Concurrent optimal design of modular robotic configuration

Concurrent Optimal Design of Modular Robotic Configuration

Z. M. Bi and W. J. Zhang*
Advanced Engineering Design Laboratory
Department of Mechanical Engineering
University of Saskatchewan
Saskatoon, S7N 5A9, SK, Canada

Received 25 March 2000; accepted 18 October 2000

This paper presents a new optimization design methodology that is applicable to modular systems. This new methodology is called concurrent optimization design method (CODM). A modular robot is taken as a case study. The CODM is superior to the existing methods for modular robot configuration design in the sense that traditional type synthesis and dimensional synthesis now can be treated once. This mathematically implies that (i) variables are defined for both types and dimensions, and (ii) all the variables are defined in one optimization problem formulation. This paper illustrates that, in fact, optimization design for modular architectures necessitates a multiobjective optimization problem. A genetic algorithm is used to solve for this complex optimization model which contains both discrete and continuous variables. © 2001 John Wiley & Sons, Inc.

1. INTRODUCTION

A modular robotic system consists of several modular links and modular joints with standardized connecting interfaces. It can be easily assembled and disassembled, so it can accomplish a larger number of classes of tasks through the reconfiguration of a small inventory of modules. This method adds an additional dimension to the flexibility of a robotic system. One of the issues to be addressed to make modular robot systems effective is to generate a task-oriented optimal modular robot, ideally with a computer program. There are some studies on this issue, in which the methodology for robots without modular architecture generally prevails. The observation made in this paper is that modular robot architecture has some distinct features, due to which a more effective method can be developed to generate a task-oriented optimal configuration of modular robot. This new method will be described.

*To whom all correspondence should be addressed; e-mail: chris_zhang@engr.usask.ca.

2. RELATED WORK

The concept of and techniques for modular robots have been of interest in the robotics field since the 1980s. A lot of modular robotic prototypes have been developed since then. Based on the architectures of existing modular robotic systems, this study recognizes the three levels of modular robot architecture in terms of how the system flexibility is achieved:

1. **Module-level reconfigurable modular robotic system:** Its flexibility is achieved by adjusting parameters in one module or selecting an alternative module for a different parameter.
2. **Assembly-level reconfigurable modular robotic system:** Its flexibility is achieved by permutation of modules or selection of assembly ports for a module.
3. **Configuration-level reconfigurable modular robotic system:** Its flexibility is achieved by varying different configurations where the selection of types of modules and their connections is of a concern.

The three levels of modular architecture are useful to classify the existing work as follows. There are many studies on kinematics, dynamic, and control design of modular robots. Paredis and Khosla[1,2] described a general flow chart of a selection program for modular robots, which covers three phases: kinematics, dynamics, and sensor-based control. Furthermore, in their scheme, kinematics design played a dominant role in the sense that the other two phases contribute to modification of the result derived from the kinematics design. In this paper, such a design procedure is called a sequential design procedure. Matsumaru[3] presented a kind of sequential design procedure for his reconfigurable modular manipulator system. Cohen[4] and Hooper and Tesar[5] also used this method for synthesizing a modular robotic configuration. Fryer, McKee, and Schenker[6] proposed that an object-oriented concept could provide a useful tool for verifying the configuration of modular robotics systems, and they modeled robot resources and considered how the models could be adapted through the introduction of semantic annotations to accommodate the configuration process. These works fall in the first level of modular architecture. Chen and Burdick[7,8] presented an approach to task optimal modular robot assembly configuration. Philosophically, their work was built upon the concept of traditional mechanism type synthesis, where the joints actually make no physical sense, so their work is more at the third level of modular architecture. Fukuda and Kawauchi[9] proposed a synthesis approach for their CEBOT systems. Their approach assumed that each module had the same length and they considered only the reachability of working points with a robot hand as a performance index. Chen and Burdick,[8] Han et al.,[10] and Chocron and Bidaud[11] employed genetic algorithms (GA) for task-oriented design of modular robots, but considered only kinematics synthesis, that is, at the first level of modular architecture. In short, previous works in this research field implied that synthesis for robots with nonmodular architecture does not differ from synthesis for robots with modular architecture. The approach described in this paper goes away from this thought and is explained in detail in the remainder of the paper.

