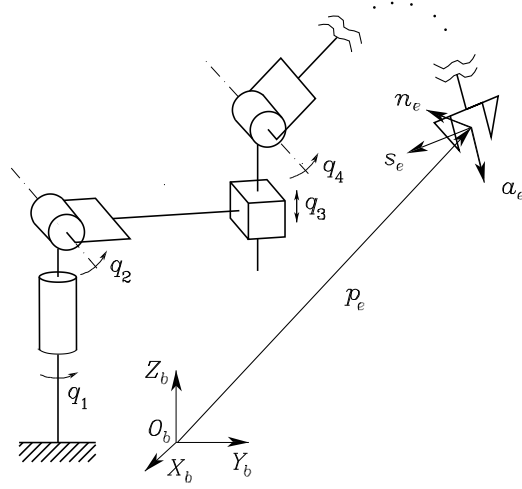


Figure 3. A Serial Manipulator

This representation of position and orientation allows the description of the end-effector task in terms of a number of inherently independent parameters. Nevertheless, the computation of Euler angles from the joint variables goes through the computation of the end-effector frame rotation matrix

$$\mathbf{R}_e = [\mathbf{n}_e \quad \mathbf{s}_e \quad \mathbf{a}_e] \quad (8)$$

and, as such, it suffers from the so-called representation singularities. These clearly appear in the mapping between the time derivatives of the Euler angles and the end-effector angular velocity, i.e.

$$\boldsymbol{\omega}_e = \mathbf{T}(\boldsymbol{\varphi}_e) \dot{\boldsymbol{\varphi}}_e, \quad (9)$$

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

In particular, the angular velocity $\boldsymbol{\omega}_e$ is related to the time derivative of the rotation matrix as

$$\dot{\mathbf{R}}_e = \mathbf{S}(\boldsymbol{\omega}_e) \mathbf{R}_e, \quad (10)$$

where $\mathbf{S}(\cdot)$ is the matrix operator performing the cross product between two (3×1) vectors.

Therefore, the differential kinematics equation for a serial manipulator is established in the form

$$\mathbf{v} = \begin{bmatrix} \dot{\mathbf{p}}_e \\ \boldsymbol{\omega}_e \end{bmatrix} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \quad (11)$$

where $\mathbf{J}(\mathbf{q})$ is the *geometric Jacobian*. The computation of this matrix usually follows a geometric procedure that is based on computing the contributions of each joint velocity to the linear and angular end-effector velocities [Whitney, 1972].

It can be easily recognized that the geometric Jacobian in general differs from the analytical Jacobian. Concerning their use, the geometric Jacobian is adopted when physical quantities are of interest while the analytical Jacobian is adopted when task space quantities are of interest. It is always possible to pass from one Jacobian to the other, except at a representation singularity.

The algorithm based on (4) for a serial manipulator constitutes an inverse kinematics algorithm which makes use of the analytical Jacobian. More insight about the implications of different end-effector orientation descriptions can be gained by separating the position from the orientation components. Using the geometric Jacobian in lieu of the analytical Jacobian, the solution can be rewritten as

$$\dot{\mathbf{q}} = \mathbf{J}^\dagger(\mathbf{q})\mathbf{v} \quad (12)$$

with

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_p \\ \mathbf{v}_o \end{bmatrix}, \quad (13)$$

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

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

$$\mathbf{v}_p = \dot{\mathbf{p}}_d + \mathbf{K}_p \mathbf{e}_p, \quad (14)$$

where the position error

$$\mathbf{e}_p = \mathbf{p}_d - \mathbf{p}_e(\mathbf{q}) \quad (15)$$

between the desired and actual end-effector positions has been defined.

On the other hand, for what concerns the orientation error, some considerations are in order depending on the type of description adopted. If *Euler angles* are adopted, the resolved angular velocity in (13) is chosen as

$$\mathbf{v}_o = \mathbf{T}(\boldsymbol{\varphi}_e)(\dot{\boldsymbol{\varphi}}_d + \mathbf{K}_o \mathbf{e}_{o,\text{Eul}}), \quad (16)$$

where

$$\mathbf{e}_{o,\text{Eul}} = \boldsymbol{\varphi}_d - \boldsymbol{\varphi}_e(\mathbf{q}) \quad (17)$$

is the orientation error.

In order to overcome the drawback of representation singularities in (16), an algorithm based on an *alternative Euler angles* description can be conceived [Caccavale *et al*, 1998] which makes use of the rotation matrix describing the mutual orientation between the desired and the actual end-effector frame, i.e.

$${}^e\mathbf{R}_d = \mathbf{R}_e^T(\mathbf{q})\mathbf{R}_d. \quad (18)$$

Differentiating (18) with respect to time and accounting for (10) gives

$${}^e\dot{\mathbf{R}}_d = \mathbf{S}({}^e\boldsymbol{\omega}_{de}){}^e\mathbf{R}_d, \quad (19)$$

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

Let $\boldsymbol{\varphi}_{de}$ denote the set of Euler angles that can be extracted from ${}^e\mathbf{R}_d$. Then, in view of (9) and (10), the angular velocity ${}^e\boldsymbol{\omega}_{de}$ in (19) is related to the time derivative of $\boldsymbol{\varphi}_{de}$ as

$${}^e\boldsymbol{\omega}_{de} = \mathbf{T}(\boldsymbol{\varphi}_{de})\dot{\boldsymbol{\varphi}}_{de}. \quad (20)$$

At this point, the resolved angular velocity in (13) can be chosen as

$$\mathbf{v}_o = \boldsymbol{\omega}_d + \mathbf{R}_e\mathbf{T}(\boldsymbol{\varphi}_{de})\mathbf{K}_o\mathbf{e}_{o,\text{EulAlt}} \quad (21)$$

with

$$\mathbf{e}_{o,\text{EulAlt}} = \boldsymbol{\varphi}_{de}. \quad (22)$$

The clear advantage of the alternative over the classical Euler angles algorithm based on (16) is that, by adopting a representation $\boldsymbol{\varphi}_{de}$ for which $\mathbf{T}(\mathbf{0})$ is nonsingular, representation singularities occur only for large orientation errors. In other words, the ill-conditioning of matrix $\mathbf{T}$ is not influenced by the desired or actual end-effector orientation but only by the orientation error. In this respect, the choice of a particular Euler angles description among the twelve possible should be carefully made, i.e. in the sense of avoiding a representation singularity.

In order to overcome the problem of representation singularities, an inverse kinematics algorithm based on the *angle/axis* description of orientation can be devised. From (18), the rotation ϑ_{de} and the unit vector $\mathbf{r}_{de}$ can be extracted using well-known formulæ. Then, the orientation error can be defined as

$$\mathbf{e}_{o,\text{AnAx}} = \sin \vartheta_{de} \mathbf{r}_{de}. \quad (23)$$

Notice that (23) gives a unique solution for $-\pi/2 < \vartheta_{de} < \pi/2$, but this interval is not limiting for a convergent inverse kinematics algorithm. It can be shown that a computational expression of the orientation error in (23) is given by [Sciavicco and Siciliano, 2000]

$$\mathbf{e}_{o, \text{AnAx}} = \frac{1}{2} \left(\mathbf{S}(\mathbf{n}_e(\mathbf{q}))\mathbf{n}_d + \mathbf{S}(\mathbf{s}_e(\mathbf{q}))\mathbf{s}_d + \mathbf{S}(\mathbf{a}_e(\mathbf{q}))\mathbf{a}_d \right), \quad (24)$$

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

Differentiation of (24) with respect to time gives

$$\dot{\mathbf{e}}_{o, \text{AnAx}} = \mathbf{L}^T \boldsymbol{\omega}_d - \mathbf{L} \boldsymbol{\omega}_e \quad (25)$$

with

$$\mathbf{L} = -\frac{1}{2} \left(\mathbf{S}(\mathbf{n}_d)\mathbf{S}(\mathbf{n}_e) + \mathbf{S}(\mathbf{s}_d)\mathbf{S}(\mathbf{s}_e) + \mathbf{S}(\mathbf{a}_d)\mathbf{S}(\mathbf{a}_e) \right). \quad (26)$$

At this point, the resolved angular velocity in (13) can be chosen as

$$\mathbf{v}_o = \mathbf{L}^{-1}(\mathbf{L}^T \boldsymbol{\omega}_d + \mathbf{K}_o \mathbf{e}_{o, \text{AnAx}}). \quad (27)$$

Another inverse kinematics algorithm is based on the *unit quaternion* description of orientation, so as to overcome the above limitation on the angle/axis description. Let $\mathcal{Q}_d = \{\eta_d, \boldsymbol{\epsilon}_d\}$ and $\mathcal{Q}_e = \{\eta_e, \boldsymbol{\epsilon}_e\}$ represent the unit quaternions associated with $\mathbf{R}_d$ and $\mathbf{R}_e$, respectively. The mutual orientation can be expressed in terms of the unit quaternion $\mathcal{Q}_{de} = \{\eta_{de}, \boldsymbol{\epsilon}_{de}\}$ where

$$\eta_{de} = \eta_e(\mathbf{q})\eta_d + \boldsymbol{\epsilon}_e^T(\mathbf{q})\boldsymbol{\epsilon}_d \quad (28)$$

$$\boldsymbol{\epsilon}_{de} = \eta_e(\mathbf{q})\boldsymbol{\epsilon}_d - \eta_d\boldsymbol{\epsilon}_e(\mathbf{q}) - \mathbf{S}(\boldsymbol{\epsilon}_d)\boldsymbol{\epsilon}_e(\mathbf{q}).$$

It can be recognized that $\mathcal{Q}_{de} = \{1, \mathbf{0}\}$ if and only if $\mathbf{R}_e$ and $\mathbf{R}_d$ are aligned, and thus it is sufficient to consider $\boldsymbol{\epsilon}_{de}$ to express an end-effector orientation error, i.e. [Chiaverini and Siciliano, 1999a]

$$\mathbf{e}_{o, \text{Quat}} = \boldsymbol{\epsilon}_{de}. \quad (29)$$

Notice that the explicit computation of $\eta_e(\mathbf{q})$ and $\boldsymbol{\epsilon}_e(\mathbf{q})$ is not possible, but it requires the intermediate computation of the rotation matrix $\mathbf{R}_e(\mathbf{q})$ that is available from the manipulator direct kinematics; then, the unit quaternion can be extracted using well-known formulæ, e.g.

