An optimum design procedure for both serial and parallel manipulators

G Carbone*, **E Ottaviano,** and **M Ceccarelli**
Laboratory of Robotics and Mechatronics, DIMSAT, University of Cassino, Cassino, Italy

The manuscript was received on 21 April 2006 and was accepted after revision for publication on 30 March 2007.

DOI: 10.1243/0954406JMES367

Abstract: Serial and parallel manipulators can be used in different manipulative tasks when their peculiarities in kinematic and dynamic behaviours are properly considered from the design stage. The basic performance in workspace, mobility constraints, and stiffness makes them alternative solutions and not competitive manipulator chains. Thus, it is convenient to deduce a common design procedure that considers common design criteria, but specific numerical evaluations. In this paper, a multi-objective optimization problem has been proposed to formulate a unique design procedure that takes into account the contradicting design optimality criteria in terms of suitable general algorithms for workspace volumes, Jacobian matrices, and compliant displacements. Numerical examples are reported to show not just the feasibility but also the numerical efficiency of the proposed formulations.

Keywords: robotics, manipulator design, robot performance, optimum design procedure

1 INTRODUCTION

Robotized manipulations are widely used in industrial applications and even in non-industrial environments, manipulators are needed to help human beings and/or to execute manipulative tasks.

The duality between serial and parallel manipulators is not anymore understood as a competition between the two kinematic architectures. The intrinsic characteristics of each architecture make each architecture as devoted preferably to some manipulative tasks more than an alternative to the counterpart. The complementarities of operation performance of serial and parallel manipulators make them as a complete solution for manipulative operations. The differences but complementarities in their performance have given the possibility in the past to treat them separately, mainly for design purposes. Several analysis results and design procedures have been proposed in a very rich literature in the last two decades.

Significant works on the topics can be found in the pioneer papers [**1**–**8**] and, more recently in the papers [**9**–**11**]; just to cite a few references with a very rich literature. Algorithms have been proposed for the design of the Gough–Stewart platform as based on workspace characteristics [**12**, **13**], stiffness properties [**14**], and global isotropy property [**15**], separately.

Only recently, it has been possible to consider simultaneously several design aspects in design procedures for manipulators. Multi-criteria optimal designs have been proposed for example in references [**16**–**18**] as applied to parallel manipulators. Modern design procedures make use more and more of the formulation of optimization problems that can be solved by using well-established mathematical techniques in commercial software packages.

Since the beginning of 1990s at LARM: Laboratory of Robotics and Mechatronics in Cassino, a research line has been dedicated to the development of analysis formulation of manipulator performance [**19**–**25**], that could be used in design algorithms and even in proper optimization problems by taking advantage of the peculiarity of solving techniques in commercial software packages [**26**–**30**]. Recent results are reported in reference [**31**] as regarding serial manipulators, and in reference [**17**] as referring to parallel manipulators, just to cite illustrative experiences. However, since manipulative characteristics are fundamental for the operation and design of both manipulator

* *Corresponding author: Laboratory of Robotics and Mechatronics (LARM), DIMSAT, University of Cassino, Via Di Biasio 43, 03043 Cassino, Italy. email: carbone@unicas.it*

architectures, the authors have attempted analysis procedures that could be used with few adjustments and no great computational efforts in the performance analysis both for serial and parallel architectures. The several experiences have been summarized in the recent textbook [**32**], in which the analysis of workspace, singularity, and stiffness has been attached in a unified approach.

Following this idea, in this paper an attempt has been made to treat the design problem of serial and parallel manipulators in a unified formulation, since the optimality design criteria can be differentiated only in a characterization of results and not so much in the numerical algorithms. Therefore, in this paper basic characteristics for manipulation purposes, such as workspace, singularity, and stiffness are overviewed with numerical evaluation procedures that are useful in a design optimization problem both for serial and parallel manipulators. The feasibility of the approach and proposed formulation has been tested and illustrative examples are reported, also with the aim to clarify the computational efforts.

2 THE PROBLEM AND ITS FORMULATION

Manipulators are said to be useful to substitute/help human beings in manipulative operations and therefore their basic characteristics are usually referred to human manipulation performance aspects.

A well-trained person is usually characterized for manipulation purpose mainly in terms of positioning skill, arm mobility, arm power, movement velocity, and fatigue limits. Similarly, robotic manipulators are designed and selected for manipulative tasks by looking mainly at workspace volume, payload capacity, velocity performance, and stiffness. Therefore, it is quite reasonable to consider those aspects as fundamental criteria for manipulator design. Because they can give contradictory results in design algorithms, a formulation can be convenient as multi-objective optimization problem in order to consider them simultaneously. Thus, the optimum design of manipulators can be formulated the form proposed in reference [**33**], as

$$\min_{\boldsymbol{X}} [\boldsymbol{F}(\boldsymbol{X})] = \min_{\boldsymbol{X}} \left\{ \max_{i=1,\ldots,N} \left[w_i f_i(\boldsymbol{X}) \right] \right\} \tag{1}$$

subjected to

$$\boldsymbol{G}(\boldsymbol{X}) < 0; \tag{2}$$

$$\boldsymbol{H}(\boldsymbol{X}) = 0 \tag{3}$$

where min is the operator for calculating the minimum of a vector function $\boldsymbol{F}(\boldsymbol{X})$; similarly max determines the maximum value among the N functions $[w_i f_i(\boldsymbol{X})]$ at each iteration; $\mathbf{G}(\boldsymbol{X})$ is the vector of constraint functions that describes limiting conditions, and $\mathbf{H}(\boldsymbol{X})$ is the vector of constraint functions that describes design prescriptions; $\boldsymbol{X}$ is the vector of design variables.

The proposed optimization formulation uses the objective function $\boldsymbol{F}(\boldsymbol{X})$ at each iteration by choosing the worst-case value among all the scalar objective functions for minimizing it in the next iteration, as outlined in reference [**34**]. In particular, the worst-case value is selected in equation (1) at each iteration as the objective function with maximum value among the N available objective functions. This approach for solving multi-objective problems with several objective functions and complex trade-offs among them is known as 'minimax method', [**33**]. The 'minimax method' is widely indicated in the literature for many problems, like for example for estimating model parameters by minimizing the maximum difference between model output and design specification, as outlined for example in references [**35**] to [**37**].

Additionally, it is worth to note that the proposed scalar objective functions $f_i(\boldsymbol{X})$ have been formulated to be dimensionless. Moreover, weighting factors w_i (with $i = 1, \ldots, N$) have been used in order to scale all the objective functions. In particular, weighting factors w_i are chosen so that each product $w_i f_i(\boldsymbol{X})$ is equal to one divided by N for an initial guess of a design case. The above-mentioned conditions on the objective functions can be written in the form

$$\sum_{i=1}^{N} \left(w_i f_i \right)_0 = 1 \tag{4}$$

$$N \left(w_i f_i \right)_0 = 1 \tag{5}$$

where the subscript 0 indicates that the values are computed at an initial guess of the design case. Bigger/lower weighting factors can be chosen in order to increase/reduce the significance of an optimal criterion with respect to others.

Although interval analysis [**38**], can be very effective for an optimal design algorithm it can have too high computational costs. Therefore, numerical procedures are still widely used in optimization processes even if they can suffer of known drawbacks. Some algorithms such as flooding techniques, simulated annealing, genetic algorithms can be faster in finding an optimal solution with a single objective function. However, they still cannot guarantee the convergence and that an optimal solution is a global optimum. Also they cannot guarantee that any single optimality criterion is satisfied. Thus, min–max code of the Matlab optimization toolbox [**33**], has been used in the proposed optimal design procedure. The selected 'minmax' algorithm in Matlab environment is a suitable attempt to address the last point. In fact, it has

been specifically designed for solving multi-objective problems. Therefore, this algorithm has been selected as a suitable compromise among computational costs and accuracy of results.

In addition, optimality criteria for manipulator design can be identified in performance evaluations regarding positioning and orientating capability, velocity response, and static behaviour.

Positioning and orientation capability can be evaluated by computing position and orientation workspaces that give the reachable regions by the manipulator extremity as function of the mobility range of the manipulator joints. Position workspace refers to reachable points by a reference point on manipulator extremity, and orientation workspace describes the angles that can be swept by reference axes on manipulator extremity. Thus, an objective function f_1 can be formulated as regarding with a numerical evaluation f_{PW} of the position workspace in the form

$$f_1 = f_{\mathrm{PW}} \tag{6}$$

Similarly, a numerical evaluation f_{OW} of the orientation workspace can be used as objective function as

$$f_2 = f_{\mathrm{OW}} \tag{7}$$

Velocity response can be evaluated by looking at the velocity mapping that can be described by the Jacobian of a manipulator. The Jacobian is also useful to identify singular configurations (singularities) of a manipulator at which degrees of freedom (DOF) are lost or gained producing undesirable motion uncertainties or self-motions that should be avoided in a controlled movement. Thus, Jacobian evaluation f_J can be used as objective function as

$$f_3 = f_{\mathrm{J}} \tag{8}$$

Static behaviour can be evaluated by computing the stiffness characteristics that are responsible also for the accuracy of the manipulative operation. Therefore, compliant response can be conveniently used as optimality design criterion, when similarly to the workspace capability the compliance is evaluated through the position and orientation counterparts. An evaluation f_{ST} of linear compliant displacements can be used as objective function as

$$f_4 = f_{\mathrm{ST}} \tag{9}$$

as well as an evaluation f_{SO} of the angular compliant displacements as

$$f_5 = f_{\mathrm{SO}} \tag{10}$$

Thus, the multi-objective function $\boldsymbol{F}$ can be formulated with computer-oriented algorithms when its components f_i $(i = 1, \ldots, 5)$ are computed numerically through suitable analysis procedures. It is worth to note that the above-mentioned split into translational and orientational aspects can be adopted also for Jacobian analysis and, however, it has been thought mainly for simplifying the computational efforts and for interpretation aims, without neglecting the coupling effects because of spatial design and operation of manipulators.