3. CONCURRENT DESIGN PROCEDURE FOR MODULAR ROBOT

In designing a robotic configuration with the fixed configuration, a large number of design variables can be involved. For example, a serial-connected manipulator with 6 DOF (degrees of freedom) has 18 geometric variables, 60 mass variables, 42 rigidity variables, and over 12 end-effector variables.[12] A complex design problem has to be decomposed into subactivities, because computation can be a problem. The decomposition can be carried out in either a sequential or a concurrent way.

3.1. Sequential Decomposition

To carry out a sequential decomposition, the original design problem should be reorganized as follows:

Objectives:

$$\min\{\mathbf{f}_1(\mathbf{x}_0), \mathbf{f}_2(\mathbf{x}_0, \mathbf{x}_1), \ldots, \mathbf{f}_i(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}), \ldots, \mathbf{f}_m(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{m-1})\}$$

Variables:

$$\mathbf{x} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{m-1}\}$$

Constraints:

$$\{\mathbf{g}_1(\mathbf{x}_0), \mathbf{g}_2(\mathbf{x}_0, \mathbf{x}_1), \ldots, \mathbf{g}_i(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}), \ldots, \mathbf{g}_m(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{m-1})\}$$

In the above expression, **f** stands for the vector of objective functions, **x** for the set of design variables, and **g** for the set of constraint equations. As a result, the whole optimization is decomposed into a series of subproblems, each involved in a subset of the original design variables, constraints, and objectives.

A sequential decomposition supposes that the original problem has a set of separate subobjectives and constraints, and that they are related to a different portion of design variables. In each sequential step, only a subset of design variables is involved. Some variables involved in previous steps are to be determined. By reducing the dimensionality of the design space, each design subactivity is made manageable. One of the types of sequential optimization may allow changing the variable, which has already been determined at some preceding optimization stages.

3.2. Concurrent Decomposition

Concurrent decomposition of a complex optimization can be expressed as

Objectives

$$\min\{\mathbf{f}_1(\mathbf{x}_0), \mathbf{f}_2(\mathbf{x}_0, \mathbf{x}_1), \ldots, \mathbf{f}_i(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{i-1}), \ldots, \mathbf{f}_m(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{m-1})\}$$

Variables:

$$(\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{n-1})$$

Constraints:

$$(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{m-1}) = \mathbf{h}(\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{n-1}), \qquad n \ll m$$

$$\{\mathbf{g}_1(\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{n-1}), \mathbf{g}_2(\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{n-1}), \ldots\}$$

In the above expression, **f** stands for the vector of objective functions, **y** for the set of modular design variables, **x** for the set of disciplinary variables, **h** stands for the mapping from **y** to **x**, and **g** for the set of constraint equations. The above expression implies that the design variables of an original problem can be organized into groups based on some measures or performance indices. Consequently, each group is expressed by very few independent variables, and most of the original variables become their dependent variables. Due to the reduction of the number of independent variables in the whole system, it is possible to apply the optimization design to the whole system. This is called concurrent optimization in the following discussion.

3.3. Necessity of CODM in Designing Modular Configurations

Using the concept of modular elements, an original optimal problem is decomposed by organizing the design variables into groups. Each group is represented by one or more design variable; the dimensionality of the design space is reduced greatly. As an example, take the 6-DOF serial robot mentioned in Section 2. If modular architecture is applied, the corresponding modular configuration has six link and six joint modules with 1 DOF. One discrete variable is defined to represent the type selection of each link module or joint module. Another discrete variable is defined to represent assembly selection of each link module. Together with the continuous design variables that describe the length variants of each link module, it has totally 24 design variables as follow:

1. Six discrete design variables for the type selections of six joint modules with 1 DOF.
2. Six discrete design variables for the type selections of six link modules.
3. Six discrete design variables for the assembly selections for six link modules.
4. Six continuous design variables for determination of lengths of six link modules.