[Shepperd, 1978]. At this point, the resolved angular velocity in (13) can be chosen as

$$\mathbf{v}_o = \boldsymbol{\omega}_d + \mathbf{K}_o \mathbf{e}_{o, \text{Quat}}, \quad (30)$$

which can be shown to ensure null tracking errors.

As a final remark, the above kinematic control schemes can be easily extended to the second order, i.e. solving the joint motion at the acceleration level instead of velocity level. This may be advantageous for model-based dynamic control in the joint space [Caccavale *et al*, 1997].

3. Redundant Manipulators

Whenever a manipulator possesses more joint variables than the number of task variables, it is said to be kinematically *redundant*.

If it is desired to exploit redundant degrees of freedom, solution (5) can be generalized to

$$\dot{\mathbf{q}} = \mathbf{J}_A^\dagger(\mathbf{q}) (\dot{\mathbf{x}}_d + \mathbf{K} \mathbf{e}) + (\mathbf{I} - \mathbf{J}_A^\dagger(\mathbf{q}) \mathbf{J}_A(\mathbf{q})) \dot{\mathbf{q}}_0, \quad (31)$$

where the matrix $(\mathbf{I} - \mathbf{J}^\dagger \mathbf{J})$ is a projector of the joint vector $\dot{\mathbf{q}}_0$ onto the null space of $\mathbf{J}$ [Chiacchio *et al*, 1991].

This result is of fundamental importance for redundancy resolution, since solution (31) evidences the possibility of choosing the vector $\dot{\mathbf{q}}_0$ so as to exploit the redundant degrees of freedom. In fact, the contribution of $\dot{\mathbf{q}}_0$ is to generate null space motions of the structure that do not contribute to the end-effector motion but allow the manipulator to reach configurations which are more dexterous for the execution of the given task [Siciliano, 1990b]. In case of numerical problems in the neighborhood of singularities, the pseudoinverse can be replaced with a suitable damped least-squares inverse [Chiaverini, 1997].

An effective way to choosing $\dot{\mathbf{q}}_0$ can be derived by combining the Jacobian pseudoinverse solution with the Jacobian transpose solution as illustrated below. This is carried out in the framework of the so-called *augmented task space* approach to exploit redundancy in robotic systems [Sciavicco and Siciliano, 1988]. The idea is to introduce an additional constraint task by specifying a vector $\mathbf{x}_C$ as a function of the joint variables, i.e.

$$\mathbf{x}_C = \mathbf{k}_C(\mathbf{q}), \quad (32)$$

so as to constrain at most all the available redundant degrees of freedom. The constraint task vector $\mathbf{x}_C$ can be chosen by embedding scalar objective functions, e.g. to improve dexterity, avoid obstacles, etc.

Differentiating (32) with respect to time gives

$$\dot{\mathbf{x}}_C = \mathbf{J}_C(\mathbf{q}) \dot{\mathbf{q}}, \quad (33)$$

where $\mathbf{J}_C(\mathbf{q}) = \partial \mathbf{k}_C / \partial \mathbf{q}$ is the constraint Jacobian. The result is an augmented differential kinematics equation given by (2) and (33), based on a Jacobian matrix

$$\mathbf{J}' = \begin{bmatrix} \mathbf{J}_A \\ \mathbf{J}_C \end{bmatrix}. \quad (34)$$

When a constraint task is specified independently of the end-effector task, there is no guarantee that the matrix $\mathbf{J}'$ remains full-rank along the entire task path; singularities of $\mathbf{J}'$ are termed *artificial singularities* and it can be shown that those are given by singularities of the matrix $\mathbf{J}_C(\mathbf{I} - \mathbf{J}_A^\dagger \mathbf{J}_A)$.

The above discussion suggests that, when solving for joint velocities, a *task priority strategy* is advisable so as to avoid conflicting situations between the end-effector task and the constraint task [Nakamura *et al*, 1987]. Substituting (31) into (33) gives

$$\dot{\mathbf{x}}_C = \mathbf{J}_C(\mathbf{q})\mathbf{J}_A^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}) + \mathbf{J}_C(\mathbf{q})\left(\mathbf{I} - \mathbf{J}_A^\dagger(\mathbf{q})\mathbf{J}_A(\mathbf{q})\right)\dot{\mathbf{q}}_0 \quad (35)$$

which could be solved for $\dot{\mathbf{q}}_0$ provided that artificial singularities are avoided. Observing that equality (35) can be achieved only for the components of $\dot{\mathbf{x}}_C$ belonging to the range space of $\mathbf{J}_C$, it is sufficient to consider the equation

$$\mathbf{J}_C^\dagger(\mathbf{q})\dot{\mathbf{x}}_C = \mathbf{J}_A^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}) + \left(\mathbf{I} - \mathbf{J}_A^\dagger(\mathbf{q})\mathbf{J}_A(\mathbf{q})\right)\dot{\mathbf{q}}_0 \quad (36)$$

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

$$\dot{\mathbf{q}}_0 = \left(\mathbf{I} - \mathbf{J}_A^\dagger(\mathbf{q})\mathbf{J}_A(\mathbf{q})\right)^\dagger \left(\mathbf{J}_C^\dagger(\mathbf{q})\dot{\mathbf{x}}_C - \mathbf{J}_A^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e})\right). \quad (37)$$

By recalling that $(\mathbf{I} - \mathbf{J}_A^\dagger \mathbf{J}_A)^\dagger = (\mathbf{I} - \mathbf{J}_A^\dagger \mathbf{J}_A)$, solution (37) reduces to the simple form [Chiaverini, 1997]

$$\dot{\mathbf{q}}_0 = \left(\mathbf{I} - \mathbf{J}_A^\dagger(\mathbf{q})\mathbf{J}_A(\mathbf{q})\right)\mathbf{J}_C^\dagger(\mathbf{q})\dot{\mathbf{x}}_C. \quad (38)$$

Folding (38) back into (31) and exploiting the idempotence of $(\mathbf{I} - \mathbf{J}_A^\dagger \mathbf{J}_A)$ gives

$$\dot{\mathbf{q}} = \mathbf{J}_A^\dagger(\mathbf{q})(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}) + \left(\mathbf{I} - \mathbf{J}_A^\dagger(\mathbf{q})\mathbf{J}_A(\mathbf{q})\right)\mathbf{J}_C^\dagger(\mathbf{q})(\dot{\mathbf{x}}_{Cd} + \mathbf{K}_C\mathbf{e}_C) \quad (39)$$

with $\mathbf{e}_C = \mathbf{x}_{Cd} - \mathbf{x}_C$, being $\mathbf{x}_{Cd}$ the desired value of the constraint task, and $\mathbf{K}_C$ is a positive definite matrix. The operator $(\mathbf{I} - \mathbf{J}_A^\dagger \mathbf{J}_A)$ projects the secondary velocity contribution $\dot{\mathbf{q}}_0$ on the null space of $\mathbf{J}_A$, guaranteeing correct execution of the primary end-effector task while

the secondary constraint task is correctly executed as long as it does not interfere with the end-effector task. Obviously, if desired, the order of priority can be switched, e.g. in an obstacle avoidance task when an obstacle comes to be along the end-effector path.

In the case when $\mathbf{J}_C$ becomes singular, a damped least-squares inverse of $\mathbf{J}_C$ in lieu of the pseudoinverse in (38) can be used. Otherwise, by recalling the Jacobian transpose solution for the end-effector task (6), the null space joint velocity vector can be conveniently chosen as [Chiacchio *et al*, 1991]

$$\dot{\mathbf{q}}_0 = \mathbf{J}_C^T(\mathbf{q})\mathbf{K}_C(\mathbf{x}_{Cd} - \mathbf{x}_C). \quad (40)$$

which allows the algorithm to work at a singularity of $\mathbf{J}_C$ and even at an artificial singularity. A tracking error arises for the constraint task but, observing that the desired constraint task is often constant over time ($\dot{\mathbf{x}}_{Cd} = \mathbf{0}$), it can be concluded that the solution based on (40) performs equally well at steady state.

4. Parallel Manipulators

In the latest years, a novel type of closed-chain manipulators have been receiving quite a deal of attention; namely, the *parallel* manipulators [Merlet, 2000] which are constituted by a fixed base and a mobile base, connected by a number of independent kinematic chains. This allows obtaining high structural stiffness and performing high-speed motions, e.g. the Delta robot and the Hexa robot. One drawback with respect to open-chain manipulators, though, is a typically reduced workspace.

A particular family of parallel manipulators is characterized by having a radial link connected to the end effector of variable length. To this family belongs the industrial robot Tricept in Fig. 4. It is a six-degree-of-freedom (dof) robot manipulator comprising a three-dof parallel structure and a spherical wrist.

Let $\mathbf{p}$ denote the position of the origin of the frame attached to the mobile base. The inverse differential kinematics can be written as

$$\dot{\mathbf{q}} = \mathbf{J}^{-1}(\mathbf{p})\mathbf{v}, \quad (41)$$

where $\mathbf{q}$ is the vector of joint variables and $\mathbf{v}$ is the linear velocity vector. The geometric Jacobian inverse $\mathbf{J}^{-1}$ can be calculated from the inverse kinematics function, which is available via simple geometry.

On the other hand, the direct kinematics problem for a parallel robot consists of finding the vector of position coordinates $\mathbf{p}$ as a function of the vector of joint variables $\mathbf{q}$. Typically, such a problem does not admit closed-form solutions and numerical algorithms should be used.

