

Lorenzo Sciacivco and Bruno Siciliano

Dipartimento di Informatica e Sistemistica
 Università di Napoli
 Via Claudio 21, 80125 Napoli, Italy

Abstract

A systematic way of designing a robot control unit which presents the characteristics of simplicity of implementation and modularity is presented in this paper. Two strategies are adopted and compared, one in the joint space and the other in the task space, both leading to a parallel multi-processor decentralized system which appears very promising for real-time robot control.

The Robot Control Problem

The typical goal a robot control unit must achieve is to make the robot's end-effector execute prescribed motions/forces which are assigned in a suitable coordinate frame. These motion/force requirements are naturally given in the so-called task space, that is the same coordinate frame in which the robot and the working environment are described.

On the other hand, the mechanical structure of a robotic manipulator consists of a chain (usually an open chain) of rigid bodies linked by means of articulated joints. A number of actuators develop generalized forces at the joints, rotational torques for revolute joints and axial forces for prismatic joints respectively. These forces are the control inputs at designer's disposal, whereas the relative positions of the consecutive rigid bodies are the joint variables to be controlled such that the robot may execute the prescribed task. The joint variables constitute a set of generalized coordinates for the mechanical system in the so-called joint space, in which control actions are naturally developed. For the purpose of "transforming" the control objective, specified in the task space, into a control strategy in the joint space, it is crucial, therefore, to characterize the relation between the two spaces; i.e.,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{q}) \quad (1)$$

where $\mathbf{x}$ is the vector of variables which describe position and orientation of the robot's end-effector, $\mathbf{q}$ is the vector of joint variables, and $\mathbf{f}$ is a continuous, nonlinear vector-valued function, whose structure is completely known, once the kinematic parameters of the manipulators are specified [1].

Solving equation (1) for the joint variables $\mathbf{q}$ certainly is the key point for robot control. As a matter of fact, eq. (1) is solvable only for simple manipulator geometries, such as the usual spherical wrist configuration, for which the end-effector position is independent of its orientation. Pieper [1] gave a sufficient condition under which a kinematic structure is solvable, i.e. at least three consecutive revolute axes must intersect in one point. Nevertheless, a structure that does not satisfy the above condition can be found to be solvable, such as the so-called elbow manipulator which has three consecutive parallel axes. Furthermore,

in case of a redundant structure, i.e. a structure that has more degree-of-freedom's than it is needed to execute a certain task, equation (1) is not solvable directly, unless an opportune number of constraints are added to the problem, so as to "span" the space of redundancy available.

Usually, another relation is considered together with the kinematic equation (1). It is formally obtained by differentiating eq. (1) with respect to time, i.e.

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

where $\dot{\mathbf{x}}$ is the vector of end-effector velocities, $\dot{\mathbf{q}}$ is the vector of joint velocities, and $\mathbf{J}(\mathbf{q})$ is the so-called Jacobian matrix $\partial \mathbf{f} / \partial \mathbf{q}$.

On the basis of equation (2), it is possible to compute the joint velocities which correspond to a given end-effector velocity, i.e. the increments $\delta \dot{\mathbf{q}}$ corresponding to the increments $\delta \dot{\mathbf{x}}$. This procedure fails, however, when a kinematic singularity occurs (the matrix $\mathbf{J}(\mathbf{q})$ has not full rank), i.e. when the manipulator attains a configuration at which it has a reduced mobility, loosing the possibility to move along or rotate about a certain direction in space.