On a general note, any system with modular architecture tends to reduce the total number of independent design variables. This is mainly because modularization can impose extra constraints on the system such that extra dependent relationships are created among variables. This also implies that the process of modularization of systems could affect the design methodology that is applied afterward to the design of these systems.

It should be further noted that the modular robotic architecture results in high coupling of modular design variables. These variables determine all kinematic and dynamic variables in a modular configuration. The kinematic and dynamic variables are dependent on modular design variables. Therefore, the simultaneous consideration of kinematic and dynamic constraints and objectives is the only appropriate way to design a modular robotic configuration.

4. CONCURRENT DESIGN OF MODULAR ROBOTIC CONFIGURATION

To formulate an optimal problem, we wish to know the design variables, design constraints, and objective functions. It is also important to note that from a given set of modules it may be possible to construct robots with various topologies—robots with serial or parallel kinematic structures.

In the following discussion, we focus on the task-oriented optimal design of modular robotic configuration at the second level of modular robotic architecture. The design problem is then stated as: given a set of module types, determine modules, and their ports for connections and connectivities, among modules.

4.1. Design Variables in a Modular Robotic Configuration

Suppose a modular robotic system consists of a set of link and joint modules, and it thus has NT_{link} types of link modules and NT_{joint} types of joint modules. NLM number of link modules and NJM number of joint modules form a modular robotic configuration. These modules are coded in a sequential way, and link and joint modules are coded separately. Because the modules are manufactured as individual physical entities, module types determine kinematic and dynamic variables/parameters. Modular variables or design variables involved in designing a modular robotic configuration and their dependent factors are illustrated in Table I.

4.2. Design Constraints

4.2.1. Definition Domains of the Design Variables

Design variables have their definition domains. For a modular robotic system, the domains of design variables are given in Table I. Notice further that, from the optimization viewpoint, the domains will be converted into constraints for the corresponding variables.

4.2.2. Task Specification

In the concurrent optimization of a modular robotic configuration, task specification includes the kinematics and dynamic requirements on the given n_w

Table I. Design variables and their definition domains.

	Variables	Definition domain	Dependent factors
Discrete variables for configuration variants	The type selected for the l_ith link module	$LM(l_i) \in \{1, 2, \ldots, NT_{\text{link}}\}$ $l_i = 1, 2, \ldots, NLM$	• Link mass • Position of the link mass center • Inertia • Postures of assemble ports • Limitation of the link length
	The type selected for the j_ith joint module	$JM(j_i) \in \{1, 2, \ldots, NT_{\text{joint}}\}$ $j_i = 1, 2, \ldots, NJM$	• Joint mass • Position of the mass centers • Inertia of the joint • Postures of assemble ports • Allowable accuracy • Allowable torque • Allowable velocity
	Assemble ports selected for the l_ith link	$LS(LM(l_i), NJ(l_i)) \in S_{LA}(LM(l_i), NJ(l_i))$	
	Assemble ports selected for the j_ith link	$JS(JM(j_i), 2) \in S_{JA}(JM(j_i), 2)$	
Continuous variables for a configuration	Adjustable length of the l_ith link	Determined by the link type	
	Robot location (x_0, y_0, z_0)	$\Re^3$	

work points. Typically, they can include

Posture:

$$(x_i, y_i, z_i, \theta_i, \varphi_i, \psi_i) \qquad i = 1, 2, \ldots, n_w$$

Velocity:

$$\left(\dot{x}_i, \dot{y}_i, \dot{z}_i, \dot{\theta}_i, \dot{\varphi}_i, \dot{\psi}_i\right) \qquad i = 1, 2, \ldots, n_w$$

Acceleration:

$$\left(\ddot{x}_i, \ddot{y}_i, \ddot{z}_i, \ddot{\theta}_i, \ddot{\varphi}_i, \ddot{\psi}_i\right) \qquad i = 1, 2, \ldots, n_w$$

Precision:

$$\left.\begin{aligned}\sqrt{\Delta x_i^2 + \Delta y_i^2 + \Delta z_i^2} &\le \Delta_i^T \\ \sqrt{\delta x_i^2 + \delta y_i^2 + \delta z_i^2} &\le \delta_i^T,\end{aligned}\right\} \qquad i = 1, 2, \ldots, n_w$$

External loads:

$$(f_{x,i}, f_{y,i}, f_{z,i}, m_{x,i}, m_{y,i}, m_{z,i}) \qquad i = 1, 2, \ldots, n_w$$

When there are obstacles in the working environment, more complex constraints should be considered in modeling robotic kinematic behavior.