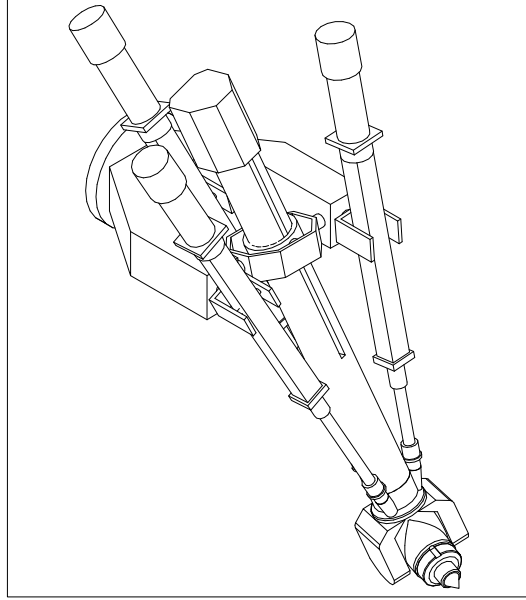


Figure 4. The Tricept Parallel Manipulator

Therefore, an effective solution to such kind of kinematic control problem for a parallel manipulator can be devised by transposing the above Jacobian-based algorithm, i.e. [Siciliano, 1999a]

$$\mathbf{v} = \mathbf{J}(\mathbf{p})(\dot{\mathbf{q}}_d + \mathbf{K}\mathbf{e}), \quad (42)$$

where $\mathbf{e} = \mathbf{q}_d - \mathbf{q}$ denotes the error between $\mathbf{q}_d$ and the computed joint variables $\mathbf{q}$. The resulting scheme is illustrated in Fig. 5.

An alternative and computationally more efficient solution can be devised which avoids the inversion of the inverse geometric Jacobian to compute $\mathbf{J}$ in (42). In fact, the choice

$$\mathbf{v} = \mathbf{J}^{-T}(\mathbf{p})\mathbf{K}\mathbf{e} \quad (43)$$

is based on the transpose of the inverse geometric Jacobian, and can be proved to guarantee limited tracking errors and null steady-state errors. The resulting scheme is illustrated in Fig. 6.

5. Cooperative Multiarm Systems

Various robotic applications require the adoption of *multiarm* systems in lieu of single robot manipulators. These include, for instance, manipulation of heavy or non-rigid objects and mating of mechanical pieces.

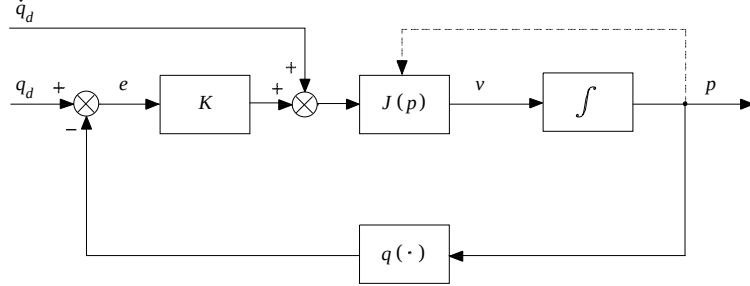


Figure 5. The Jacobian Kinematics Algorithm for Parallel Manipulators

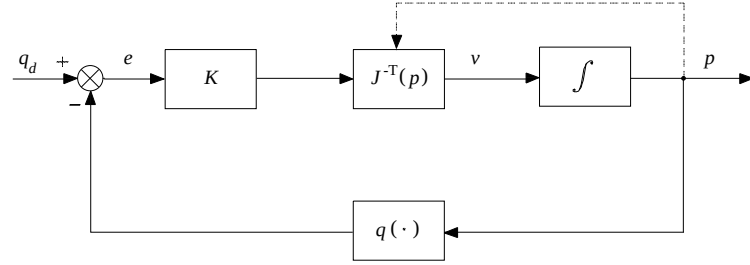


Figure 6. The Inverse-Jacobian Transpose Kinematics Algorithm for Parallel Manipulators

In all such cases the multiple robots should operate in a *cooperative* fashion so as to synchronize the relative motions, avoid undesired collisions, maintain the grasp between the arms and the object, etc.

Without loss of generality, consider a system of two cooperative manipulators. For each manipulator ($i = 1, 2$), let $\mathbf{p}_i$ and $\mathbf{R}_i$ respectively denote the end-effector position vector and rotation matrix which are referred to a common base frame.

The absolute position of the system can be defined as the origin of the absolute frame (a) which can be expressed as a function of the positions of the two end effectors. One simple choice is

$$\mathbf{p}_a = \frac{1}{2}(\mathbf{p}_1 + \mathbf{p}_2). \quad (44)$$

Then, the rotation matrix giving the absolute orientation can be defined as

$$\mathbf{R}_a = \mathbf{R}_1 \mathbf{R}_{1k_{12}}(\vartheta_{12}/2), \quad (45)$$

where ${}^1\mathbf{k}_{12}$ and ϑ_{12} are respectively the unit vector and the angle that realize the rotation described by ${}^1\mathbf{R}_2$, i.e. the orientation of frame 2 with respect to frame 1. Therefore, the above choice corresponds to make a

rotation about axis ${}^1\mathbf{k}_{12}$ by an angle which is half the angle needed to align $\mathbf{R}_2$ with $\mathbf{R}_1$.

The absolute position and orientation describe the task in terms of the composition of the position and orientation of the single manipulators. It is clear that there exist infinite end-effector configurations giving the same absolute position and orientation. Therefore, in order to fully describe a coordinated motion, the position and orientation of one manipulator relative to the other is also of concern.

The relative position between the two end effectors can be defined as

$$\mathbf{p}_r = \mathbf{p}_2 - \mathbf{p}_1. \quad (46)$$

The relative orientation between the two end effectors can be defined with reference to the end-effector frame of either manipulator —say the first one— in terms of the rotation matrix

$${}^1\mathbf{R}_r = {}^1\mathbf{R}_2. \quad (47)$$

The position and orientation of the various frames is illustrated in Figure 7.

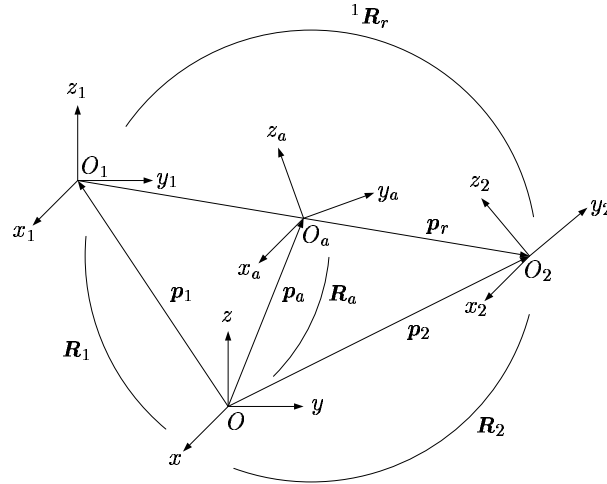


Figure 7. Absolute and Relative Frames in a Dual-Arm System

Notice that, in order to be independent of the absolute motion of the system, it is more convenient to specify the relative position with reference to the absolute frame, i.e. ${}^a\mathbf{p}_r$. The relationship between ${}^a\mathbf{p}_r$ and $\mathbf{p}_r$ is given by

$$\mathbf{p}_r = \mathbf{R}_a {}^a\mathbf{p}_r. \quad (48)$$

A distinguished feature of the proposed formulation is that coordinated motion of the system is achieved without necessarily assuming that the two manipulators are kinematically constrained through the presence of an object between the two end effectors. Nevertheless, if the two end effectors hold a common object, general manipulation tasks can be described by the above formulation. For instance, if the task is to move a tightly grasped object without deforming it, a trajectory has to be assigned to $\mathbf{p}_a$ and $\mathbf{R}_a$ while ${}^a\mathbf{p}_r$ and ${}^1\mathbf{R}_r$ have to be kept constant. Yet, if the task is to stretch, bend or shear the object, suitable trajectories have to be specified for the relative variables too. Cases of non-tight grasp can be handled as well [Chiacchio *et al*, 1996].

Having established a task formulation for the direct kinematics of the two-manipulator system, it is useful to derive also the differential kinematics relating the coordinated (absolute and relative) velocities to the corresponding velocities of the two manipulators. This is of interest not only for characterizing the velocity mappings, similar to task formulations using the grasp matrix, but also for solving the inverse kinematics of the two-manipulator system as well as for handling the presence of redundant degrees of freedom in the system.

The absolute linear velocity of the system is obtained as the time derivative of (44), i.e.

$$\dot{\mathbf{p}}_a = \frac{1}{2}(\dot{\mathbf{p}}_1 + \dot{\mathbf{p}}_2). \quad (49)$$

Differentiating (45) with respect to time and using (10) yields

$$\boldsymbol{\omega}_a = \frac{1}{2}(\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2). \quad (50)$$

The relative linear velocity of the system is obtained as the time derivative of (46), which in view of (48) gives

$$\dot{\mathbf{p}}_r = \mathbf{R}_a^a \dot{\mathbf{p}}_r + \mathbf{S}(\boldsymbol{\omega}_a) \mathbf{p}_r \quad (51)$$

with $\boldsymbol{\omega}_a$ as in (50). Finally, differentiating (47) with respect to time and using (10) yields

$$\boldsymbol{\omega}_r = \boldsymbol{\omega}_2 - \boldsymbol{\omega}_1 \quad (52)$$

which has been expressed in the base frame.

The differential kinematics of each manipulator is described by

$$\begin{bmatrix} \dot{\mathbf{p}}_i \\ \boldsymbol{\omega}_i \end{bmatrix} = \mathbf{J}_i(\mathbf{q}_i) \dot{\mathbf{q}}_i \quad i = 1, 2, \quad (53)$$

with obvious meaning of the quantities.