Similarly, the constraint functions $\boldsymbol{G}$ and $\boldsymbol{H}$ can be formulated by using suitable evaluation of design and operation constraints as well as those additional constraints that are needed for computational issues.

Thus, the problem for achieving optimal results from the formulated multi-objective optimization problem consists mainly in two aspects, namely to formulate the optimality criteria with computational efficiency and to choose a proper numerical solving technique.

Indeed, the solving technique can be selected among the many available, even in commercial software packages, by looking at a proper fit and/or possible adjustments to the formulated problem in terms of number of unknowns, non-linearity type, and involved computations for the optimality criteria and constraints. On the other hand, the formulation and computations for the optimality criteria and design constraints can be conceived and performed by looking also at the peculiarity of a chosen numerical solving technique.

Those two aspects can be very helpful in achieving an optimal design procedure that can give solutions with no great computational efforts and with possibility of engineering interpretation and guide. Since the formulated design problem is intrinsically high non-linear, a solution is obtained when the numerical evolution from a tentative solution converges because of the iterative process to a solution that can be considered optimal within the explored range of design variable domain. Therefore, a solution can be considered an optimal design as a local optimum, in general terms. This last remark makes clear once more the influence of a suitable formulation with computational efficiency for the involved criteria and constraints in order to have a design procedure, which is significant from engineering viewpoint and numerically efficient.

3 PERFORMANCE EVALUATION FOR OPTIMALITY CRITERIA

Once a numerical technique is chosen or is advised for solving the proposed multi-objective optimization problem, the main efforts can be addressed to the formulation of common algorithms for numerical

evaluation of optimality criteria and design procedure constraints.

In particular, in this paper optimality criteria have been chosen according to the most common design requirements for robotic systems, which are the workspace both for position and orientation capabilities, singularities, and stiffness behaviour. In fact, is of great interest to design a robot with desired workspace, free of singularities and with adequate stiffness. The latter two requirements are also related to robot accuracy and path planning.

The choice of each design criterion has been made according to two aspects, namely computational efficiency and possibility to be used both for serial and parallel manipulators.

Flexibility for computational issues can be reached by using a general formulation that can have even computational complexity or can require certain computational efforts. Alternatives in formulating and choosing optimality criteria are always possible depending of the designer experience, design goals, and manipulator applications. Indeed, any choice of optimality criteria can be questionable when considering the above-mentioned aspects. In fact, many different indices have been proposed in a rich literature on manipulators both for analysis and design purposes. Those indices can be used and they have been used with proper formulation as optimality criteria in specific algorithms for optimal design of specific manipulators. In this paper, possible alternatives of formulation have not been mentioned for the proposed optimality criteria since the main scope of the paper is to show the possibility to use a unique design procedure for designing serial and parallel manipulators. Of course, any optimality criteria as well as their formulation can suffer drawbacks in terms of conceptual aim and numerical efficiency, but in this paper focus has been made on their generality and numerical versatility. Within such aspects, computation of the proposed optimality criteria can require computational efforts, like for example those for time-consuming iterative or scanning processes, and they can need additional numerical constraints, like for example those for avoiding singular numerical situations and bounding the feasible ranges for design parameters. The generality of a formulation, which can ensure a successful application to any manipulator topology, has been treated as much as possible by using standard expressions with adequate precisions with the aim to give a fairly simple numerical evaluation of the proposed optimality criteria and to obtain numerical versatility of the algorithms for efficiency within the overall design procedure.

In the following, main aspects are overviewed by emphasizing the common numerical evaluations both for serial and parallel manipulators in terms of workspace, singularity, and stiffness.

3.1 Workspace evaluation

The workspace is one of the most important kinematic properties of manipulators, even by practical viewpoint because of its impact on manipulator design and location in a work cell.

The analytic mapping for the forward kinematics of a n-DOF manipulator with r-dimensional taskspace can be expressed in the form

$$k = \mathfrak{R}^n \longrightarrow \mathfrak{R}^r, p = k(\boldsymbol{q}) \tag{11}$$

The taskspace represents all feasible positions and orientations of the manipulator's end-effector. The taskspace can be conveniently represented in Cartesian coordinates and Euler angles.

A general numerical evaluation of the workspace can be deduced by formulating a suitable binary representation of a cross-section in the taskspace. A cross-section can be obtained with a suitable scan of the computed reachable positions and orientations p, once the forward kinematic problem has been solved to give p as function of the kinematic input joint variables q. A binary matrix $\boldsymbol{P}$ can be defined in a cross-section plane for a cross-section of the workspace as follows: if the (i, j) grid pixel includes a reachable point, then $P_{ij} = 1$; otherwise $P_{ij} = 0$, as shown in Fig. 1.

For example, one can consider a cross-section at a given value of Z-coordinate, in which P_{ij} is an entry of matrix P and a box in the grid mesh is indicated again with P_{ij}. Then a point in the grid is indicated as P_{ij}, with i along X-axis and j along the Y-axis, that can be calculated by

$$i = \text{round}\left(\frac{x + \Delta x}{x}\right) \quad j = \text{round}\left(\frac{y + \Delta y}{y}\right) \tag{12}$$

in which the round operator gives i and j as the nearest integers to the indicated ratio values, respectively. The quantities Δx and Δy are the resolution parameters of

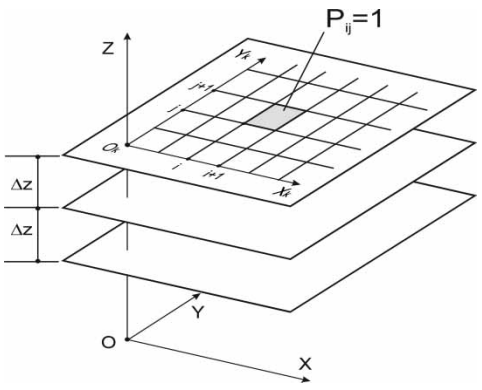


Fig. 1 A general scheme for a binary representation and evaluation of manipulator workspace

the grid mesh for the scanning process of the cross-section workspace region.

Therefore, a binary mapping for a workspace cross-section can be given as

$$P_{ij} = \begin{cases} 0 & \text{if } P_{ij} \notin W(H) \\ 1 & \text{if } P_{ij} \in W(H) \end{cases} \tag{13}$$

where $W(H)$ indicates workspace region; $\in$ stands for 'belonging to' and $\notin$ 'not belonging to'.

In addition, the proposed binary representation is useful for a numerical evaluation of the position workspace by computing the cross-section areas A_z as

$$A_z = \sum_{i=1}^{i_{\max}} \sum_{j=1}^{j_{\max}} \left(P_{ij} \Delta x \Delta y\right) \tag{14}$$

Finally, the workspace volume V can be computed considering n_z, which is the number of slices which have been considered for the workspace volume evaluation, according to the scheme of Fig. 1, as

$$V = \sum_{z=1}^{n_z} A_z \Delta_z \tag{15}$$

Similarly, the orientation workspace can be analysed by using a suitable binary representation with another binary matrix **P** for a workspace region that can be described in terms of orientation angles whose values can be considered in the reference axes of a grid mesh. Consequently, a numerical evaluation of orientation workspace can be carried out by using the formulation of equations (12) to (15) in order to identify a corresponding binary matrix and to compute the corresponding orientation performance measures cross-sections areas A_φ, and orientation workspace volume V_{or}, when a three-dimensional representation of the orientation capability is obtained by using three angular coordinates as Cartesian coordinates. It is worth to note that this formulation allows to include joint limits in the algorithm.

The above-mentioned numerical technique has been used as based on discretization for the workspace computation. It is general; it can be used both for serial and parallel manipulators; and it allows the use of the explicit kinematics for each robot under study. The workspace region is spanned and workspace points are determined and then the proposed computation can be performed to obtain a numerical evaluation of the workspace through the volumes and cross-section areas. A disadvantage of the proposed procedure can be recognized in the fact that, in order to get a sharp representation of the computed workspace, one has to use a high resolution for the grid mesh in the scanning process for the binary representation, which may increase the computational time and numerical load.

One can use equations (12) to (15) in order to evaluate any cross-section by properly adapting the formulation to the cross-section plane and intervals of a scanning process. Therefore, the optimum design problem with objective functions $f_1(\boldsymbol{X})$ and $f_2(\boldsymbol{X})$ can be formulated as finding the optimal design parameters values to obtain the position and orientation workspace volumes that are as close as possible to prescribed ones in the form

$$f_1(\boldsymbol{X}) = \left| 1 - \frac{V_{\text{pos}}}{V'_{\text{pos}}} \right| \tag{16}$$

$$f_2(\boldsymbol{X}) = \left| 1 - \frac{V_{\text{or}}}{V'_{\text{or}}} \right| \tag{17}$$

where $|.|$ is the absolute value; the subscripts pos and or indicate position and orientation, respectively; and prime refers to prescribed values. The workspace optimization problem can be also subject to design constraints such as

$$x_{\max} \leqslant x'_{\max}; \quad y_{\max} \leqslant y'_{\max}; \quad z_{\max} \leqslant z'_{\max} \tag{18}$$

$$\varphi_{\max} \leqslant \varphi'_{\max}; \quad \psi_{\max} \leqslant \psi'_{\max}; \quad \theta_{\max} \leqslant \theta'_{\max} \tag{19}$$

where the left-hand values correspond to the computed workspace volume and prime values describe the prescribed extreme reaches.