4.2.3. Capability Constraints

This kind of constraint is caused by the joint capabilities, which includes velocity, acceleration, and torque:

Joint velocity:

$$\dot{\boldsymbol{\theta}}_i^B \le \dot{\boldsymbol{\theta}}_i \le \dot{\boldsymbol{\theta}}_i^T \qquad i = 1, 2, \ldots, LJM$$

Joint acceleration:

$$\ddot{\boldsymbol{\theta}}_i^B \le \ddot{\boldsymbol{\theta}}_i \le \ddot{\boldsymbol{\theta}}_i^T \qquad i = 1, 2, \ldots, LJM$$

Joint torque:

$$\boldsymbol{\tau}_i^B \le \boldsymbol{\tau}_i \le \boldsymbol{\tau}_i^T \qquad i = 1, 2, \ldots, LJM$$

4.3. Design Objectives

In a concurrent optimization design, both kinematic and dynamic objectives are included in an optimal formulation. In our design, the dimensions of these objectives are unified and the following objectives are considered to evaluate the total performance of a robotic candidate. All of the following indexes can be derived from kinematic and dynamic models obtained from automatic modeling.

4.3.1. The Manipulabilities on the Working Points

The manipulability measure is widely used to evaluate kinematic and dynamic performance at particular working points. To eliminate the dimensional inhomogeneities. The manipulability measure is expressed as[13]

$$E_1 = \sum_{i=1}^{i=n_w} \frac{1}{n_w} \cdot (\boldsymbol{\omega}_1 \cdot \mathrm{cond}[J_{v,i}] + \boldsymbol{\omega}_2 \cdot \mathrm{cond}[J_{\omega,i}]) \quad (1)$$

where E_1 is the nondimensional manipulability measure, $J_{v,i}, J_{\omega,i}$ are Jacobian matrices of the ith working point corresponding to $\Re^3$ and $SO(3)$, respectively, cond[•] is the condition number of the matrix [•], and ω_1, ω_2 are weighting coefficients and $\omega_1 + \omega_2 = 1$.

4.3.2. Error Measures on the Working Points

Dimensional inhomogeneity exists between translational and rotational errors. Therefore, the relative ratio between the actual error and the requirement of error limitation is used as the error measure for the working points:

$$E_2 = \max_{i \in (1,2,\ldots,n_w)} \left(\lambda_1 \cdot \frac{\max|\Delta_i|}{\Delta_i^T} + \lambda_2 \cdot \frac{\max|\delta_i|}{\delta_i^T} \right) \quad (2)$$

where E_2 is the nondimensional error measure, $\max|\Delta_i|, \max|\delta_i|$ are the maximum translational and rotational errors caused by the joint transmissions, respectively, Δ_i^T, δ_i^T are the requirements of translational and rotational error limitations, and λ_1, λ_2 are weighting coefficients and $\lambda_1 + \lambda_2 = 1$.

4.3.3. Required Torque for the Working Points

The required torques are measured to evaluate the energy cost, and the measure is expressed as

$$E_3 = \max_{i \in (1,2,\ldots,n_w)} \left(\sum_{j=1}^{j=LJM} \frac{|\boldsymbol{\tau}_{j,i}|}{NJM \cdot \boldsymbol{\tau}_j^T} \right) \quad (3)$$

where E_3 is the measure of required torques, $\boldsymbol{\tau}_{i,j}$ is the required force or torque of the jth joint at the

ith working point, and $\boldsymbol{\tau}_j^T$ is the maximum force or torque of the jth joint.