Kinematic Control

The equations (1) and (2) completely describe the direct kinematic model of a robotic manipulator. These equations are also at the basis of the control solution normally adopted in today's industrial robots, i.e. the so-called kinematic control, whose block scheme is illustrated in Fig. 1. It is easy to recognize that the scheme of Fig. 1 is a decentralized control on the single joint variables, whose coordination occurs only at the stage of the inverse kinematics. More specifically, once a trajectory $\dot{\mathbf{x}}(t)$ is assigned to the robot's end-effector, a corresponding joint trajectory $\dot{\mathbf{q}}(t)$ is sought, which constitutes the reference input to the joint servos. In practice, the inversion of the kinematic equation (1) cannot be made continuously, but only for a finite number of points $\dot{\mathbf{x}}_k \rightarrow \dot{\mathbf{q}}_k$ [1]. Either in the case the kinematic structure is solvable, or in the case equation (2) is inverted, the computational burden sets an upper limit on the sampling rate, that is usually around 100 Hz. In the former case, indeed, the inversion technique is naturally sequential (one joint variable at time), whereas in the latter case the inversion of the Jacobian matrix is time-consuming. In Fig. 1 it is shown also an interpolator, which is necessary in order to obtain rates higher than 300 Hz, so as demanded by the joint servos.

Dynamic Control

The above control strategy does not account for masses, inertias, backlashes and elasticities at all, which modify the dynamic behavior of the structure. Thus, if high performance is desired in terms of high

operational velocities, and high tracking accuracy, the kinematic control is likely to fail. In that case, it is necessary to consider the robot nonlinear dynamic model which, under the assumption of rigid bodies, can be written as [1]

$$M(q)\ddot{q} + h(q, \dot{q}) + g(q) = \tau \quad (3)$$

where M is the generalized positive definite inertia matrix, h is the vector of Coriolis and centrifugal forces, g is the vector of gravitational forces, and τ is the vector of generalized forces developed at joints.

The knowledge of the robot dynamic model (3) is exploited in the so-called dynamic control. This approach is based on the computation of the so-called inverse dynamics. In other words, once the reference joint trajectories are available ($\ddot{q}(t)$, $\dot{q}(t)$, $q(t)$), under the assumption of negligible disturbance forces, one can utilize eq. (3) to compute the generalized forces τ which are to be developed at joints in order to obtain the desired trajectories. Being this control strategy open-loop, it usually needs a feedback loop to compensate for the deviations from the nominal motion imposed [1]. In any event, the success of this strategy relies on the possibility of obtaining "good" estimates of the dynamic terms in (3) on-line, which may be much too computational demanding. In order to overcome that drawback, alternative control strategies have been recently proposed which are based on a partial knowledge of the robot dynamic model [2-4]. The supporting idea is to design a feedforward compensator for a "reduced" dynamic model plus a decentralized "robust" feedback controller at joints so as to obtain satisfactory dynamic performance. This approach will be outlined in the following section.

The Proposed Solution

It should be clear now from the discussion above that in order that a robot control system be effective, it must have the following requirements:

- 1) easy to implement, without requiring a very accurate dynamic model on-line,
- 2) good motion/force profile tracking accuracy, in spite of inertia and load variations,
- 3) possibility of designing a decentralized control for the single degree-of-freedom's.

A solution that satisfies all the above requirements is illustrated by the block scheme of Fig. 2. First, a fast inverse kinematic computational algorithm is utilized [5], which allows the direct computation of the joint trajectories corresponding to a given end-effector trajectory. No interpolation is needed, since the algorithm guarantees small tracking errors and fast computation time. Then, a centralized control is in charge to "lighten" the dynamic model to be controlled by means of robust decentralized servos at joints, that guarantee high dynamic performance. In the following the features of the inverse kinematic algorithm and of the dynamic controller are presented.

The Inverse Kinematic Algorithm

It has been widely argued that, in order to provide the manipulator controlled in the joint space with the reference inputs, a coordinate transformation from the task space to the joint space is needed (inverse kinematic problem). Any inverse kinematic technique is to be evaluated on the basis of a trade-off between computation and precision requirements. It results from the robotics literature [1] that no exact kinematic inversion routines exist for general kinematic structures, but only a limited number of software packages specially derived for simple geometries on the basis of geometric intuition. This point has recently motivated

these authors to seek alternative computational algorithms which not only compete with the existing closed-form solutions for solvable geometries, but mainly be suitably adopted for general unsolved geometries [5]. The basic block scheme of such algorithms is illustrated in Fig. 3.