At this point, combining (49),(50) and taking into account (53) yields

$$\begin{bmatrix} \dot{\mathbf{p}}_a \\ \boldsymbol{\omega}_a \end{bmatrix} = \mathbf{J}_a(\mathbf{q}_1, \mathbf{q}_2) \begin{bmatrix} \dot{\mathbf{q}}_1 \\ \dot{\mathbf{q}}_2 \end{bmatrix}, \quad (54)$$

where the *absolute Jacobian* is defined as

$$\mathbf{J}_a = \begin{bmatrix} \frac{1}{2}\mathbf{J}_1 & \frac{1}{2}\mathbf{J}_2 \end{bmatrix}. \quad (55)$$

Further, combining (51),(52) and taking into account (53) yields

$$\begin{bmatrix} \dot{\mathbf{p}}_r \\ \boldsymbol{\omega}_r \end{bmatrix} = \mathbf{J}_r(\mathbf{q}_1, \mathbf{q}_2) \begin{bmatrix} \dot{\mathbf{q}}_1 \\ \dot{\mathbf{q}}_2 \end{bmatrix}, \quad (56)$$

where the *relative Jacobian* is defined as

$$\mathbf{J}_r = \begin{bmatrix} -\mathbf{J}_1 & \mathbf{J}_2 \end{bmatrix}. \quad (57)$$

The algorithm based on (12) can be keenly applied to solve the inverse kinematics for the two-manipulator system at issue. To this purpose, it is sufficient to define the joint variable vector as

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_1 \\ \mathbf{q}_2 \end{bmatrix}, \quad (58)$$

and the Jacobian as

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_a \\ \mathbf{J}_r \end{bmatrix}, \quad (59)$$

where $\mathbf{J}_a$, $\mathbf{J}_r$ are given as in (55) and (57), respectively.

The resolved velocity $\mathbf{v}$ in (12) can be chosen as

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_{ad} + \mathbf{K}_a \mathbf{e}_a \\ \mathbf{v}_{rd} + \mathbf{K}_r \mathbf{e}_r \end{bmatrix}. \quad (60)$$

The absolute velocity term is given by

$$\mathbf{v}_{ad} = \begin{bmatrix} \dot{\mathbf{p}}_{ad} \\ \boldsymbol{\omega}_{ad} \end{bmatrix}, \quad (61)$$

where $\dot{\mathbf{p}}_{ad}$ and $\boldsymbol{\omega}_{ad}$ are respectively the desired absolute linear and angular velocities specified by the user in the base frame. The relative velocity term is given by

$$\mathbf{v}_{rd} = \begin{bmatrix} \mathbf{R}_a^a \dot{\mathbf{p}}_{rd} + \mathbf{S}(\boldsymbol{\omega}_a) \mathbf{R}_a^a \mathbf{p}_{rd} \\ \boldsymbol{\omega}_{rd} \end{bmatrix}, \quad (62)$$

where ${}^a\dot{\mathbf{p}}_{rd}$ is the desired relative linear velocity specified by the user in the object frame and ${}^1\dot{\boldsymbol{\omega}}_{rd}$ is the desired relative angular velocity specified by the user in the end-effector frame of the first manipulator. Notice that the expression of the translational part of the relative velocity presents an additional term which is a consequence of having assigned the relative position with reference to the absolute frame.

Finally, the absolute error term in (60) has a position and an orientation component and is given by

$$\mathbf{e}_a = \begin{bmatrix} \mathbf{p}_{ad} - \mathbf{p}_a \\ \frac{1}{2} \left(\mathbf{S}(\mathbf{n}_a)\mathbf{n}_{ad} + \mathbf{S}(\mathbf{s}_a)\mathbf{s}_{ad} + \mathbf{S}(\mathbf{a}_a)\mathbf{a}_{ad} \right) \end{bmatrix}, \quad (63)$$

where $\mathbf{p}_{ad}$ is the desired absolute position specified by the user in the base frame, $\mathbf{p}_a$ is the actual absolute position that can be computed as in (44), $\mathbf{n}_{ad}, \mathbf{s}_{ad}, \mathbf{a}_{ad}$ are the column vectors of the rotation matrix $\mathbf{R}_{ad}$ giving the desired absolute orientation specified by the user in the base frame, and $\mathbf{n}_a, \mathbf{s}_a, \mathbf{a}_a$ are the column vectors of the rotation matrix $\mathbf{R}_a$ in (45).

The relative error is given by

$$\mathbf{e}_r = \begin{bmatrix} \mathbf{R}_a {}^a\mathbf{p}_{rd} - \mathbf{p}_r \\ \frac{1}{2} \mathbf{R}_1 \left(\mathbf{S}({}^1\mathbf{n}_r){}^1\mathbf{n}_{rd} + \mathbf{S}({}^1\mathbf{s}_r){}^1\mathbf{s}_{rd} + \mathbf{S}({}^1\mathbf{a}_r){}^1\mathbf{a}_{rd} \right) \end{bmatrix}. \quad (64)$$

The rotation $\mathbf{R}_a$ is aimed at expressing the desired relative position ${}^a\mathbf{p}_{rd}$, assigned by the user in the absolute frame, in the base frame; in this way, the specification of the desired relative position between the two end effectors is not affected by the absolute frame orientation. Further in (64) notice that: $\mathbf{p}_r$ can be computed as in (46); ${}^1\mathbf{n}_{rd}, {}^1\mathbf{s}_{rd}, {}^1\mathbf{a}_{rd}$ are the column vectors of the rotation matrix ${}^1\mathbf{R}_{rd}$ giving the desired relative orientation specified by the user in the end-effector frame of the first manipulator; ${}^1\mathbf{n}_r, {}^1\mathbf{s}_r, {}^1\mathbf{a}_r$ are the column vectors of the rotation matrix ${}^1\mathbf{R}_r$ in (47); and the rotation $\mathbf{R}_1$ is aimed at expressing the orientation error in the base frame.

6. Underwater Vehicle–Manipulator Systems

Execution of *underwater* manipulation tasks requires the use of a robot *manipulator* mounted onboard a *vehicle* actuated by thrusters; for that, an Underwater Vehicle–Manipulator System (UVMS) is always kinematically redundant due to the dof's provided by the vehicle itself besides those provided by the robot arm. This naturally poses a redundancy resolution problem for motion coordination between the vehicle and the manipulator, whose solution can significantly increase efficiency of the system.

As a matter of fact, while it is obvious that gross motion of the UVMS must be provided by vehicle movements and fine motion of the end effector is best accomplished by the sole manipulator motion, reconfiguration of the whole system is required when the manipulator would be asked to work at the boundaries of its workspace or close to a kinematic singularity. The reconfiguration of the UVMS must suitably trade off the need of saving energy and keeping the control bandwidth high (that would call for motion of the manipulator) against the need of ensuring high manipulability for the robot arm (that would require continuous adjustment of the vehicle position and orientation).

The above redundancy resolution problem can be dealt with in the framework of task-priority Jacobian-based inverse kinematics algorithms as follows [Antonelli and Chiaverini, 2003a].

The vehicle is completely described by its position and orientation with respect to a base frame (frame b) that is assumed to be earth-fixed and inertial. Define the vector $\mathbf{x}_v = [\mathbf{p}_v^T \ \boldsymbol{\varphi}_v^T]^T$, where $\mathbf{p}_v$ is the vector of vehicle position coordinates and $\boldsymbol{\varphi}_v$ is the vector of vehicle Roll-Pitch-Yaw Euler-angle coordinates, both quantities defined with reference to the base frame.

It is useful to define the vehicle's velocity in a frame attached to the vehicle (frame v); let ${}^v\mathbf{x}_v = [{}^v\dot{\mathbf{p}}_v^T \ {}^v\boldsymbol{\omega}_v^T]^T$, where ${}^v\dot{\mathbf{p}}_v$ is the linear velocity of the vehicle with respect to the base frame and ${}^v\boldsymbol{\omega}_v$ is the angular velocity of the vehicle, both quantities defined with reference to frame v .

The above velocity vectors satisfy the following equations:

$${}^v\dot{\mathbf{p}}_v = {}^v\mathbf{R} \dot{\mathbf{p}}_v \quad (65)$$

$${}^v\boldsymbol{\omega} = \mathbf{T}(\boldsymbol{\phi}_v)\dot{\boldsymbol{\phi}}_v, \quad (66)$$

where ${}^v\mathbf{R}$ is the rotation matrix from frame b to frame v , and $\mathbf{T}$ is a suitable transformation matrix as in (9).

At this point, define $\mathbf{q} = [q_1 \cdots q_n]^T$ as the vector of joint positions, n being the number of joints. By introducing the vector $\boldsymbol{\zeta} = [{}^v\dot{\mathbf{p}}_v^T \ {}^v\boldsymbol{\omega}_v^T \ \dot{\mathbf{q}}^T]^T$ it is possible to rewrite the relation between the above velocities in compact form, i.e.

$$\boldsymbol{\zeta} = \begin{bmatrix} {}^v\mathbf{R} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{T} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{x}}_v \\ \dot{\mathbf{q}} \end{bmatrix} = \mathbf{J}_S \begin{bmatrix} \dot{\mathbf{x}}_v \\ \dot{\mathbf{q}} \end{bmatrix}, \quad (67)$$

where $\mathbf{J}_S$ is the system Jacobian, while $\mathbf{I}$ and $\mathbf{O}$ respectively denote identity and null matrices of appropriate dimensions.

Since the task to be executed requires position/orientation control of the manipulator's end effector, it is necessary to consider the end-effector

posture $\mathbf{x}_e = [\mathbf{p}_e^T \ \boldsymbol{\varphi}_e^T]^T$, where $\mathbf{p}_e$ is the end-effector position and $\boldsymbol{\varphi}_e$ is the end-effector orientation (Euler angles), both quantities defined with reference to the base frame. Notice that the vector $\mathbf{p}_e$ is a function of the system configuration, i.e. $\mathbf{p}_e(\mathbf{x}_v, \mathbf{q})$, while the vector $\boldsymbol{\varphi}_e$ does only depend on the vehicle orientation, i.e. $\boldsymbol{\varphi}_e(\boldsymbol{\varphi}_v, \mathbf{q})$. The relationship between the end-effector posture and the system configuration can be expressed by the following nonlinear kinematic equation

$$\mathbf{x}_e = \mathbf{k}(\mathbf{x}_v, \mathbf{q}). \quad (68)$$

The time derivatives of the end-effector position and orientation are related to the actual end-effector velocities via relations analogous to (65) and (66); namely,

$${}^e\dot{\mathbf{p}}_e = {}^e\mathbf{R} \dot{\mathbf{p}}_e \quad (69)$$

$${}^e\boldsymbol{\omega}_e = \mathbf{T}(\boldsymbol{\varphi}_e)\dot{\boldsymbol{\varphi}}_e, \quad (70)$$

where ${}^e\mathbf{R}$ is the rotation matrix from the base frame to the end-effector frame (see Figure 8) and $\mathbf{T}$ is the matrix associated to the use of the Euler angles of the end-effector frame.