4.3.4. Joint Movement from One Working Point to Another

Another kinematics measure is the joint movement displacement from one working point to another. It is provided by the expression

$$E_4 = \sum_{j=1}^{j=LJM} \frac{\max_{i=(1,2,\ldots,n_w)}(q_{i,j}) - \min_{i=(1,2,\ldots,n_w)}(q_{i,j})}{\left(q_j^T - q_j^B\right)\cdot NJM} \tag{4}$$

where E_4 is the measure of required torque, $\max_{i=(1,2,\ldots,n_w)}(q_{i,j})$, $\min_{i=(1,2,\ldots,n_w)}(q_{i,j})$ are the maximum and minimum joint movements of the jth joint for all of the working points, respectively, and q_j^T, q_j^B are the given maximum and minimum limitation of the jth joint movement. Therefore, the global measure corresponding to a configuration candidate can be written as

$$E_G = E_1 - E_2 - E_3 - E_4 \tag{5}$$

where E_G is the global measure corresponding to a configuration candidate under a task specification.

From our definitions, we can get

$$0 \le E_i \le 1, \qquad i = 1,2,3,4 \tag{6}$$

Therefore,

$$-3.0 \le E_G \le 1 \tag{7}$$

4.4. Optimization via GA

We know from the above discussion that the design problem of a robotic configuration is an optimization problem with mixed discrete/continuous variables, complex constraints, and multiobjectives. Modularization imposes extra constraints on the design space: the kinematic and dynamic models derived from the design variables are coupled tightly. The traditional sequential decomposition becomes impractical. Moreover, the discrete variables generate a search space with discontinuities and nonlinearities. GA is employed in our study. There are two reasons that make GA attractive for this problem. First, mixed discrete/continuous variables can be handled easily. Second, they are theoretically and empirically proven to provide robust search in complex spaces to find nearly global optima.

A critical aspect in designing a GA is the basic mechanism that links the GA to the real design problem. This mechanism is twofold: first, a way of to encode solutions to the real problem on artificial chromosomes; second, an evaluation of a function (fitness) that returns a measure of how good an encoding is.[14]

As shown in Figure 1, in this algorithm, all of discrete/continuous design variables are represented in a string that consists of four substrings, which are used to represent type and assembly selections of link and joint modules, the lengths of link modules, and location of the robotic configuration, respectively.

GA is essentially an unconstrained search procedure within the representation space. Therefore, the constraints in the optimization problem have to be combined into the fitness. The following fitness function was selected for use:

$$\text{fitness} = \begin{cases} E_G & \text{if all constraints are satisfied} \\ -3.0 & \text{if all constraints are not satisfied} \end{cases} \tag{8}$$

The source codes of GA implementation were developed by Carroll.[15] It can be downloaded at no charge from the website: http://www.staff.uiuc.edu/~carroll/ga.html.

5. CASE STUDY

We demonstrate a 2-DOF robotics design problem. Suppose there is one link type and two joint types in a modular robotic system. The design variables are given in Table II, and the physical parameters of joint modules are given in Table III. For the link module, the mass is 1.5 kg, the inertia is 0.1680 kg·m^2, and the length is adjustable from 0.12 to 0.18 m. The assembly ports of this link module are shown in Figure 2.

The task specifications are given in Table IV. The design procedure to determine a modular robotic configuration is a mixed discrete/continuous optimization. A genetic algorithm is employed for concurrent optimization. For this simple case, the combination of configuration variables generates eight feasible configuration variants as shown in Figure 3. They are expressed by one discrete variable. There are four local continuous variables for each configuration candidate, i.e., two variables