Let $\hat{x}(t)$ be the desired trajectory at the robot's end-effector (position + orientation), and let $x(t)$ be the trajectory computed via the direct kinematic function $f(q)$. It can be shown that, by means of a proper choice of the positive definite matrix K , it is possible to ensure that the tracking error is bounded (with $x(0) = \hat{x}(0)$), and the steady-state error ($\dot{x} = 0$) is null. If this holds, the algorithm solution $q(t)$ well approximates the desired solution $\hat{q}(t)$. The proof of algorithm convergence can be derived by the Lyapunov direct method for the analysis of the stability of nonlinear systems [5-8]. It is useful here to report the attractive features of the solution proposed:

- 1) it only requires the computation of direct kinematic functions (f, J); this allows for sampling rates higher than 500 Hz for typical three-to-six degree-of-freedom manipulators,
- 2) it allows for parallel multi-processor computation, on condition that q is available on a common bus,
- 3) it can be utilized for any kinematic structure, redundant or nonredundant; in particular for a redundant structure it is possible to introduce a number of motion constraints in a systematic way, such as obstacle avoidance, limited joint range, and so forth [6],
- 4) joint velocities $\dot{q}$ are directly generated without requiring any further computation,
- 5) it avoids the drawback of multiple solutions which is typical of the analytical technique when interpolation between two configurations with different aspects, say "elbow-up" and "elbow-down", is attempted; the proposed algorithm generates "adjacent" joint configurations, starting from the initial configuration,
- 6) it allows for solutions even in case of kinematic singularities encountered along the trajectory; no inversion of kinematic functions is required (J^{-1}) [5-8],
- 7) it leads to an even simpler computational scheme in case of typical kinematic designs at the end-effector, such as "spherical wrist" [7], "two-by-two intersecting axes" [8] etc.; indeed, the algorithm can be suitably partitioned into two stages, one relative to end-effector position and the other relative to end-effector orientation;
- 8) if utilized off-line, it allows for the evaluation of the manipulator static performance, such as the determination of "dexterous" configurations, as well as the force/velocity control/actuation compatibility along a given direction in the task space [9], the tracing of the robot workspace [10], and so forth.

The Robust Decentralized Control

Among the several dynamic control techniques proposed in the literature [2,3], the approach which is based on the use of a reference model to follow seems one of the most attractive [11]. The scheme of Fig. 4 is considered, where Λ_0 and Λ_1 are two diagonal matrices which define an asymptotically stable reference model. First, it can be recognized that the centralized control of the principle scheme of Fig. 2 is actually formed by two separate nonlinear actions, feedforward and feedback respectively. Then, a static compensator D combines the position and velocity errors, where D is determined through the off-line solution of the Lyapunov equation associated with the reference model [11]. Finally, the decentralized control is simply given by independent robust joint controllers. The proposed solution presents the following features:

- 1) it allows for parallel multi-processor computation, on condition that $\dot{q}$, $\ddot{q}$ are available on a common bus,
- 2) the nonlinear compensations have the function of balancing the effects of time-varying inertial and gravitational terms, but they can be further simplified by compensating only for the dominant inertias (diagonal terms) and constant gravitational terms,
- 3) the robust joint servos have the function of counteracting the effects of noncompensated dynamic terms (centrifugal and Coriolis terms, together with off-diagonal inertial terms and time-varying gravitational terms eventually), as well as the function of counteracting the effects of parametric uncertainties and inaccurate modeling.

Dynamic Control in the Task Space

The dynamic control strategy in the joint space presented in the previous section consists of two stages: the inverse kinematic algorithm and the robust decentralized joint control. An alternative way of controlling a robot is represented by a dynamic control in the task space which does not apparently require any transformation from the task space to the joint space. At the basis of this approach there is the relation between the vector of generalized forces applied at the end-effector $\bar{Y}$ and the vector of generalized forces applied at joints τ , through the transpose of the Jacobian matrix [1], i.e.

$$\tau = J^T \bar{Y} \quad (4)$$

which is "dual" of equation (2). This relation, indeed, is adopted also to analyze the manipulator static performance, so as anticipated at point 8) in the section on the inverse kinematic algorithm.