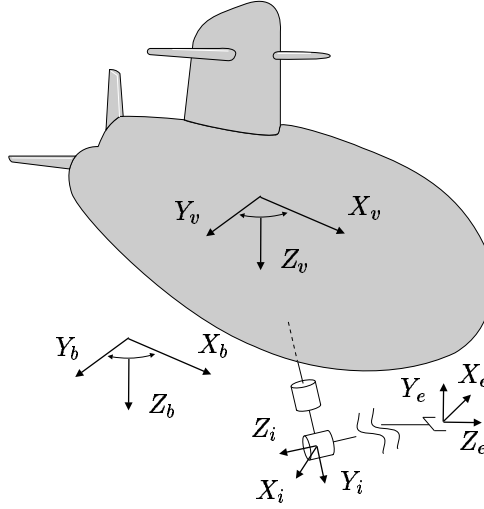


Figure 8. Sketch of a UVMS with Relevant Frames

The end-effector velocities in the base frame are related to the system velocities by a suitable Jacobian matrix, i.e.

$$\mathbf{v}_e = \begin{bmatrix} \dot{\mathbf{p}}_e \\ \mathbf{R}_e^e \boldsymbol{\omega}_e \end{bmatrix} = \mathbf{J}(\mathbf{x}_v, \mathbf{q})\boldsymbol{\zeta}. \quad (71)$$

Notice that the Jacobian has been derived with respect to the angular velocity of the end effector expressed in the base frame. This is done in order to allow implementation of an inverse kinematics algorithm in terms of quaternions as described in the following.

The objective of kinematic control for a UVMS is to find suitable vehicle/joint trajectories $\mathbf{x}_v(t)$, $\mathbf{q}(t)$ that correspond to a desired end-effector trajectory $\mathbf{x}_{ed}(t)$. The outputs of the inverse kinematics algorithm provide the reference values to the control law of the UVMS. This control law will be in charge of computing the driving forces aimed at tracking the reference trajectory for the system while counteracting dynamic effects, external disturbances, and modeling errors.

Equation (71) maps the $(6+n)$ -dimensional vehicle/joint velocities into the m -dimensional end-effector task velocities, where the typical case $(6+n) \geq m$ will be considered. Notice that, in case of UVMS, the Jacobian has always full rank due to the mobility of the vehicle, i.e. a rigid body with 6 dof's. However, movement of the vehicle has to be avoided when unnecessary.

Considering as primary task the end-effector position/orientation, the Jacobian-based inverse kinematics algorithm (39) becomes [Antonelli and Chiaverini, 1998]:

$$\begin{aligned} \zeta_r = & \mathbf{J}^\dagger(\mathbf{x}_v, \mathbf{q}) (\mathbf{v}_{ed} + \mathbf{K}\mathbf{e}) \\ & + \left(\mathbf{I} - \mathbf{J}^\dagger(\mathbf{x}_v, \mathbf{q}) \mathbf{J}(\mathbf{x}_v, \mathbf{q}) \right) \mathbf{J}_C^\dagger(\mathbf{x}_v, \mathbf{q}) (\dot{\mathbf{x}}_{Cd} + \mathbf{K}_C \mathbf{e}_C), \end{aligned} \quad (72)$$

where the constraint task is to be defined later, while $\mathbf{e}$ and $\mathbf{e}_C$ are suitable expressions of the errors.

As for the vector components related to the position variables, the error is simply given by the difference between the desired and the actual values. By using the quaternion attitude representation for the error components related to the orientation variables, the vector $\mathbf{e}$ turns out to be [Chiaverini and Siciliano, 1999a]:

$$\mathbf{e} = \begin{bmatrix} \mathbf{p}_{ed} - \mathbf{p}_e \\ \eta \epsilon_d - \eta_d \epsilon - \mathbf{S}(\epsilon_d) \epsilon \end{bmatrix}, \quad (73)$$

where $\mathcal{Q}_d = \{\eta_d, \epsilon_d\}$ and $\mathcal{Q} = \{\eta, \epsilon\}$ are the desired and actual attitudes expressed by quaternions, respectively.

Because of the different inertia characteristics of the vehicle and of the manipulator, it would be preferable to perform fast motions of small amplitude by means of the manipulator while leaving the vehicle with the task of executing slow gross motions. This might be achieved by adopting a weighted pseudoinverse

$$\mathbf{J}_w^\dagger = \mathbf{W}^{-1} \mathbf{J}^T \left(\mathbf{J} \mathbf{W}^{-1} \mathbf{J}^T \right)^{-1}, \quad (74)$$

with the $(6+n) \times (6+n)$ matrix

$$\mathbf{W}^{-1}(\beta) = \begin{bmatrix} (1-\beta)\mathbf{I} & \mathbf{O} \\ \mathbf{O} & \beta\mathbf{I} \end{bmatrix}, \quad (75)$$

where β is a weighting factor belonging to the interval $[0, 1]$ such that $\beta = 0$ corresponds to sole vehicle motion and $\beta = 1$ to sole manipulator motion.

During task execution, setting a constant value of β would mean to fix the motion distribution between the vehicle and the manipulator. Nevertheless, the use of a fixed weighting factor inside the interval $[0, 1]$ has a drawback: it causes motion of the manipulator also if the desired end-effector posture is out of reach; on the other hand, it causes motion of the vehicle also if the manipulator alone could perform the task.

Another problem is the necessity to handle a large number of variables; UVMS's, in fact, are complex systems and several variables must be monitored during the motion, e.g. the manipulator manipulability, the joint range limits to avoid mechanical breaks, the vehicle roll and pitch angles for correct tuning of the proximity sensors, the yaw angle to exploit the vehicle shape in presence of ocean current, etc. As it can be easily understood, it is quite difficult to handle all these terms without a kinematic control approach. Nevertheless, the existing techniques do not allow finding a flexible and reliable solution.

To overcome this drawback a fuzzy theory approach can be considered at two different levels. First, it is desired to manage the distribution of motion between the vehicle and the manipulator; second, it is possible to consider multiple secondary tasks that are activated only when the corresponding variable is outside (inside) a desired range. This can be done using different weighting factors adjusted on-line by a fuzzy inference system [Antonelli and Chiaverini, 2003a; Antonelli and Chiaverini, 2003b]. In detail, the crisp outputs are the scalar β in (75) and a vector of coefficients α_i that are used in the task priority equation as follows

$$\boldsymbol{\zeta} = \mathbf{J}_W^\dagger (\mathbf{v}_{ed} + \mathbf{K}\mathbf{e}) + \left(\mathbf{I} - \mathbf{J}_W^\dagger \mathbf{J}_W \right) \left(\sum_i \alpha_i \mathbf{J}_{Ci}^\dagger \mathbf{w}_{Ci} \right), \quad (76)$$

where $\mathbf{w}_{Ci}$ are suitably defined constraint task variables and $\mathbf{J}_{Ci}$ are the associated Jacobians. Both β and α_i 's are tuned according to the state of the system and to given behavioral rules. The inputs of the fuzzy inference system depend on the variables of interest in the specific mission. As an example, the end-effector error, the ocean current measure, the system's dexterity, as well as the force sensor readings, can be easily taken into account by setting up a suitable set of fuzzy rules.

To avoid the exponential growth of the fuzzy rules to be implemented as the number of tasks is increased, the secondary tasks are suitably organized in a hierarchy. Also, the rules have to guarantee that only one α_i is high at a time so as to avoid conflict between the secondary tasks.

7. Spacecraft–Manipulator Systems

Entrusting extravehicular activities to a mechanical robot manipulator rather than to an astronaut is foreseen to reduce the danger of *space* servicing jobs.

Consider a system composed by an n -dof *manipulator* with rigid links mounted on a rigid body *spacecraft*, free-floating in a zero-gravity environment. In the following, $\mathbf{q}$ will denote the $(n \times 1)$ vector of joint variables. The vector $\mathbf{p}_s$ represents the position of a spacecraft-fixed coordinate frame (s) with respect to an inertial reference frame; $\mathbf{R}_s$ is the rotation matrix expressing the spacecraft attitude, i.e. the orientation of the spacecraft frame with respect to the inertial frame. Let also $\mathbf{p}_e$ and $\mathbf{R}_e$ express, respectively, the position and orientation of the end-effector frame with respect to the same inertial reference frame.

The kinematic equation relating the joint and spacecraft variables to the end-effector position can be written as

$$\mathbf{p}_e = \mathbf{p}_s + \mathbf{R}_s {}^s\mathbf{p}_{es}(\mathbf{q}), \quad (77)$$

where ${}^s\mathbf{p}_{es}$ is the position of the end-effector frame relative to the spacecraft frame; the end-effector orientation can be described by the rotation matrix

$$\mathbf{R}_e = \mathbf{R}_s {}^s\mathbf{R}_e(\mathbf{q}), \quad (78)$$

where ${}^s\mathbf{R}_e = \mathbf{R}_s^T \mathbf{R}_e$ is the rotation matrix expressing the relative orientation between the end-effector frame and the spacecraft frame. Notice that ${}^s\mathbf{p}_{es}(\mathbf{q})$ and ${}^s\mathbf{R}_e(\mathbf{q})$ represent the usual direct kinematics equations of a ground-fixed manipulator with respect to its base frame (see Figure 9).