3.2 Singularity analysis

Design requirements and operation feasibility can also be focused conveniently on a free singularity condition. In fact, it is desirable to ensure a given workspace volume within which the manipulator extremity can be movable, controllable, and far enough from singularities. The singularity analysis both for serial and parallel manipulators can be performed by means of Jacobian matrices. The instantaneous relationship between the velocity in the taskspace and active joint velocity can be expressed as

$$\mathbf{J}(\boldsymbol{q}) : \Re^n \rightarrow \Re^r, \mathbf{B}\boldsymbol{t} = \mathbf{A}\dot{\boldsymbol{q}} \tag{20}$$

For serial manipulators **B** is the identity matrix, while for many parallel manipulators including the modified Gough–Stewart platform design, **A** is the identity matrix. For manipulator designs for which neither **A** nor **B** become the identity matrix, a single Jacobian can be defined as either $\mathbf{A}^{-1}\mathbf{B}$ or $\mathbf{B}^{-1}\mathbf{A}$ provided **A** or **B** is invertible, the difference being the direction from taskspace to joint space and vice versa, in which the Jacobian is defined. Indeed, equation (20) can be conveniently expressed, as

$$\boldsymbol{t} = \mathbf{J}\dot{\boldsymbol{q}} \tag{21}$$

Matrix **J** represents the analytic Jacobian of the manipulator at the configuration $\boldsymbol{q} \in \Re^n$ and it indicates how

infinitesimal changes of the configuration $\boldsymbol{q}$ translate into infinitesimal end-effector motions in the vicinity of $k(\boldsymbol{q})$. Vector $\dot{\boldsymbol{q}}$ represents the joint rates, and $\boldsymbol{t}$ is the twist array containing the linear velocity vector $\boldsymbol{v}$ and the angular velocity vector $\boldsymbol{\omega}$ of the moving platform.

In general, the conditions for identifying singular configurations can be represented by surfaces in the n-dimensional joint space and they can be obtained by vanishing the determinant of the Jacobian matrices (for square matrices).

In particular, if equation (20) is considered, matrix **A** gives the inverse kinematics singularities; and **B** gives the direct kinematics singularities. Direct kinematics singularities can be defined for parallel manipulators only, and they occur inside the workspace. In such configurations a parallel manipulator loses its rigidity, becoming locally movable, even if the actuated joints are locked.

The concept of singularity has been extensively studied and several classification methods have been defined. Manipulator singularities can be classified into three main groups [**4**]. The first type of singularity occurs when a manipulator reaches internal or external boundaries of its workspace and the output link loses one or more DOFs. This kind of singularity occurs both for serial and parallel manipulators. The second type of singularity is related to those configurations in which the output link is locally movable even if all the actuated joints are locked. The third type is related to linkage parameters and occurs when both the first and second types of singularities are involved. Both second and third kind of singularities can arise for parallel manipulators only.

Summarizing, parallel manipulator singularities arise whenever **A**, **B**, or both, become singular; serial manipulator singularities arise if **A** becomes singular.

Thus, a distinction can be made among three types of singularities, by considering equation (20), namely:

(a) the first type of singularity occurs when **A** becomes singular but **B** is invertible, being

$$\mathbf{A} \Rightarrow \text{ not full rank and } \mathbf{B} \Rightarrow \text{ full rank} \tag{22}$$

(b) the second type of singularity occurs only in closed kinematic chains and arises when **B** becomes singular but **A** is invertible, that is

$$\mathbf{A} \Rightarrow \text{ full rank and } \mathbf{B} \Rightarrow \text{ not full rank} \tag{23}$$

(c) the third type of singularity occurs when **A** and **B** are simultaneously singular, while none of the rows of **B** vanish.

Under a singularity of this type, configurations arise for which the movable plate can undergo finite motions even if the actuators are locked or, equivalently, it cannot resist forces or moments into one or more directions over a finite portion of the workspace, even if all actuators are locked. A finite motion can be very small but even very large to be considered sometimes as an extra DOF for specific manipulator configurations.

For the well-known Gough–Stewart platforms and many other parallel structures, each leg is connected to the moving platform through connecting points Q_i by means of spherical joints.

For some parallel mechanisms with less than six DOFs the condition based on an input–output velocity relation does not hold with suitable size of Jacobians since each leg has more freedoms and fewer constraints than the platform. Therefore, only a combination of the legs constraints can give results for a desired mobility of the platform, as pointed out in reference [**39**]. Those corresponding singularities can be classified as belonging to the second type, i.e. as related to a configuration with increased instantaneous mobility of the mechanism, as indicated in reference [**40**]. Therefore, in addition to the above-mentioned classification, additional singularities should be considered as those that may arise from passive joints.

When $\$_{\text{eff},i}$ is the effective screw of the ith leg, $\$_{\text{act},i}$ is the actuated screw of the ith leg and $\dot{\boldsymbol{q}}$ is the array of joint velocities, the general expressions of matrices **A** and **B** for parallel mechanisms can be expressed as

$$\mathbf{A} = \left[\$_{\text{eff},i}^{\mathrm{T}}\right] \tag{24}$$

$$\mathbf{B} = \operatorname{diag}\left[\$_{\text{eff},i}^{\mathrm{T}}\$_{\text{act},i}\right] \tag{25}$$

If the particular case of the Gough–Stewart platforms is considered and $\boldsymbol{d}_i$ is the vector along the line between the centre points of the spherical pairs connecting the ith leg to the base and the platform, then it holds

$$\$_{\text{eff},i}^{\mathrm{T}} = \left[\boldsymbol{d}_i^{\mathrm{T}}, \left(\boldsymbol{b}_i \times \boldsymbol{d}_i\right)^{\mathrm{T}}\right]$$

$$\$_{\text{act},i}^{\mathrm{T}} = \left[\boldsymbol{d}_i^{\mathrm{T}}, 0^{\mathrm{T}}\right] \tag{26}$$

with $\boldsymbol{b}_i$ being the vector along the line between the centre points of the spherical pairs connecting the ith leg to the platform centre.

If serial manipulators are considered, **B** matrix is the identity matrix and the ith column of matrix **A** represents the ith joint screw $\$_i$ then it holds

$$\mathbf{A} = \$_i$$

$$\$_i^{\mathrm{T}} = \left[\boldsymbol{d}_i^{\mathrm{T}}, \left(\boldsymbol{b}_i \times \boldsymbol{d}_i\right)^{\mathrm{T}}\right] \tag{27}$$

where $\boldsymbol{d}_i$ is the vector along the joint axis and $\boldsymbol{b}_i$ is a vector from the reference point to a point on such an axis.

Therefore, in a singularity analysis, if a serial manipulator is considered, then matrix **A** should be considered as in equation (27), since **B** is the identity matrix. For many parallel manipulators, including the modified Gough–Stewart platform design, **A** is the identity matrix, and therefore matrix **B** will be considered as in equations (25) and (26). For manipulator designs for which neither **A** nor **B** become the identity matrix, a single Jacobian can be defined as in equation (21).

Because of the above-mentioned expressions, the Jacobian matrix is pose dependent and non-isotropic. Consequently, performances such as rigidity, velocities, and forces, which can be expressed as functions of the Jacobian are pose dependent and therefore, it is important to consider the Jacobian in a rational design procedure, also because of those influences. Indeed, one can propose an objective function f_3 that can be deduced by analysing the analytical expression of the determinant of matrix **J** in the form

$$f_3 = \frac{\min \left|\det (\mathbf{J})\right|}{\left|\det (\mathbf{J})\right|_0} \tag{28}$$

with the condition

$$\det (\mathbf{J}) \neq 0 \tag{29}$$

that can take into account somehow all the situations in a singularity analysis, when the initial guess value $\mathbf{J}_0$ is considered to let f_3 be a dimensional. Furthermore, since Jacobian matrices are pose dependent, the minimum value has been considered convenient to obtain a single value function f_3.

3.3 Stiffness evaluation

Stiffness and accuracy of a robotic manipulator are strongly related to each other since positioning and orientating errors are because of compliant displacements and clearances as well as to control, construction, and assembling errors. The last errors can be evaluated by a kinematic analysis (calibration) by considering uncertainties in the kinematic parameters because of tolerances of construction and assembling of a robotic manipulator mechanism.