For the purpose of designing a dynamic control in the task space, the vector $\bar{Y}$ is assumed as control input and an equivalent robot dynamic model is derived in the task space [12,13]. The scheme of Fig. 5 is considered, which is formally equivalent to the scheme of Fig. 4 from the control standpoint. The occurrence of the matrices J_1 and J is justified by the necessity of transforming accelerations and forces in the task space respectively into accelerations and forces in the joint space, according to eqs. (2) and (4). Also, the vector $\bar{f}$ and the matrix J in the feedback loop have been introduced in order to reconstruct task position and velocity measurements from joint position and velocity measurements respectively. The features of the proposed solution can be summarized as follows:

- 1) it allows for parallel multi-processor computation, on condition that $\dot{q}$, $\ddot{q}$ are available on a common bus,
- 2) the computational burden is heavier, due to the more complex structure of the centralized control, but no kinematic inversion is needed,
- 3) the only nonlinear terms to be computed on-line with high accuracy are the direct kinematic functions $\bar{f}$, J ,
- 4) the task direction robust controllers have the function of balancing the effects of those noncompensated dynamic terms, as well as the function of counteracting the effects of parametric uncertainties and inaccurate modeling (similar remarks about the computation accuracy of such terms are in order so as in the previous section),
- 5) it is "transparent" to the problems involved in the kinematic inversion (singularities, redundancy, etc.),
- 6) it is apparently advantageous for typical hybrid tasks (position + force) at the robot's end-effector, that is for all those cases when there is interaction of the end-effector with the

workpiece,

- 7) in case of a redundant manipulator, it is necessary to introduce a number of constraints so as to span the space of redundancy; the square Jacobian matrix of the scheme of Fig. 5 will be that matrix which includes the derivatives of such constraints with respect to the joint variables.

Conclusions and Acknowledgements

The problem of systematic design of robot control units has been the focus of this contribution. Two control strategies have been presented: a two-stage joint space control and a task space control. Pros and cons of the two approaches have been discussed. The strong point is that both lead to a parallel multi-processor decentralized control system which can be regarded as a standard to design robot control units. A high number of simulation results have confirmed the validity of both approaches. Further research is certainly needed to individuate those computational tools which allow the implementation of the proposed control laws, as well as to run some physical experiments on an actual industrial robot.

The support of this research by MPI is gratefully acknowledged.

References

- [1] J.J. Craig, Introduction to Robotics: Mechanics and Control. Reading, MA: Addison-Wesley, 1986.
- [2] J.J. Craig, Adaptive Control of Mechanical Manipulators. Reading, MA: Addison-Wesley, 1988.
- [3] H. Asada and J.-J.E. Slotine, Robot Analysis and Control. New York, NY: Wiley-Interscience, 1986.
- [4] G. De Maria, L. Sciavicco, and B. Siciliano, "Robust control for industrial robots," 1st IFAC SY.RO.CO., Barcelona, Spain, Nov. 1985.
- [5] L. Sciavicco and B. Siciliano, "Solving the inverse kinematic problem for robotic manipulators," 6th CISM-IFTOMM Ro.Man.Sy., Cracow, Poland, Sept. 1986.
- [6] L. Sciavicco and B. Siciliano, "A solution algorithm to the inverse kinematic problem for redundant manipulators," IEEE J. Rob., Autom., 1988, to be published.
- [7] A. Balestrino, G. De Maria, L. Sciavicco, and B. Siciliano, "An algorithmic approach to coordinate transformation for robotic manipulators," Advanced Robotics, 1988, to be published.
- [8] L. Sciavicco and B. Siciliano, "Coordinate transformation: A solution algorithm for one class of robots," IEEE Trans. Syst., Man, Cybern., vol. SMC-16, no. 4, July/Aug. 1986.
- [9] L. Sciavicco and B. Siciliano, "An inverse kinematic solution algorithm for dexterous redundant manipulators," 3rd ICAR, Versailles, France, Oct. 1987.
- [10] L. Sciavicco and B. Siciliano, "Inverse kinematics and related issues: A unified approach to their solution," 5th Yugoslav Symp. Rob., Flex. Autom., Bled, Yugoslavia, June 1987.
- [11] A. Balestrino, G. De Maria, and L. Sciavicco, "An adaptive model following control for robotic manipulators," ASME J. Dyn., Syst., Meas., Contr., vol. 105, no. 3, Sept. 1983.
- [12] A. Balestrino, G. De Maria, and L. Sciavicco, "Adaptive control of manipulators in the task oriented coordinate space," 13th ISIR/Robots 7, Chicago, IL, Apr. 1983.
- [13] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," IEEE J. Rob., Autom., vol. 3, no. 1, Feb. 1987.