In view of solving the inverse kinematics for such a system, it is convenient to consider differential kinematics in lieu of (77),(78). Let $\mathbf{v}_e = [\dot{\mathbf{p}}_e^T \boldsymbol{\omega}_e^T]^T$ be the vector of generalized end-effector velocity, where $\dot{\mathbf{p}}_e$ and $\boldsymbol{\omega}_e$ denote the linear and angular velocity, respectively. Let ${}^s\mathbf{J}_{es}$ be the manipulator geometric Jacobian relating the joint velocities $\dot{\mathbf{q}}$ to the end-effector velocity relative to the spacecraft frame ${}^s\mathbf{v}_{es} = [{}^s\dot{\mathbf{p}}_{es}^T {}^s\boldsymbol{\omega}_{es}^T]^T$, where ${}^s\boldsymbol{\omega}_{es} = \mathbf{R}_s^T(\boldsymbol{\omega}_e - \boldsymbol{\omega}_s)$. Let also $\mathbf{v}_s = [\dot{\mathbf{p}}_s^T \boldsymbol{\omega}_s^T]^T$ be the vector of generalized spacecraft velocity. Then, by differentiating (77) and (78),

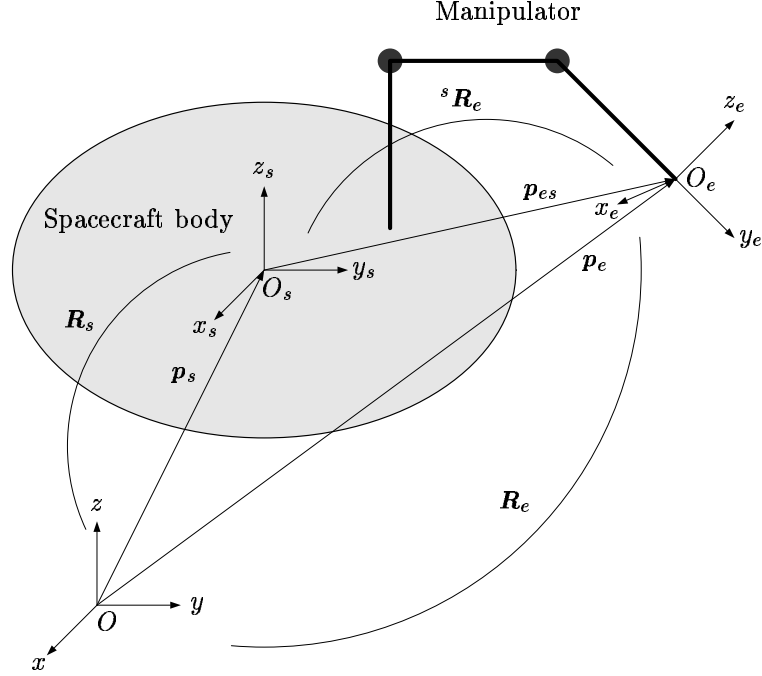


Figure 9. Sketch of a Spacecraft–Manipulator System with Relevant Frames

it follows

$$\mathbf{v}_e = \mathbf{J}_s(\mathbf{q}, \mathbf{R}_s) \mathbf{v}_s + \mathbf{R}_s {}^s\mathbf{J}_{es}(\mathbf{q}) \dot{\mathbf{q}}, \quad (79)$$

with

$$\mathbf{J}_s = \begin{bmatrix} \mathbf{I} & -\mathbf{S}(\mathbf{R}_s {}^s\mathbf{p}_{es}) \\ \mathbf{O} & \mathbf{I} \end{bmatrix}. \quad (80)$$

Since the spacecraft position is of no concern, $\dot{\mathbf{p}}_s$ can be eliminated from (79). This can be achieved by exploiting the geometrical definition of the mass center of the system $\mathbf{p}_G$

$$\mathbf{p}_G \sum_i m_i = \sum_i m_i \mathbf{r}_i, \quad (81)$$

where m_i is the mass of the i th rigid body in the system and $\mathbf{r}_i$ represents the position of its center of mass with respect to the inertial reference frame. Hence, by following the guidelines in [Umetani and Yoshida, 1989] and assuming null initial velocity for the system's center of mass, the following expression can be derived from (79) and (81)

$$\mathbf{v}_e = \bar{\mathbf{J}}_s(\mathbf{q}, \mathbf{R}_s) \boldsymbol{\omega}_s + \bar{\mathbf{J}}_e(\mathbf{q}, \mathbf{R}_s) \dot{\mathbf{q}}, \quad (82)$$

where the matrices $\bar{\mathbf{J}}_s$ and $\bar{\mathbf{J}}_e$ depend on the Jacobian matrices $\mathbf{J}_s$ and ${}^s\mathbf{J}_{es}$ in (79), the masses m_i and the position vectors $\mathbf{r}_i$.

It is assumed that no external forces or torques act on the center of mass of the system. Thus, there are no devices intended to change spacecraft attitude, e.g. reaction wheels or thrusters. In this case, momentum conservation dictates that

$$\sum_i m_i \dot{\mathbf{r}}_i = \mathbf{0} \quad (83)$$

for the translational momentum and

$$\sum_i (\mathbf{M}_i \boldsymbol{\omega}_i + m_i \mathbf{r}_i \times \dot{\mathbf{r}}_i) = \mathbf{0}, \quad (84)$$

where $\boldsymbol{\omega}_i$ is the angular velocity of the i th body in the system and $\mathbf{M}_i$ is the inertia matrix around its center of mass. It has been assumed that the initial momentum of the free-floating system is null, without loss of generality.

The quantities $\mathbf{r}_i$, $\dot{\mathbf{r}}_i$, $\boldsymbol{\omega}_i$ and $\mathbf{M}_i$ are referred to the inertial frame, and it is not difficult to compute their expressions as a function of $\mathbf{R}_s$ and $\mathbf{q}$ [Nakamura and Mukherjee, 1991]. Substituting such expressions into eqs. (83),(84), momentum conservation can be compactly expressed as

$$\mathbf{M}_s \boldsymbol{\omega}_s + \mathbf{M}_e \dot{\mathbf{q}} = \mathbf{0}, \quad (85)$$

where $\mathbf{M}_s$ is a (3×3) matrix related to the spacecraft inertia and $\mathbf{M}_e$ is a $(3 \times n)$ matrix related to the manipulator inertia.

Equations (82) and (85) are fundamental for analyzing the motion of the system composed by the robotic manipulator mounted on the free-floating spacecraft. Since $\mathbf{M}_s \boldsymbol{\omega}_s$ represents the spacecraft rotational momentum, $\mathbf{M}_s$ is a non-singular matrix; then, solving (85) for $\boldsymbol{\omega}_s$ and substituting in (82) allows eliminating the dependence on the spacecraft attitude changes, i.e.

$$\mathbf{v}_e = \mathbf{J}_G \dot{\mathbf{q}} \quad (86)$$

where the matrix

$$\mathbf{J}_G = \bar{\mathbf{J}}_e - \bar{\mathbf{J}}_s \mathbf{M}_s^{-1} \mathbf{M}_e \quad (87)$$

is termed the *generalized Jacobian* for the spacecraft/manipulator system [Umetani and Yoshida, 1989].

The attractive feature of (86) is its formal analogy with the well-known differential kinematics equation for ground-fixed manipulators. The manipulator Jacobian $\bar{\mathbf{J}}_e$ is modified by the presence of a term accounting for the relative inertial weight between the spacecraft and the manipulator. The larger the spacecraft inertia, the smaller the reaction caused

by the manipulator motion; in the limit of a very massive spacecraft, the generalized Jacobian will tend to the manipulator Jacobian.

The algorithm based on (12) can be keenly applied to solve the inverse kinematics for the free-floating manipulator system at issue with $\mathbf{J}_G$ in lieu of $\mathbf{J}$.

With reference to the use of the unit quaternion for the orientation error, the resolved velocity can be chosen as

$$\mathbf{v} = \begin{bmatrix} \dot{\mathbf{p}}_d + \mathbf{K}_p \mathbf{e}_p \\ \boldsymbol{\omega}_d + \mathbf{K}_o \boldsymbol{\epsilon}_{de} \end{bmatrix}. \quad (88)$$

In the framework of inverse kinematics algorithms, it is important to recognize the presence of redundant degrees of freedom in the system with respect to the required task. As pointed out in [Nenchev *et al*, 1992], three cases of redundancy can be distinguished in connection with the number of degrees of mobility (joint variables) n versus the number of dof's characterizing the assigned task (task space variables) $m_s + m_e$, where m_s and m_e refer to the spacecraft and manipulator task, respectively. If $n < m_s + m_e$, the manipulator can be redundant with respect either to the spacecraft task ($n > m_s$) or to the end-effector task ($n > m_e$), but redundancy will not allow specifying a coordinated task for the spacecraft and the end-effector. If $n = m_s + m_e$, the available dof's can be exploited to coordinate the motion of the spacecraft with that of the end-effector. If $n > m_s + m_e$, it is possible to introduce additional constraints to be satisfied along with spacecraft/manipulator motion coordination. In the following it is assumed that $n \geq m_s + m_e$.

Of course, the number of dof's n becomes larger than $m_s + m_e$ either by increasing the number of joints or by relaxing some task variables. Hence, the Jacobian matrix to be considered in (86) may be obtained by eliminating some rows of $\mathbf{J}_G$ corresponding to the relaxed task variables, i.e. its dimensions become $(m_e \times n)$ with $m_e \leq 6$ and $n \geq 6$. This implies that suitable strategies have to be pursued to manage both the presence of a non-square Jacobian matrix and the redundant dof's in the inverse kinematics algorithms previously defined.

One possibility would be to apply the Jacobian transpose algorithm to the robotic system described by (86). With this solution, however, the resulting spacecraft attitude varies as the manipulator end-effector moves along the trajectory. It is then advisable to exploit the redundant dof's $n - m_e \geq m_s$ to impose a desired time evolution of spacecraft attitude $\mathbf{R}_{s_d}$. This is a typical case of augmented task space for redundant manipulators.

Since a conflict may arise between the end-effector and the constraint (spacecraft) task, an order of priority should be assigned. By revisiting

the solution algorithm for redundant manipulators in (39), the joint velocity solution can be computed as [Caccavale and Siciliano, 2001]

$$\dot{\mathbf{q}} = \mathbf{J}_G^\dagger \mathbf{v} + (\mathbf{I} - \mathbf{J}_G^\dagger \mathbf{J}_G) \mathbf{J}_C^\top \mathbf{K}_C \mathbf{e}_C, \quad (89)$$

where $\mathbf{e}_C$ is the error for the constraint task and $\mathbf{J}_C$ is the associated Jacobian.