The stiffness properties of a manipulator can be defined through a matrix that is called 'Cartesian stiffness matrix **K**', [**41**–**44**]. This matrix gives the relation between the vector of the compliant displacements $\Delta \boldsymbol{S} = (S_x, S_y, S_z, S_\varphi, S_\psi, S_\theta)$ occurring at the movable platform when a static wrench $\boldsymbol{W} = (F_x, F_y, F_z, T_x, T_y, T_z)$ acts upon it, and $\boldsymbol{W}$ itself in the form

$$\mathbf{K}(\boldsymbol{q}) : \Re^6 \rightarrow \Re^6, \quad \boldsymbol{W} = K \Delta \boldsymbol{S} \tag{30}$$

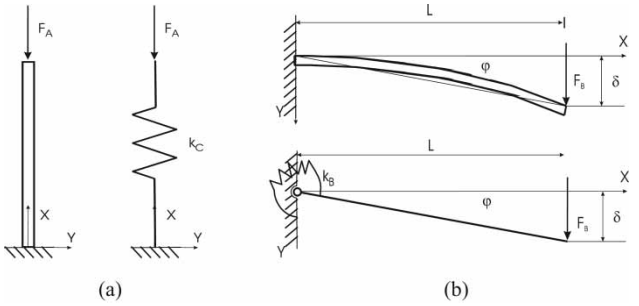


Fig. 2 Models for stiffness evaluation of a planar beam: (a) the case with axial load $\boldsymbol{F}_\mathrm{A}$ acting along X-axis and its model with lumped stiffness parameter k_C; (b) the case with a bending force $\boldsymbol{F}_\mathrm{B}$ and its model with lumped stiffness parameter k_B.

The stiffness matrix can be numerically computed by defining a suitable model of the manipulator, which takes into account lumped stiffness parameters of links and motors.

The proposed stiffness models with lumped parameters can take into account both the compliance of actuators and links along and about $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$ directions. They are based on the assumption of small compliant displacements. Under this assumption the superposition principle holds. Thus, the compliance of each link and actuator can be considered as an additive term to the overall compliance. Moreover, also the effects of tension/compression, bending and torsion stiffness of a link can be considered as an additive term to the stiffness of the link itself. These additive terms can be defined as lumped parameters and they can be represented as linear or torsion springs. For example, a planar beam under an axial load along X-axis can be represented with lumped stiffness parameter as shown in Fig. 2(a). In the model of Fig. 2(a) the symbol k_C is the lumped stiffness parameter of the compression/tension stiffness of the beam that is represented by a linear spring. Similarly, a scheme of a planar beam with a bending force $\boldsymbol{F}_\mathrm{B}$ can be represented as in Fig. 2(b). In the model of Fig. 2(b) the symbol k_B is the lumped stiffness parameter of the bending stiffness of the beam. Thus, a torsional spring will represent effects of torsion and bending.

Therefore, the stiffness matrix of a generic beam element can be written as reported for example in reference [**45**]

$$\mathbf{K} = \begin{bmatrix} EA/L & 0 & 0 \\ 0 & 12EI_Z/L^3 & 0 \\ 0 & 0 & 12EI_Y/L^3 \\ 0 & 0 & 0 \\ 0 & 0 & 6EI_Y/L^2 \\ 0 & -6EI_Z/L^2 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -6EI_Z/L^2 \\ 0 & 6EI_Y/L^2 & 0 \\ GJ/L & 0 & 0 \\ 0 & 4EI_Y/L & 0 \\ 0 & 0 & 4EI_Z/L \end{bmatrix} \tag{31}$$

where I_Y and I_Z are the two principal moments of inertia of the cross-sections, J is the equivalent torsional moment of inertia, L is the beam length, $\boldsymbol{A}$ is the cross-section area, E is the Young module, and G is the shear modulus. The stiffness matrix in equation (31) takes into account the effects of tension/compression, bending and torsion for a generic beam element. The matrix in equation (31) refers to a general model for a link with three linear springs and three torsional springs. Similar matrices can be derived for links having more complex shapes, even by means of finite-element analysis. It is also worth to note that some entries of the stiffness matrix in equation (31) can be considered as negligible with respect to others. Thus, simplified stiffness models with lumped parameters can be defined by considering only the non-negligible terms and therefore, corresponding linear and torsional springs can be identified as related to the non-zero entries only. In addition, the stiffness of the actuators can be taken into account in a similar manner and they can be represented as linear or torsion springs too. Indeed the superposition principle can be applied by combining contributions into single springs. Thus, each spring can model compliance both of links and actuators.

It is worth noting that the wrench acting on a manipulator link is a vector whose components are the reactional forces and torques that are transmitted to the link by the pairing elements connecting it with the neighbouring links.

A 6×6 stiffness matrix $\mathbf{K}$ of a manipulator can be derived through the composition of suitable matrices.

A first matrix C_{F} gives the relationship between an array vector of all the wrenches acting on each link a manipulator when a wrench $\boldsymbol{W}$ acts on the manipulator extremity according to the expression

$$\boldsymbol{W} = \mathbf{C}_{\mathrm{F}} \boldsymbol{W}_L \tag{32}$$

The matrix $\mathbf{C}_{\mathrm{F}}$ represents the force transmission capability of the manipulator mechanism and it has a size $6 \times N$ with N being the number of links of a manipulator. $\boldsymbol{W}_L$ is an array having size $N \times 1$ and can be written in a simplified form as an array of motor torques when the only source of compliance is assumed to be given by motors.

The second matrix $\mathbf{K}_{\mathrm{p}}$ gives the possibility to compute the vector $\Delta\boldsymbol{v}$ of all the deformations of the links when each wrench $\mathbf{W}_{Li}$ on a ith link given by $\boldsymbol{W}_L$ acts on the legs according to

$$\boldsymbol{W}_L = \mathbf{K}_{\mathrm{p}} \Delta\boldsymbol{v} \tag{33}$$

The matrix $\mathbf{K}_p$ groups the values of the lumped stiffness parameters for the deformable components of a manipulator structure and has a size $N \times N$. The vector $\Delta\boldsymbol{v}$ can be written in a simplified form as an array of joint angular displacements when the only source of compliance is assumed to be given by motors.

A third matrix $\mathbf{C}_{\mathrm{K}}$ gives the vector $\Delta\boldsymbol{S}$ of compliant displacements of the manipulator extremity due to the displacements of the manipulator links, as expressed in the form

$$\Delta\boldsymbol{v} = \mathbf{C}_{\mathrm{K}} \Delta\boldsymbol{S} \tag{34}$$

The matrix $\mathbf{C}_{\mathrm{K}}$ expresses the kinematics of a manipulator and has a size $N \times 6$.

Therefore, the stiffness matrix $\mathbf{K}$ can be computed as

$$\mathbf{K} = \mathbf{C}_{\mathrm{F}} \mathbf{K}_{\mathrm{p}} \mathbf{C}_{\mathrm{K}} \tag{35}$$

The stiffness matrix $\mathbf{K}$ can also be used to compute accuracy performance. In fact, the vector of compliant displacements $\Delta\boldsymbol{S} = [\Delta\boldsymbol{U}, \Delta\boldsymbol{Y}]^{\mathrm{T}}$ can be computed with its translating component $\Delta\boldsymbol{U}$ and rotational component $\Delta\boldsymbol{Y}$ by using equation (30) once the matrix $\mathbf{K}$ is determined when a static wrench acting on the movable platform is given.

From the above-mentioned considerations two objective functions that take into account stiffness performance can be defined as

$$f_4(\boldsymbol{X}) = \left| 1 - \frac{|\Delta\boldsymbol{U}_{\mathrm{d}}|}{|\Delta\boldsymbol{U}_{\mathrm{g}}|} \right| \tag{36}$$

$$f_5(\boldsymbol{X}) = \left| 1 - \frac{|\Delta\boldsymbol{Y}_{\mathrm{d}}|}{|\Delta\boldsymbol{Y}_{\mathrm{g}}|} \right| \tag{37}$$

where $|\,|$ is the operator for obtaining positive absolute values; $\Delta\boldsymbol{U}_{\mathrm{d}}$ and $\Delta\boldsymbol{U}_{\mathrm{g}}$ are maximum compliant displacements along X, Y, and Z-axes; $\Delta\boldsymbol{Y}_{\mathrm{d}}$ and $\Delta\boldsymbol{Y}_{\mathrm{g}}$ are vectors whose components are the maximum compliant rotations φ, θ, and ψ about X, Y, and Z-axes, respectively; d and g subscripts stand for design and given values, respectively.

Criteria f_4 and f_5 of equations (36) and (37) can be considered separately or in a single objective function component, according to specific requirements. However, because of the definition in equation (30) this formulation needs the condition

$$\det \mathbf{K} \neq 0 \tag{38}$$

that can be used as additional constraint.

4 NUMERICAL EXAMPLES

The practical feasibility of the proposed optimum design procedure has been experienced at LARM with several design tests that refer also to prototypes of manipulator architectures for robotic systems available at LARM [**46**]. In the following, two specific examples are discussed by illustrating numerical results that clarify the practical feasibility of the proposed design procedure as applied to the cases of a 6R PUMA-like robot and a Cassino parallel manipulator (CaPaMan) design. For these examples the numerical analyses are differentiated but the numerical evaluations of optimality criteria are computed with same formulations that are proposed in the paper.

There are a number of alternative methods to solve numerically a multi-objective optimization problem as reported in reference [**34**]. The specific formulated multi-objective problems has been solved by using a min–max algorithm through a code that is available within the commercial Matlab optimization toolbox [**33**]. At each iteration the largest objective function is minimized and the computation is terminated when all the objective functions have converged to minimum values. Those minima can be understood as local minima since in general there are no conditions in the solving techniques that may ensure to reach global minima.