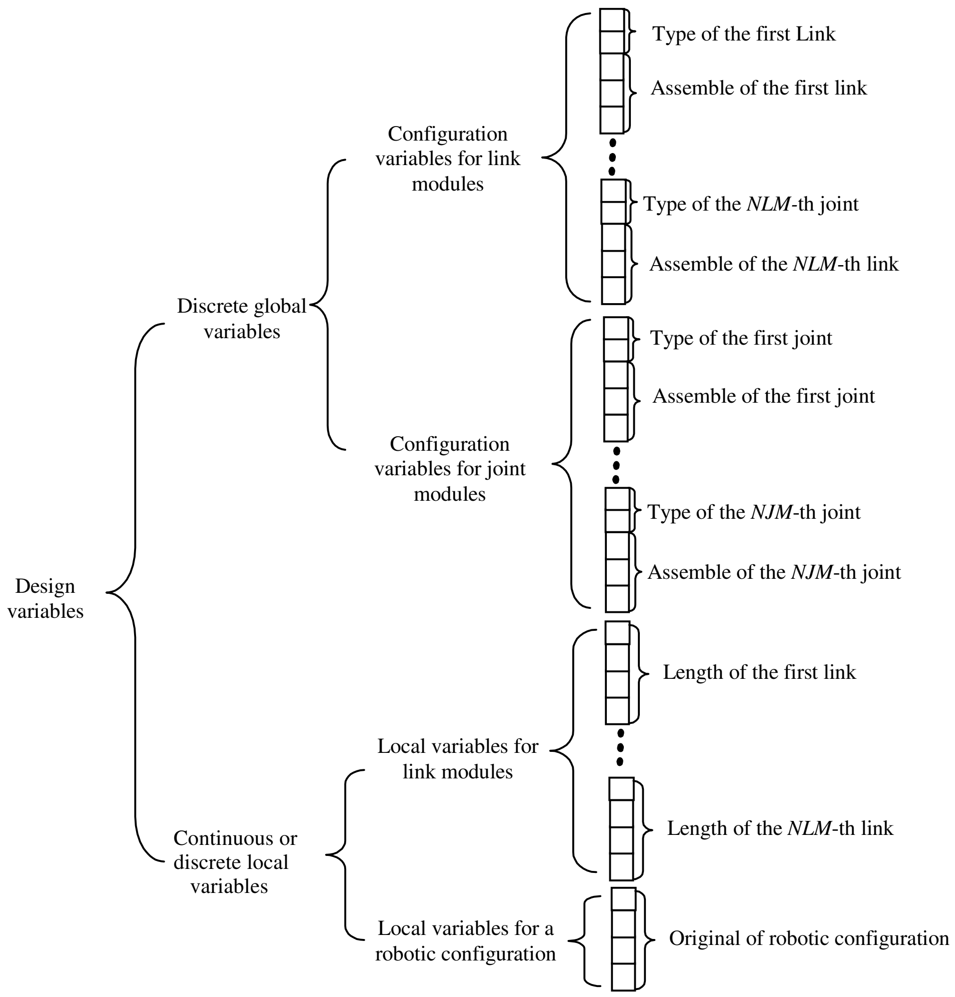
Design variables
Discrete global variables
Configuration variables for link modules
Type of the first Link
Assemble of the first link
Type of the NLM-th joint
Assemble of the NLM-th link
Configuration variables for joint modules
Type of the first joint
Assemble of the first joint
Type of the NJM-th joint
Assemble of the NJM-th joint
Continuous or discrete local variables
Local variables for link modules
Length of the first link
Length of the NLM-th link
Local variables for a robotic configuration
Original of robotic configuration

Figure 1. String representation of design variables in GA.

Table II. Design variables and their definition domains in the case study.

Variables	Name	Symbol	Definition domain
Configuration	Link type	$LM(1)$, $LM(2)$	$\{1\}$
	Joint type	$JM(1)$, $JM(2)$	$\{1, 2\}$
	Link assemble	$S_{LA}(1,2)$	$\{\langle 1,2\rangle, \langle 1,3\rangle\}$
		$S_{LA}(2,2)$	$\{\langle 1,2\rangle\}$
	Joint assemble	$S_{JA}(1,2)$, $S_{JA}(2,2)$	$\{\langle 1,2\rangle\}$
Local	Local robotic variable	d_{m1}, d_{m2}	$({}_{LT}d_1^B, {}_{LT}d_1^T)$
	Original of working site	(x_0, y_0)	R^2

Table III. Dependent parameters of the joint module types.