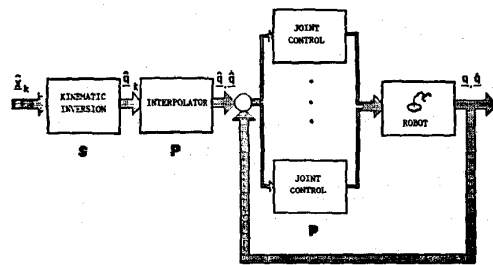


Fig. 1 - The kinematic control scheme.

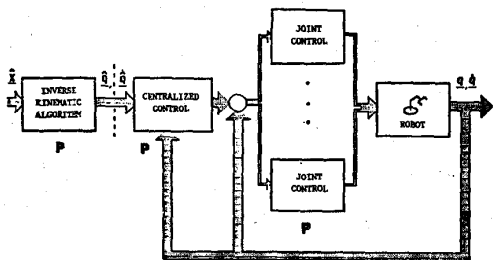


Fig. 2 - The proposed two-stage control scheme.

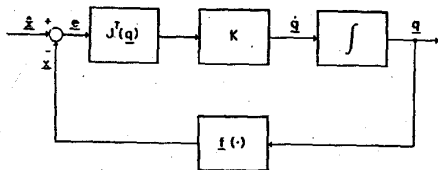


Fig. 3 - The inverse kinematic algorithm scheme.

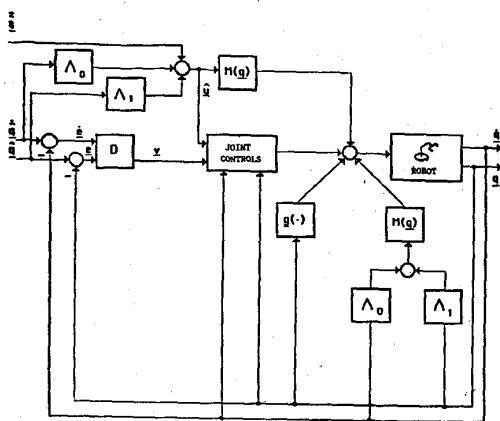


Fig. 4 - The joint space dynamic control scheme.

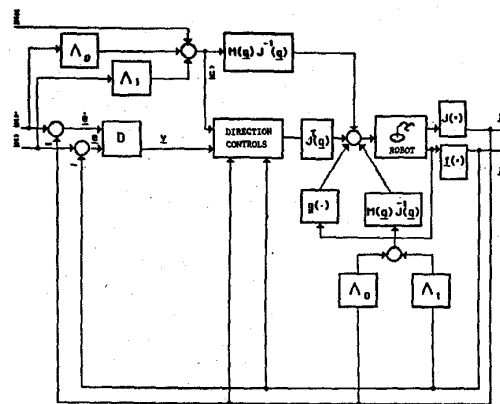


Fig. 5 - The task space dynamic control scheme.