The used min–max code of the Matlab optimization toolbox, [**33**], is based on sequential quadratic programming (SQP) technique. One can search the minimum value by using SQP that is successfully used for solving optimization problems with non-linear objective functions and constraints as functions of several variables. This numerical procedure works in such a way that at each step k a solution is found along a search direction δ with variable update Ψk. The iteration continues until the vector of variables converges. The numerical procedure has been developed so that the formulation for the manipulator design has been easily included within the solving procedure for the optimization problem by using the facilities of the optimization toolbox of Matlab [**33**], which permits an easy arrangement for an optimum design with analytical expressions.

The herein used numerical procedure for the solution of a multi-objective optimization design problem is outlined in the flowchart of Fig. 3.

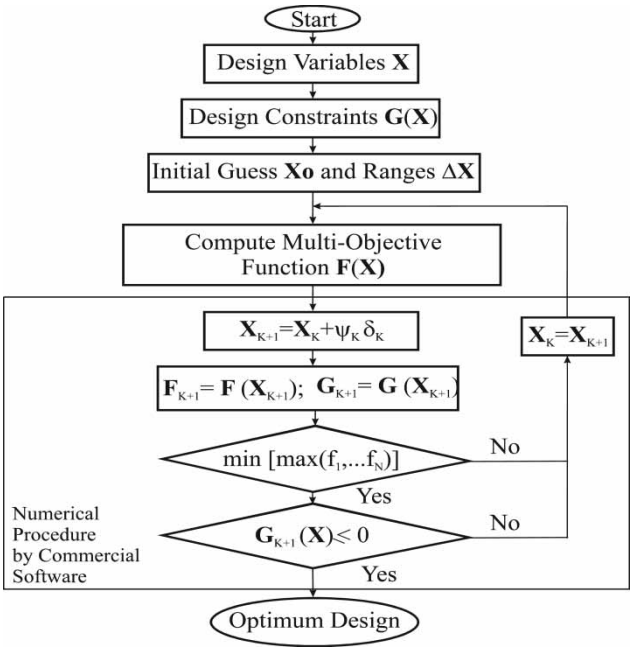


Fig. 3 A flowchart for optimum design procedure both for serial and parallel manipulators

Table 1 Design parameters for the optimum designed PUMA-like manipulator of Figs 4 to 7

Values	a_2 (mm)	a_3 (mm)	d_3 (mm)	d_4 (mm)
Initial guess	431.8	20.3	125.4	431.8
Optimal	1000.0	100.0	269.6	1000.0

4.1 A 6R serial manipulator

A six DOFs PUMA-like manipulator has been considered to test the engineering feasibility of the above-mentioned formulation for optimum design of manipulators as specifically applied to serial architectures. Main design parameters for a PUMA-like manipulator are shown in Fig. 4(a). Values of the design parameters a_2, a_3, d_3, and d_4 are reported in Table 1. The initial guess values have been defined by referring to a PUMA 562 design and the mobility ranges for the joint angles have been assumed equal to 180° for the first three joints (of the arm) and 90° for the last three joints (of the wrist).

A simplified stiffness model for a PUMA-like manipulator is shown in Fig. 4(b). In this model, the links have been considered as rigid bodies. In fact, in this type of robots the payloads are limited and the compliant displacements are in general because of the flexibility of joints only. The link compliant displacements are much smaller than the compliant displacements because of the compliance of motors as pointed out for example in reference [**43**]. Thus, in the model of Fig. 4(b) the lumped parameters k_{T1} to k_{T6} take into account the stiffness motors and joints only.

Moreover, if the only contributions to the overall compliance are given by motor compliances, the stiffness matrix **K** can be computed through equation (29) with

$$\mathbf{C}_F = \mathbf{J}^{-T}; \quad \mathbf{C}_K = \mathbf{J}^{-1} \tag{39}$$

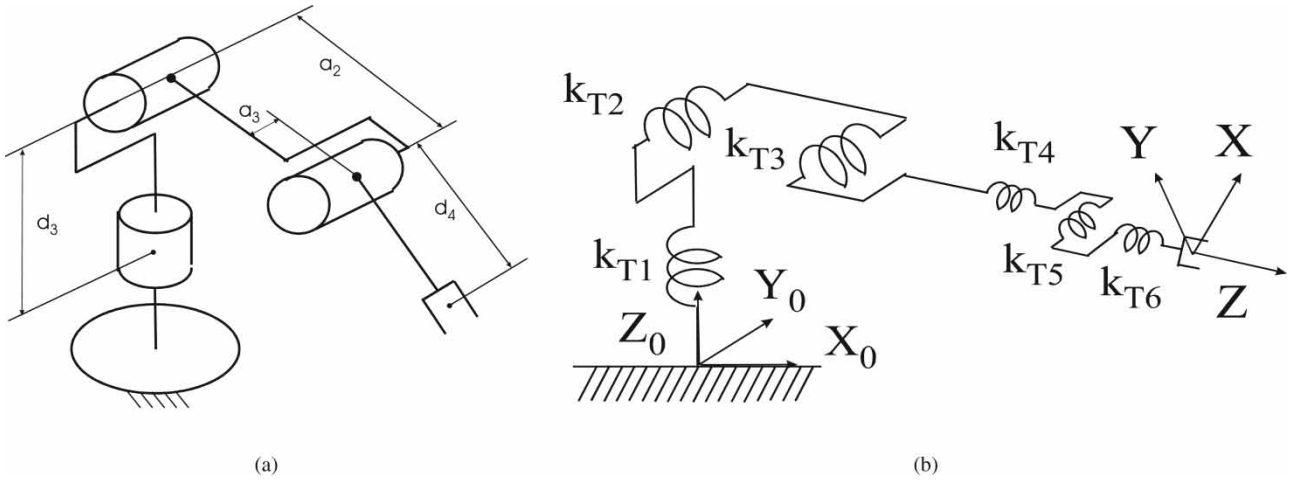


Fig. 4 Main design parameters for a PUMA-like manipulator: (a) dimensional parameters; (b) stiffness springs

where **J** is the Jacobian matrix of the PUMA-like robot. The matrix $\mathbf{K}_{\mathrm{P}}$ in equation (25) can be computed as a diagonal matrix with the lumped stiffness parameters of the motors given as $k_{\mathrm{T1}} = k_{\mathrm{T2}} = k_{\mathrm{T3}} = 4.67 \times 10^{10}$ a_i N/m and $k_{\mathrm{T3}} = k_{\mathrm{T4}} = k_{\mathrm{T5}} = 4.67 \times 10^{7}$ N/m. The stiffness matrix of the PUMA-like robot that can be computed through equations (35) and (39) as function of the input joint angles. The expressions of the input joint angles can be computed as functions the coordinates $(x, y, z, \phi, \psi, \theta)$ for the position and orientation of the end-effector from the well-known inverse kinematics of PUMA-like robot.

A singularity condition can be obtained by using equation (27). Position and orientation workspace volumes can be conveniently evaluated by using the well-known closed-form kinematics of PUMA manipulators and then the proposed algorithm with binary representation. Workspace constraints and prescriptions of equations (18) and (19) have been specifically formulated in the form

$$\begin{aligned} G(1) &= \mathrm{Max}(\Delta x) - 1.2\mathrm{Max}(\Delta x)_{\mathrm{p}} \\ G(2) &= \mathrm{Max}(\Delta y) - 1.2\mathrm{Max}(\Delta y)_{\mathrm{p}} \\ G(3) &= \mathrm{Max}(\Delta z) - 1.2\mathrm{Max}(\Delta z)_{\mathrm{p}} \\ G(4) &= \mathrm{Max}(\Delta \varphi) - 1.2\mathrm{Max}(\Delta \varphi)_{\mathrm{p}} \\ G(5) &= \mathrm{Max}(\Delta \psi) - 1.2\mathrm{Max}(\Delta \psi)_{\mathrm{p}} \\ G(6) &= \mathrm{Max}(\Delta \theta) - 1.2\mathrm{Max}(\Delta \theta)_{\mathrm{p}} \end{aligned} \tag{40}$$

in which 'Max' is the operator giving the maximum of a function. The constraint functions $G(1)$ to $G(6)$ express limits for the maximum components of the compliant displacements in an optimal solution. The maximum is prescribed to be no more than 20 per cent bigger than a given design value. In particular, $\mathrm{Max}(\Delta x)$, $\mathrm{Max}(\Delta y)$, $\mathrm{Max}(\Delta z)$, $\mathrm{Max}(\Delta \varphi)$, $\mathrm{Max}(\Delta \psi)$, and $\mathrm{Max}(\Delta \theta)$ are the maximum computed values of the compliant displacements along and about X, Y, and Z directions, respectively. $\mathrm{Max}(\Delta x)_{\mathrm{p}}$, $\mathrm{Max}(\Delta y)_{\mathrm{p}}$, $\mathrm{Max}(\Delta z)_{\mathrm{p}}$, $\mathrm{Max}(\Delta \varphi)_p$, $\mathrm{Max}(\Delta \psi)_{\mathrm{p}}$, and $\mathrm{Max}(\Delta \theta)_{\mathrm{p}}$ are the prescribed maximum values of compliant displacements along and about X, Y, Z directions, respectively, as they can be chosen by a designer/user.

Results of the proposed design procedure as applied to the PUMA-like architecture are reported in Tables 1 and 2 and Figs 5–7.