If $n - m_e = m_s$, then the error relative to the constraint task can be computed in terms of the vector part of the unit quaternion $\{\eta_{s_d s}, \boldsymbol{\epsilon}_{s_d s}\}$, representing the orientation displacement between the desired and the actual spacecraft frame: $\mathbf{e}_C = \boldsymbol{\epsilon}_{s_d s}$; the corresponding Jacobian matrix can be derived from (85)

$$\mathbf{J}_C = -\mathbf{M}_s^{-1} \mathbf{M}_e. \quad (90)$$

In this case, manipulator redundancy is exploited to reach the desired spacecraft attitude $\mathbf{R}_{s_d}$ while tracking the desired end-effector trajectory. On the other hand, if $n - m_e > m_s$, then the constraint task can encompass also an additional constraint, such as mechanical joint range, obstacle avoidance etc.

With solution (89), the constraint task is not guaranteed to be satisfied along the whole motion execution. Therefore, for those applications where the constraint task is judged to be more important than the end-effector task, the order of priority can be switched with obvious transposition of subscripts in (89).

An interesting case is that when it is desired to keep the spacecraft attitude constant during manipulator motion ($\boldsymbol{\omega}_{s_d} = \mathbf{0}$), e.g. not disturbing the orientation of some antenna for communication between spacecraft and earth. According to the above technique, the joint velocity solution can be computed as

$$\dot{\mathbf{q}} = \mathbf{J}_C^\dagger \mathbf{K}_C \mathbf{e}_C + (\mathbf{I} - \mathbf{J}_C^\dagger \mathbf{J}_C) \mathbf{J}_G^\top \mathbf{K}_P \mathbf{e}_{de} \quad (91)$$

with $\mathbf{e}_{de} = [\mathbf{e}_p^\top \quad \boldsymbol{\epsilon}_{de}^\top]^\top$.

If no other constraint is imposed and $n - m_e = m_s$, then $\mathbf{J}_C$ is given as in (90); it is assumed that $\mathbf{J}_C$ is non-singular, otherwise a transpose should be employed. By using the expression for $\mathbf{J}_G$ given in (87), one can write

$$\mathbf{J}_G = \bar{\mathbf{J}}_e + \bar{\mathbf{J}}_s \mathbf{J}_C; \quad (92)$$

then computing the matrix $\mathbf{J}_G(\mathbf{I} - \mathbf{J}_C^\dagger \mathbf{J}_C)$ leads to

$$\mathbf{J}_G(\mathbf{I} - \mathbf{J}_C^\dagger \mathbf{J}_C) = \bar{\mathbf{J}}_e(\mathbf{I} - \mathbf{M}_e^\dagger \mathbf{M}_e), \quad (93)$$

which coincides with the so-called *fixed-attitude-restricted (FAR) Jacobian* introduced in [Nenchev et al, 1990].

As a consequence, solution (91) provides end-effector trajectories not changing spacecraft attitude which are computed via the transpose of the fixed-attitude-restricted Jacobian and then can be simplified into

$$\dot{\mathbf{q}} = \mathbf{J}_C^\dagger \mathbf{K}_C \mathbf{e}_C + (\mathbf{I} - \mathbf{M}_e^\dagger \mathbf{M}_e) \bar{\mathbf{J}}_e^\top \mathbf{K}_P \mathbf{e}_{de}. \quad (94)$$

8. Flexible manipulators

In order to improve the performance of typically bulky industrial robots, one of the current trends is to adopt lightweight materials in the construction of manipulators. These are believed to offer a number of advantages such as smaller energy consumption, higher payload-to-arm weight ratio and faster movements [Book, 1993].

From a modeling standpoint, the scenario is complicated by the presence of additional deflection variables, compared to the case of rigid manipulators where the joint variables suffice to describe the system configuration.

Without loss of generality, consider a planar n -link *flexible* manipulator with revolute joints are considered which are subject only to bending deformations in the plane of motion, i.e. torsional effects are neglected. A sketch of a two-link arm is shown in Fig. 10 with coordinate frame assignment. The rigid motion is described by the joint angles ϑ_i , while $w_i(x_i)$ denotes the transversal deflection of link i at x_i , $0 \leq x_i \leq \ell_i$, being ℓ_i the link length.

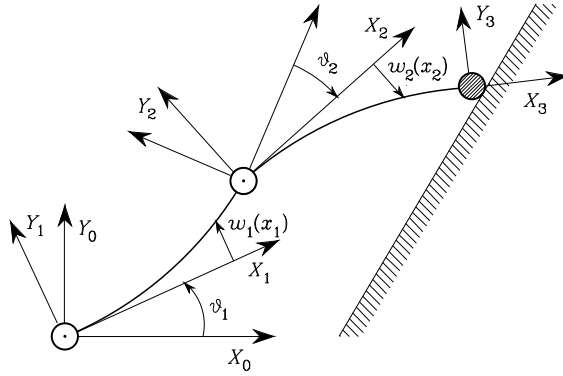


Figure 10. A Planar Two-Link Flexible Manipulator

Let ${}^i\mathbf{p}_i(x_i) = [x_i \ w_i(x_i)]^\top$ be the position of a point along the deflected link i with respect to frame (X_i, Y_i) and $\mathbf{p}_i$ be the position of the same point in the base frame. Also let ${}^i\mathbf{r}_{i+1} = {}^i\mathbf{p}_i(\ell_i)$ be the position of

the origin of frame (X_{i+1}, Y_{i+1}) with respect to frame (X_i, Y_i) , and r_{i+1} its position in the base frame.

The joint (rigid) rotation matrix R_i and the rotation matrix E_i of the (flexible) link at the end point are, respectively,

$$\mathbf{R}_i = \begin{bmatrix} \cos \vartheta_i & -\sin \vartheta_i \\ \sin \vartheta_i & \cos \vartheta_i \end{bmatrix} \quad (95)$$

and

$$\mathbf{E}_i = \begin{bmatrix} 1 & -w'_{ie} \\ w'_{ie} & 1 \end{bmatrix}, \quad (96)$$

where $w'_{ie} = (\partial w_i / \partial x_i)|_{x_i=\ell_i}$, and the small deflection approximation $\arctan w'_{ie} \simeq w'_{ie}$ has been made. Hence the above absolute position vectors can be expressed as

$$\mathbf{p}_i = \mathbf{r}_i + \mathbf{W}_i^i \mathbf{p}_i \quad (97)$$

and

$$\mathbf{r}_{i+1} = \mathbf{r}_i + \mathbf{W}_i^i \mathbf{r}_{i+1}, \quad (98)$$

where $\mathbf{W}_i$ is the global transformation matrix from the base frame to (X_i, Y_i) given by the recursive equation

$$\mathbf{W}_i = \mathbf{W}_{i-1} \mathbf{E}_{i-1} \mathbf{R}_i = \widehat{\mathbf{W}}_{i-1} \mathbf{R}_i, \quad (99)$$

with

$$\widehat{\mathbf{W}}_0 = \mathbf{I}. \quad (100)$$

On the basis of the above relations, the kinematics of any point along the manipulator is completely specified as a function of joint and link deflection.

A finite-dimensional model (of order m_i) of link flexibility can be obtained by the assumed modes technique. By exploiting separability in time and space of solutions to the Euler-Bernoulli equation for flexible beams

$$(EI)_i \frac{\partial^4 w_i(x_i, t)}{\partial x_i^4} + \rho_i \frac{\partial^2 w_i(x_i, t)}{\partial t^2} = 0, \quad (101)$$

for $i = 1, \dots, n$ where ρ_i is the uniform density and $(EI)_i$ is the constant flexural rigidity of link i , the link deflection can be expressed as

$$w_i(x_i, t) = \sum_{j=1}^{m_i} \phi_{ij}(x_i) \delta_{ij}(t), \quad (102)$$

where $\delta_{ij}(t)$ are the time-varying variables associated with the assumed spatial mode shapes $\phi_{ij}(x_i)$ of link i . The mode shapes have to satisfy

proper boundary conditions at the base (clamped) and at the end of each link (mass).

In view of (102), a direct kinematics equation can be derived expressing the position $\mathbf{p}$ of the manipulator tip point as a function of the joint variable vector $\boldsymbol{\theta} = [\vartheta_1 \ \dots \ \vartheta_n]^T$ and the deflection variable vector $\boldsymbol{\delta} = [\delta_{11} \ \dots \ \delta_{1m_1} \ \dots \ \delta_{n1} \ \dots \ \delta_{nm_n}]^T$, i.e.

$$\mathbf{p} = k(\boldsymbol{\theta}, \boldsymbol{\delta}). \quad (103)$$

For later use in the inverse kinematics scheme, also the differential kinematics is needed. The absolute linear velocity of a manipulator point is

$$\dot{\mathbf{p}}_i = \dot{\mathbf{r}}_i + \dot{\mathbf{W}}_i^i \mathbf{p}_i + \mathbf{W}_i^i \dot{\mathbf{p}}_i, \quad (104)$$

with ${}^i\dot{\mathbf{r}}_{i+1} = {}^i\dot{\mathbf{p}}_i(\ell_i)$. Since the links are assumed inextensible ($\dot{x}_i = 0$), then ${}^i\dot{\mathbf{p}}_i(x_i) = [0 \ \dot{w}_i(x_i)]^T$. The computation of (104) takes advantage of the recursion

$$\dot{\mathbf{W}}_i = \dot{\widehat{\mathbf{W}}}_{i-1} \mathbf{R}_i + \widehat{\mathbf{W}}_{i-1} \dot{\mathbf{R}}_i \quad (105)$$

with

$$\dot{\widehat{\mathbf{W}}}_i = \dot{\mathbf{W}}_i \mathbf{E}_i + \mathbf{W}_i \dot{\mathbf{E}}_i. \quad (106)$$