The numerical example for a PUMA-like manipulator has been elaborated in a Intel Pentium M 2.00 GHz with 1 GB of RAM. The algorithm takes 100 iterations

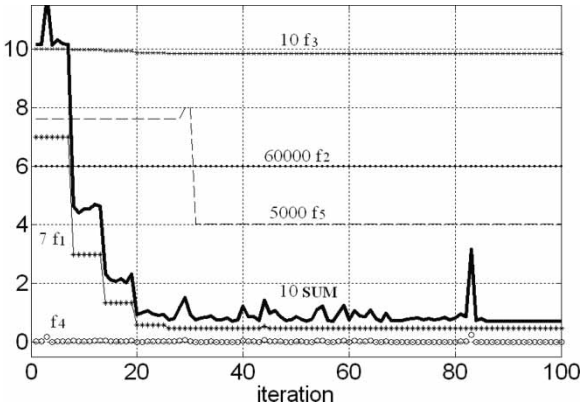


Fig. 5 Evolution of the objective functions ***F*** versus number of iterations for the example of PUMA-like robot optimal design of Fig. 4 and Table 1: position workspace volume as f_1; orientation workspace volume as f_2; singularity condition as f_3; compliant displacements and rotations as f_4 and f_5; SUM is a sum of the objective functions. (numbers before f_1 to f_5 are magnification factors that have been used to spread out the various plot lines)

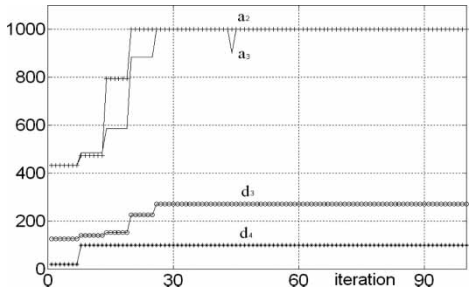


Fig. 6 Evolution of design parameters versus number of iterations for the example of PUMA-like robot optimal design of Fig. 4 and Table 1

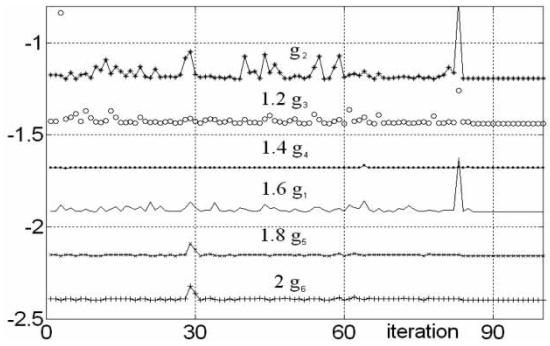


Fig. 7 Evolution of design constraints versus number of iterations for the example of PUMA-like robot optimal design of Fig. 4 and Table 1 (numbers before g_1 to g_6 are magnification factors that have been used to spread out the various plot lines)

to converge to an optimal solution with a computation time of 25 s. The accuracy for the objective function evaluations has been set equal to 1e-5 and the accuracy for the design parameters has been set equal to 1e-5.

The numbers in Figs 5 and 7 are magnification factors that have been used to spread out the various plot lines of each function in the graphs.

In particular, Fig. 5 shows the evolution of the objective functions versus number of iterations. The symbol SUM indicates a scalar value that has been computed for representation purposes by summing up the values of all the objective functions without any weighting factor. It has been useful as a synthetic measure of $\boldsymbol{F}(\boldsymbol{X})$ for a better visualization of the evolution to an optimal solution. The sum (SUM) of all the objective functions converges to a value that is about 10 per cent of the initial value after 100 iterations approximately. Figure 6 shows the evolution of the design parameters versus the number of iterations. Figure 7 shows the evolution of the design constraints versus the number of iterations. Table 1 shows the initial guess and optimal values of design parameters for the PUMA-like manipulator. Table 2 shows the main characteristics of the PUMA-like manipulator for the initial guess and the optimal values of design parameters that are reported in Table 1.

4.2 A 3 DOF parallel manipulator

The CaPaMan manipulator has been considered to test the engineering feasibility of the above-mentioned formulation for optimum design of manipulators as specifically applied to parallel architectures which can be different from a general Gough–Stewart platform. CaPaMan architecture has been conceived at LARM in Cassino since 1996, where a prototype has been built for experimental activity. Indeed, by using the existing prototype, simulations have been carried out also to validate the proposed optimum design by considering several guess solutions and imposing workspace and stiffness characteristics of the built prototype. According to those satisfactory results a numerical example has been proposed to obtain the same workspace characteristics but with enhanced stiffness and conditions for avoiding singularities.

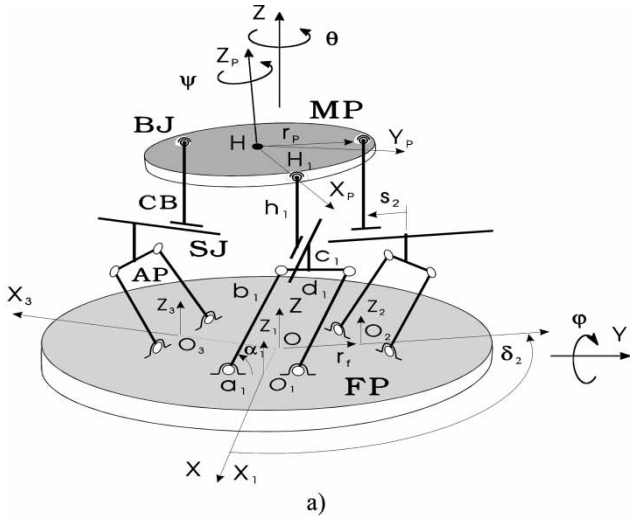
A schematic representation of the CaPaMan manipulator is shown in Fig. 8(a), and the prototype is shown in Fig. 8(b).

Position and orientation workspace volumes can be conveniently evaluated by using the algebraic formulation for the kinematics of the CaPaMan manipulator. Singularity analysis for CaPaMan manipulator has been reported in reference [**47**] and matrices **A** and **B** have been formulated in a form that is useful also for the proposed optimality criterion as

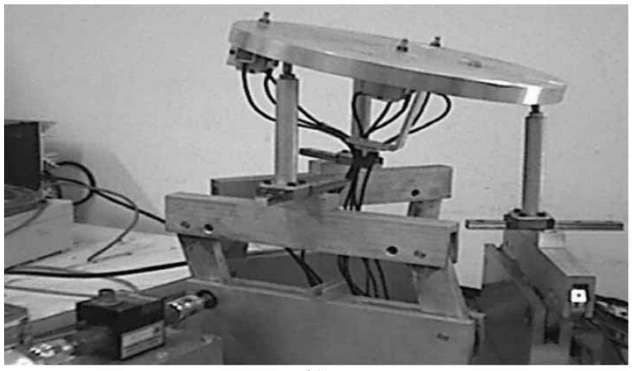
$$\mathbf{A} = \begin{bmatrix} (D-F)\, b_1 c\alpha_1 & (D+2F)\, b_2 c\alpha_2 - (2D+F)\, b_3 c\alpha_3 \\ (D+F)\, b_1 c\alpha_1 & -D b_2 c\alpha_2 \qquad -F b_3 c\alpha_3 \\ b_1 c\alpha_1 & b_2 c\alpha_2 \qquad b_3 c\alpha_3 \end{bmatrix} \tag{41}$$

Table 2 Design characteristics of optimum solution for PUMA-like manipulator of Table 1

Values of workspace ranges	Δx (mm)	Δy (mm)	Δz (mm)	$\Delta\phi(^\circ)$	$\Delta\psi(^\circ)$	$\Delta\theta(^\circ)$
Initial guess	529.2	472.4	625.0	180	180	180
Optimal	1305	1200	1485	180	180	180
Values of compliant displacements	$\boldsymbol{U}x$ (mm)	$\boldsymbol{U}y$ (mm)	$\boldsymbol{U}z$ (mm)	$\boldsymbol{Y}_\phi(^\circ)$	$\boldsymbol{Y}\psi(^\circ)$	$\boldsymbol{Y}\theta(^\circ)$
Initial guess	1.70	1.30	1.40	0.01	0.08	0.01
Optimal	1.50	1.90	1.60	0.01	0.11	0.01



a)



b)

Fig. 8 CaPaMan design: (a) a kinematic diagram; (b) a built prototype at LARM

$$\mathbf{B} = \begin{bmatrix} \left(6E/\sqrt{3}\right) & 0 & 0 \\ 0 & \sqrt{E\left(9r_p^2 - 4E\right)} & 0 \\ 0 & 0 & 3 \end{bmatrix} \tag{42}$$

in which

$$\begin{aligned} E &= z_1^2 + z_2^2 + z_3^2 - z_1 z_2 - z_2 z_3 - z_1 Z_3 \\ D &= 2Z_2 - z_1 - z_3, \quad F = 2Z_3 - z_1 - z_2 \end{aligned} \tag{43}$$

with

$$z_k = b_k \sin \alpha_k \quad \text{for } k = 1, 2, 3 \tag{44}$$

Stiffness analysis of CaPaMan has been reported in reference [**25**]. By modelling each leg of CaPaMan as in Fig. 9, the stiffness matrix of CaPaMan can be derived as

$$\mathbf{K}_{\text{CaPaMan}} = \mathbf{C}_{\text{F}} \mathbf{K}_{\text{P}} \mathbf{C}_{\text{K}} \tag{45}$$

with

$$\mathbf{C}_{\text{F}} = \mathbf{M}_{\text{FN}}; \quad \mathbf{C}_{\text{K}} = \mathbf{C}_{\text{P}}^{-1} \mathbf{A}_{\text{d}}^{-1} \tag{46}$$

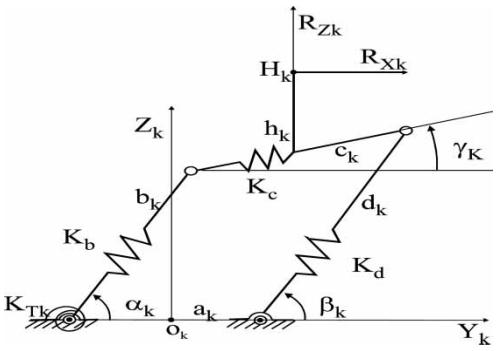


Fig. 9 A scheme for stiffness evaluation of a CaPaMan leg

where $\mathbf{M}_{\text{FN}}$ is a 6×6 transmission matrix for the static wrench applied on H and transmitted to points H_1, H_2, and H_3 of each leg; $\mathbf{K}_{\text{p}}$ is a 6×6 matrix with the lumped stiffness parameters of the three legs; $\mathbf{C}_{\text{p}}$ is a 6×6 matrix giving the displacements of the links of each leg as a function of the displacements of points H_1, H_2, and H_2; $\mathbf{A}_{\text{d}}$ is a 6×6 matrix that has been obtained by using the direct kinematics of the CaPaMan to give the position of point H on the movable plate as function of the position of points H_1, H_2, and H_2 in the form

$$\boldsymbol{X}_{\text{H}} = \mathbf{A}_{\text{d}} \boldsymbol{\nu} \tag{47}$$

with $\boldsymbol{\nu} = [y_1, z_1, y_2, z_2, y_3, \Delta z_3]^{\text{T}}$ and $\boldsymbol{X}_{\text{H}} = [x_{\text{H}}, y_{\text{H}}, z_{\text{H}}, \varphi, \theta, \psi]^{\text{T}}$. The derivation of matrices $\mathbf{M}_{\text{FN}}$, $\mathbf{K}_{\text{p}}$, $\mathbf{A}_{\text{d}}$, and $\mathbf{C}_{\text{p}}$ for CaPaMan can be found in reference [**25**].

The lumped stiffness parameters has been assumed as $k_{\text{bk}} = k_{\text{dk}} = 2.625 \times 10^6$ N/m and $k_{\text{Tk}} = 58.4 \times 10^3$ Nm/rad; the couplers c_{k} have been assumed rigid bodies because of the massive design that has been imposed to have a fix position of the sliding joints. Further details on the derivation of the matrices in equations (45) and (46) can be found in reference [**25**]. In the numerical example, for evaluation and design purposes assumed that $r_{\text{p}} = r_{\text{f}}$, $a_{\text{k}} = c_{\text{k}}$, $b_{\text{k}} = d_{\text{k}}$.

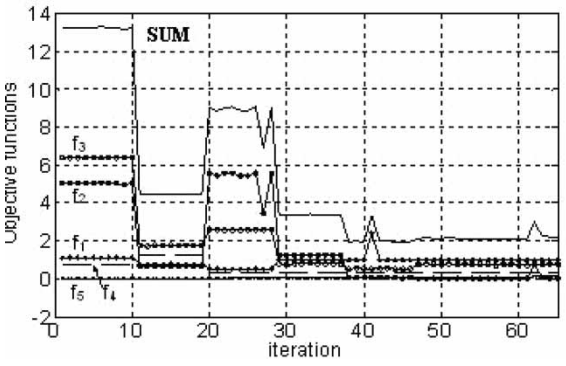


Fig. 10 Evolution of the objective functions versus number of iterations for the example of CaPaMan optimal design of Fig. 8 and Table 1

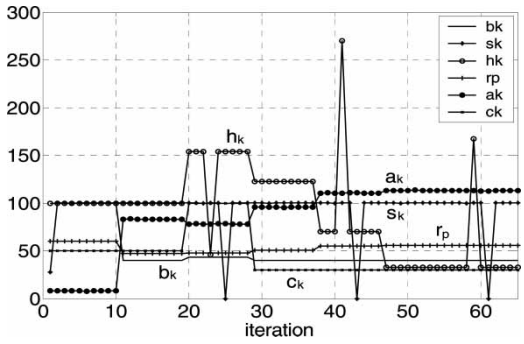


Fig. 11 Evolution of design parameters versus number of iterations for the example of CaPaMan optimal design of Fig. 8 and Table 1

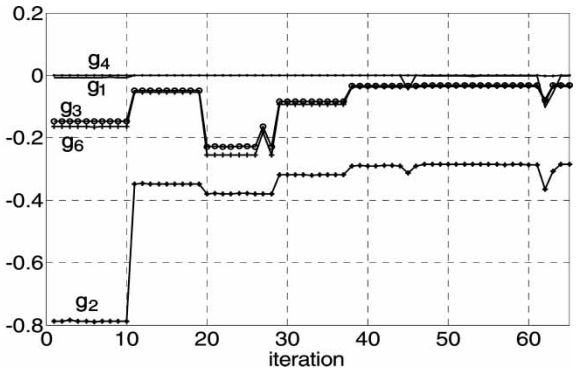


Fig. 12 Evolution of design constraint versus number of iterations for the example of CaPaMan optimal design of Fig. 8 and Table 1

Workspace constraints and prescriptions have been formulated like equation (40).

Results of the proposed design procedure as applied to the CAPAMAN architecture are reported in Figs 10 to 12 and Tables 3 and 4.

In particular, the evolution of the objective functions is reported in Fig. 10, form which one can note that the numerical procedure takes 65 iterations to converge to the optimum values that are reported in Table 3. Evolution of design parameters and constraints are shown in Figs 11 and 12. Design characteristics for the optimum solution are reported in Table 4.

For the proposed numerical example, the inverse kinematic singularities related to matrix **A** in equation (41) gives the condition that input crank angle α_i should be different from 90°, (for $i = 1, 2, 3$). This condition and direct kinematic singularities have been taken into account in the numerical procedure through the constraint that is expressed by equation (29).

Table 3 Design parameters for optimal CaPaMan design of Figs 8 to 12

Values	a_k (mm)	b_k (mm)	h_k (mm)	r_p (mm)	α_k (°)	s_k (mm)
Initial guess	27.85	100.0	100.0	60.0	45; 135	50.0
Optimal	113.1	40.0	32.9	55.8	45; 112	30.0

The numerical example for the CaPaMan manipulator has been elaborated in a Intel Pentium M 2.00 GHz with 1 GB of RAM. The algorithm takes 65 iterations to converge to an optimal solution with a computation time of 4 min and 8 s. The accuracy for the objective function evaluations has been set equal to 1e-5 and the accuracy for the design parameters has been set equal to 1e-3.

Numerical examples show satisfactory procedure with a quite rapid convergence to a feasible optimal solution. The robustness of the design algorithm is proved in some extent even by the relatively large distance of the computed optimal design solutions from the guess values in the reported examples, where as optimization algorithms often work properly when a guess case is near to an optimum solution. Nevertheless, likewise any numerical technique for solving optimization problems, there is no possibility to be sure in reaching a global optimum solution. However, the reported examples show very satisfactory results for the implementation of the proposed design procedure both in terms of reached optimality and numerical efficiency.

5 CONCLUSION

The novel contribution of the paper can be considered in the attempt to attach the design problem both for serial and parallel manipulators through an unique formulation using basic optimality criteria with common computations. In addition, practical use of the above-mentioned optimality criteria has suggested a

Table 4 Design characteristics of optimum solution for optimal CaPaMan design of Figs 7 to 11 and Table 3

Values of workspace ranges	Δx (mm)	Δy (mm)	Δz (mm)	$\Delta\varphi$ (°)	$\Delta\psi$ (°)	$\Delta\theta$ (°)
Initial guess	105.8	112.4	29.3	38.0	179.9	321.8
Optimal	48.6	55.9	11.7	16.1	179.9	212.4
Values of compliant displacements	$\boldsymbol{U}x$ (mm)	$\boldsymbol{U}y$ (mm)	$\boldsymbol{U}z$ (mm)	$Y\phi$ (°)	$Y\psi$ (°)	$Y\theta$ (°)
Initial guess	5.5×10^{-4}	6.7×10^{-6}	3.2×10^{-4}	2.4×10^{-5}	2.4×10^{-5}	2.3×10^{-9}
Optimal	0.002	1.6×10^{-6}	0.001	6.0×10^{-4}	6.5×10^{-4}	2.3×10^{-8}

design formulation as a multi-objective optimization problem whose solution has been attempted by using a suitable algorithm from a technique that is available in a commercial software package.

In this paper, a multi-objective optimum design procedure is outlined for robots by discussing optimality criteria and numerical aspects. In particular, optimality criteria have been chosen according to the most common design requirements for robotic systems, which are the workspace, singularities, and stiffness behaviour, as regarding both to position and orientation capabilities. In fact, is of great interest to design a robot with desired workspace, free of singularities and with adequate stiffness. The latter two requirements are also related to robot accuracy and path planning. The choice of each design criterion has been made according to two aspects: computational efficiency and possibility to be used both for serial and parallel manipulators. Additional alternative objective functions can be used to extend the proposed design procedure to more general design problems. The feasibility of such a complex design formulation for robotic manipulators has been illustrated by referring to experiences developed at LARM in Cassino, when optimality criteria and numerical aspects have been formulated by taking into account the peculiarity and constraints both of serial and parallel manipulator architectures.