Also, note that

$$\dot{\mathbf{R}}_i = \mathbf{S} \mathbf{R}_i \dot{\vartheta}_i, \quad \dot{\mathbf{E}}_i = \mathbf{S} \dot{w}'_{ie} \quad (107)$$

with

$$\mathbf{S} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \quad (108)$$

In view of (102), it is not difficult to show that the tip velocity can be expressed as

$$\dot{\mathbf{p}} = \mathbf{J}_\vartheta(\boldsymbol{\theta}, \boldsymbol{\delta}) \dot{\boldsymbol{\theta}} + \mathbf{J}_\delta(\boldsymbol{\theta}, \boldsymbol{\delta}) \dot{\boldsymbol{\delta}}. \quad (109)$$

In a static situation the deflections are seen to satisfy the equation [De Luca and Siciliano, 1993]

$$\mathbf{g}_\delta(\boldsymbol{\theta}) + \mathbf{K} \boldsymbol{\delta} = \mathbf{0}, \quad (110)$$

where $\mathbf{g}_\delta$ is the gravity vector in the flexible dynamic equations that is only a function of $\boldsymbol{\theta}$ and $\mathbf{K}$ is the link stiffness matrix

$$\mathbf{K} = \text{diag}(k_{11}, \dots, k_{1m_1}, \dots, k_{n1}, \dots, k_{nm_n}) \quad (111)$$

with

$$k_{ij} = \int_0^{\ell_i} (EI)_i \phi_{ij}^2(x_i) dx_i. \quad (112)$$

From (110) the deflection variables can be computed as

$$\delta = -\mathbf{K}^{-1}\mathbf{g}_\delta(\theta). \quad (113)$$

For later use in the inverse kinematics scheme, differentiating (113) with respect to time gives

$$\dot{\delta} = -\mathbf{K}^{-1}\mathbf{J}_g(\theta)\dot{\theta}, \quad (114)$$

where $\mathbf{J}_g = d\mathbf{g}/d\theta$. Plugging (113) into (109) yields

$$\dot{\mathbf{p}} = \mathbf{J}_p(\theta, \delta)\dot{\theta}, \quad (115)$$

where

$$\mathbf{J}_p = \mathbf{J}_\vartheta - \mathbf{J}_\delta\mathbf{K}^{-1}\mathbf{J}_g \quad (116)$$

is the overall Jacobian matrix relating joint velocity to tip velocity. Notice that the Jacobian in (116) is obtained by modifying the rigid-body Jacobian $\mathbf{J}_\vartheta$ with a term that accounts for the deflections induced by gravity.

The kinematic control problem for a flexible manipulator can be formulated as follows: Given a desired constant tip position $\mathbf{p}_d$, find the corresponding joint variables and deflection variables that place the arm tip under gravity at the given position.

The attractive feature of the differential kinematics equation (115) is its formal analogy with the differential kinematics equation for a rigid manipulator. Therefore any Jacobian-based inverse kinematics scheme can be adopted in principle, e.g. applying the algorithm based on (6) gives [Siciliano, 1999b]

$$\dot{\theta} = \mathbf{J}_p^T(\theta, \delta)\mathbf{K}_p\mathbf{e}_p \quad (117)$$

with $\mathbf{e}_p = \mathbf{p}_d - \mathbf{p}$.

The approach can be easily extended to the case of a flexible manipulator in contact with a compliant surface [Siciliano and Villani, 2001].

Acknowledgements

The research work surveyed here wants to be a tribute to Bernard Roth. The authors have been greatly inspired not only by his publications in the field but more importantly by several discussions and private communications with him. Thanks a lot, Bernie, for showing us the way many times throughout the past twenty years!

References

- Antonelli, G., Chiaverini, S. (1998), Task-priority redundancy resolution for underwater vehicle-manipulator systems, *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Leuven, B, pp. 768–773.

- Antonelli, G., Chiaverini, S. (2003a), Fuzzy redundancy resolution and motion coordination for underwater vehicle-manipulator systems, *IEEE Transactions on Fuzzy Systems*, vol. 11, pp. 109–120.
- Antonelli, G., Chiaverini, S. (2003b), A fuzzy approach to redundancy resolution for underwater vehicle-manipulator systems, *Control Engineering Practice*, vol. 11, pp. 445–452.
- Book, W.J. (1993), Controlled motion in an elastic world, *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 115, pp. 252–261.
- Bottema, O., Roth, B. (1979), *Theoretical Kinematics*, North Holland, Amsterdam, The Netherlands.
- Caccavale, F., Chiaverini, S., Siciliano, B. (1997), Second-order kinematic control of robot manipulators with Jacobian damped least-squares inverse: Theory and experiments, *IEEE/ASME Transactions on Mechatronics*, vol. 2, pp. 188–194.
- Caccavale, F., Natale, C., Siciliano, B., Villani, L. (1998), Resolved-acceleration control of robot manipulators: A critical review with experiment, *Robotica*, vol. 16, pp. 565–573.
- Caccavale, F., Siciliano, B. (2001), Kinematic control of redundant free-floating robotic systems, *Advanced Robotics*, vol. 15, pp. 429–448.
- Chiacchio, P., Chiaverini, S., Sciavicco, L., Siciliano, B. (1991), Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy, *International Journal of Robotics Research*, vol. 10, pp. 410–425.
- Chiacchio, P., Chiaverini, S., Siciliano, B. (1996), Direct and inverse kinematics for coordinated motion tasks of a two-manipulator system, *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 118, pp. 691–697.
- Chiacchio, P., Siciliano, B. (1989), A closed-loop Jacobian transpose scheme for solving the inverse kinematics of nonredundant and redundant wrists, *Journal of Robotic Systems*, vol. 6, pp. 601–630.
- Chiaverini, S. (1993), Estimate of the two smallest singular values of the Jacobian matrix: application to damped least-squares inverse kinematics, *Journal of Robotic Systems*, vol. 10, pp. 991–1008.
- Chiaverini, S. (1997), Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators, *IEEE Transactions on Robotics and Automation*, vol. 13, pp. 398–410.
- Chiaverini, S., Siciliano, B. (1999), The unit quaternion: A useful tool for inverse kinematics of robot manipulators, *Systems Analysis, Modelling and Simulation*, vol. 35, pp. 45–60.
- Chiaverini, S., Siciliano, B., Egeland, O. (1994), Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator, *IEEE Transactions on Control Systems Technology*, vol. 2, pp. 123–134.
- De Luca, A., Siciliano, B. (1993), Regulation of flexible arms under gravity, *IEEE Trans. on Robotics and Automation*, vol. 9, pp. 463–467.
- Khatib, O. (1987), A unified approach for motion and force control of robot manipulators: The operational space formulation, *IEEE Journal of Robotics and Automation*, vol. 3, pp. 43–53.
- Merlet, J.-P. (2000), *Parallel Robots*, Kluwer, Dordrecht, The Netherlands.

- Nakamura, Y., Hanafusa, H., Yoshikawa, T. (1987), Task-priority based redundancy control of robot manipulators, *International Journal of Robotics Research*, vol. 6, no. 2, pp. 3–15.
- Nakamura, Y., Mukherjee, R. (1991), Nonholonomic path planning of space robots via bi-directional approach, *IEEE Transactions on Robotics and Automation*, vol. 7, pp. 500–514.
- Nenchev, D., Yoshida, K., Umetani, Y. (1990), Analysis, design and control of free-flying space robots using fixed-attitude-restricted Jacobian matrix, in *Robotics Research – 5th International Symposium*, Miura, H., Arimoto, S. (Eds.), MIT Press, Cambridge, MA, pp. 251–258.
- Nenchev, D., Umetani, Y., Yoshida, K. (1992), Analysis of a redundant free-flying spacecraft/manipulator system, *IEEE Transactions on Robotics and Automation*, vol. 8, pp. 1–6.
- Pieper, D., Roth, B. (1969), The kinematics of manipulators under computer control, *Proceedings of the Second International Congress on Theory of Machines and Mechanisms*, Zakopane, Poland, vol. 2, pp. 159–169.
- Raghavan, M., Roth, B. (1995), Solving polynomial systems for the kinematics analysis and synthesis of mechanisms and robot manipulators, *Transactions of the ASME, Special 50th Anniversary Design Issue*, vol. 117, pp. 71–79.
- Sciavicco, L., Siciliano, B. (1986), Coordinate transformation: A solution algorithm for one class of robots, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 16, pp. 550–559.
- Sciavicco, L., Siciliano, B. (1988), A solution algorithm to the inverse kinematic problem for redundant manipulators, *IEEE Journal of Robotics and Automation*, vol. 4, pp. 403–410.
- Sciavicco, L., Siciliano, B. (2000), *Modelling and Control of Robot Manipulators*, 2nd Edition, Springer-Verlag, London, UK.
- Shepperd, S.W. (1978), Quaternion from rotation matrix, *AIAA J. of Guidance and Control*, vol. 1, pp. 223–224.
- Siciliano, B. (1990a), Kinematic control of redundant robot manipulators: A tutorial, *Journal of Intelligent and Robotic Systems*, vol. 3, pp. 201–212.
- Siciliano, B. (1990b), A closed-loop inverse kinematic scheme for on-line joint-based robot control, *Robotica*, vol. 8, pp. 231–243.
- Siciliano, B. (1999a), The Tricept robot: Inverse kinematics, manipulability analysis and closed-loop direct kinematics algorithm, *Robotica*, vol. 17, pp. 437–445.
- Siciliano, B. (1999b), Closed-loop inverse kinematics algorithms for constrained flexible manipulators under gravity, *Journal of Robotic Systems*, vol. 16, pp. 353–362.
- Siciliano, B., Villani, L. (2001), An inverse kinematics algorithm for interaction control of a flexible arm with a compliant surface, *Control Engineering Practice*, vol. 9, pp. 191–198.
- Umetani, Y., Yoshida, K. (1989), Resolved motion rate control of space manipulators with generalized Jacobian matrix, *IEEE Transactions on Robotics and Automation*, vol. 5, pp. 303–314.
- Whitney, D.E. (1969), Resolved motion rate control of manipulators and human prostheses, *IEEE Transactions on Man-Machine Systems*, vol. 10, pp. 47–53.
- Whitney, D.E. (1972), The mathematics of coordinated control of prosthetic arms and manipulators, *ASME J. of Dynamic Systems, Measurement, and Control*, vol. 94, pp. 303–309.