REFERENCES

1 **Shimano, B. E.** and **Roth B.** Dimensional synthesis of manipulators. 3rd CISM-FToMM Symposium on *Theory and practice of robots and manipulators*, 1980, pp. 166–187 (Elsevier, Amsterdam).

2 **Roth, B.** Analytical design of open chains. 3rd International Symposium on *Robotics research*, 1986, pp. 281–288 (MIT Press, New York).

3 **Vijaykumar, R., Waldron, K. J.,** and **Tsai, M. J.** Geometric optimization of serial chain manipulator structures for working volume and dexterit. *Int. J. Robot. Res.*, 1986, **5**(2), 91–103.

4 **Gosselin, C.** and **Angeles J.** The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator. *ASME, J. Mech. Transm. Autom. Des.*, 1988, **110**, 35–41.

5 **Paden, B.** and **Sastry, S.** Optimal kinematic design of 6R manipulators. *Int. J. Robot. Res.*, 1988, **7**(2), 43–61.

6 **Manoochehri, S.** and **Seireg, A. A.** A computer-based methodology for the form synthesis and optimal design of robot manipulators. *ASME, J. Mech. Des.*, 1990, **112**, 501–508.

7 **Gosselin, C. M.** The optimum design of robotic manipulators using dexterity indices. *Robot. Auton. Syst.*, 1992, **9**, 213–226.

8 **Park, F. C.** Optimal robot design and differential geometry. *Trans. ASME*, 1995, **117**, 87–92. http://robotics.snu.ac.kr/.

9 **Carretero, J. A., Podhorodeski, R. P., Nahon, M. A.,** and **Gosselin, C. M.** Kinematic analysis and optimization of a new three degree-of freedom spatial parallel manipulator. *ASME, J. Mech. Des.*, 2000, **122**(1), 17–24.

10 **Liu, X.-J., Jin, Z.-L.,** and **Gao, F.** Optimum design of 3-DOF spherical parallel manipulators with respect to the conditioning and stiffness indices. *Mech. Mach. Theory*, 2000, **35**(9), 1257–1267.

11 **Angeles, J.** The robust design of parallel manipulators. 1st International Colloquium Collaborative Research Center 562, Braunschweig, 2002, pp. 9–30.

12 **Merlet, J.-P.** Designing a parallel manipulator for a specific workspace. *Int. J. Robot. Res.*, 1997, **16**(4), 545–556.

13 **Schonherr, J.** Evaluation and optimum design of parallel manipulators having defined workspace. ASME 26th Biennial Mechanisms and Robotics Conference, Baltimore, 2000, paper DETC2000/MECH-14092.

14 **Bhattacharya, S., Hatwal, H.,** and **Ghosh, A.** On the optimum design of Stewart platform type parallel manipulators. *J. Robotica*, 1995, **13**, 133–140.

15 **Takeda, Y.** and **Funabashi, H.** Kinematic synthesis of in-parallel actuated mechanisms based on the global isotropy index. *J. Robot Mechatronics*, 1999, **11**(5), 404–410.

16 **Gosselin, C. M.** On the design of efficient parallel mechanisms. In Proceeding of the Advanced Study Institute of NATO: *Computational methods in mechanics* (Keynote Lecture), St. Konstantin and Elena, Bulgaria, 1997, vol. 1, pp. 157–186.

17 **Ottaviano, E.** and **Carbone, G.** A procedure for the multiobjective design of parallel manipulators. *Int. J. Mech. Control*, 2003, **4**(2), 57–62.

18 **Hao, F.** and **Merlet, J.-P.** Multi-criteria optimal design of parallel manipulators based on interval analysis. *Mech. Mach. Theory*, 2005, **40**, 157–171.

19 **Ceccarelli, M.** A synthesis algorithm for three-revolute manipulators by using an algebraic formulation of workspace boundary. *ASME, J. Mech. Des.*, 1995, **117**, 298–302.

20 **Ceccarelli, M.** A formulation for the workspace boundary of general n-revolute manipulators. *Mech. Mach. Theory*, 1996, **31**, 637–646.

21 **Ceccarelli, M.** An analytical design of telescopic manipulator arms for prescribed workspace. In *Advances in robot kinematics: analysis and control* (Eds J. Lenarcic and M. L. Husty), 1998, pp. 247–254 (Kluwer, Dordrecht).

22 **Ceccarelli, M.** Designing two-revolute manipulators for prescribed feasible workspace regions. *ASME, J. Mech. Des.*, 2002, **124**, 427–434.

23 **Ceccarelli, M.** and **Scaramuzza, G.** Analytical constraints for a workspace design of 2R manipulators. In *Computational kinematics '95* (Eds J. P. Merlet and B. Ravani), 1995, pp. 173–182 (Kluwer, Dordrecht).

24 **Ceccarelli, M.** An optimum design of parallel manipulators: formulation and experimental validation. 1st International Colloquium Collaborative Research Center 562, Braunschweig, 2002, pp. 47–63.

25 **Ceccarelli, M.** and **Carbone, G.** A stiffness analysis for CaPaMan (Cassino parallel manipulator). *Mech. Mach. Theory*, 2002, **37**(5), 427–439.

26 Mata, A. V. and **Ceccarelli, M.** Objective functions for the optimum design of robots. 1st Iberoamerican Conference on *Mechanical engineering*, Madrid, 1993, vol. 3, pp. 47–54.

27 Ceccarelli, M. Optimal design and location of manipulators. In *Computational dynamics in multibody systems* (Eds M. F. O. S. Pereira and J. A. C. Ambrosio), 1995, pp. 131–146 (Kluwer, Dordrecht).

28 Ottaviano, E. and **Ceccarelli, M.** Optimal design of CaPaMan (Cassino parallel manipulator) with prescribed workspace. IFToMM Electronic Journal Computational Kinematics EJCK, Computational Kinematics CK2001, Seoul, 2001, pp. 35–44, vol. 1(1), 2002, paper no. 04.

29 Ottaviano, E. and **Ceccarelli, M.** Optimum design of parallel manipulators for workspace and singularity performances. Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators, Quebec City, 2002, pp. 98–105.

30 Lanni, C., Saramago, S. F. P., and **Ceccarelli, M.** Optimal design of 3R manipulators by using classical techniques and simulated annealing. *J. Braz. Soc. Mech. Sci.*, 2002, **XXIV**, 294–302.

31 Ceccarelli, M. and **Lanni, C.** A multi-objective optimum design of general 3R manipulators for prescribed workspace limits. *Mech. Mach. Theory*, 2004, **39**(2), 119–132.

32 Ceccarelli, M. *Fundamentals of mechanics of robotic manipulation*, 2004 (Kluwer, Dordrecht).

33 Grace, A. *Optimization toolbox user's guide*, 1992 (The Matlab Works Inc.).

34 Vanderplaats, G. *Numerical optimization techniques for engineers design*, 1984 (McGraw-Hill, New York).

35 Eldar, Y. C. Minimax estimation of deterministic parameters in linear models with a random model matrix. *IEEE Trans. Signal Process.*, 2006, **54**(2), 601–612.

36 Pankov, A. R., Platonov, E. N., and **Siemenikhin, K. V.** On minimax identification: method of dual optimization. 39th IEEE Conference on *Decision and control*, Sydney, vol. 5, 2000, pp. 4759–4764.

37 Medanic, J. and **Andjelic, M.** Minimax solution of the multiple-target problem. *IEEE Trans. Autom. Control*, 1972, **17**(5), 597–604.

38 Merlet, J.-P. A C++ algorithms library of interval analysis for equation systems, version 2.3. The COPRIN project http://www-sop.inria.fr/coprin/equipe/merlet/merlet.html, 2004.

39 Di Gregorio, R. and **Parenti-Castelli, V.** A translational 3-DOF parallel manipulator. In *Advances in robot kinematics: analysis and control* (Eds J. Lenarcic and M. L. Husty), 1998, pp. 49–58 (Kluwer Academic Publishers, Dordrecht).

40 Zlatanov, D., Fenton, R. G., and **Benhabib, B.** Analysis of the instantaneous kinematics and singular configurations of hybrid-chain manipulators. ASME 23rd Biennial Mechanisms Conference, Minneapolis, 1994, DE-vol. 72, pp. 467–476.

41 Gosselin, C. Stiffness mapping for parallel manipulators. *IEEE Trans. Robot. Autom.*, 1990, **6**(3), 377–382.

42 Duffy, J. *Statics and kinematics with applications to robotics.* 1996, pp. 153–169 (Cambridge University Press).

43 Tsai, L. W. *Robot analysis: the mechanics of serial and parallel manipulators*, 1999, pp. 260–297 (Wiley, New York).

44 Rivin, E. I. *Stiffness and damping in mechanical design*, (Marcel Dekker, New York).

45 Kardestuncer, H. *Elementary matrix analysis of structures*, 1974 (McGraw-Hill Kogakusha, Tokyo).

46 LARM webpage http://webuser.unicas.it/weblarm/larmindex.htm, 2006.

47 Wolf, A., Ottaviano, E., Shoham, M. and **Ceccarelli, M.** Application of line geometry and linear complex approximation to singularity analysis of the 3-DOF CaPaMan parallel manipulator. *Mech. Mach. Theory*, 2004, **39**(1), 75–95.