Physical parameters	Rotational joint	Translational joint
Mass 1	1.60 kg	2.00 kg
Mass 2		
Inertia 1	$0.004\ \text{kg}\cdot\text{m}^2$	$0.0036\ \text{kg}\cdot\text{m}^2$
Inertia 2		
Joint limitation	$(-75°, 75°)$	$(0.0, 0.080)$
Velocity limitation	≤ 60 rpm	≤ 3.0 m/s
Torque limitation	$\leq 140\ \text{kg}\cdot(\text{m/s})^2$	$\leq 120\ \text{kg}\cdot\text{m}\cdot\text{s}^2$
Accuracy	0.01°	0.01 mm

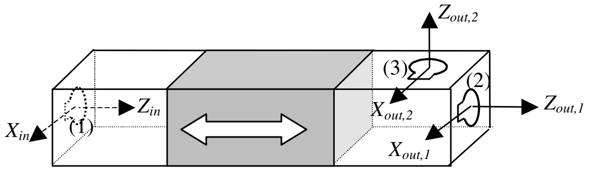

Figure 2. Link module and its assembly ports.

Table IV. Task specification of the case study.

No.	Position (m)	Velocity (m/s)	Acceleration (m/s^2)	Accuracy (m)	Loading (kg)
1	$(0.200, -0.200)$	$(0.500, 0.000)$	$(0.00, 0.00)$	0.0001	3.0
2	$(0.260, -0.200)$	$(0.000, 0.500)$			
3	$(0.260, -0.0961)$	$(-0.3535, -0.3535)$			
4	$(0.230, -0.1481)$	$(-0.3535, -0.3535)$			

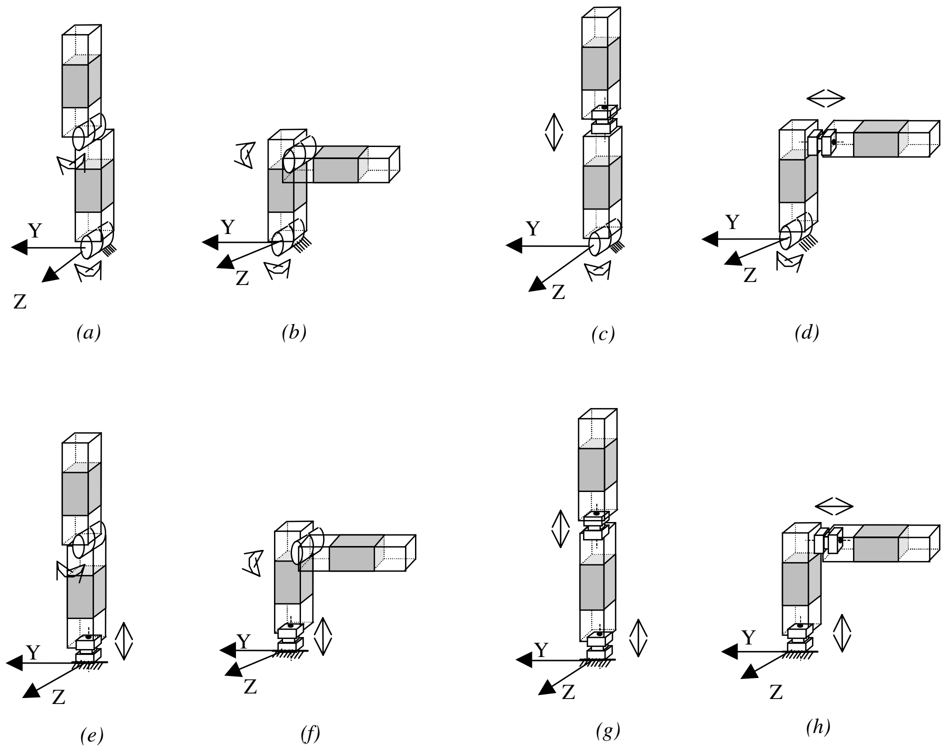

Figure 3. Eight feasible variants of robotic configurations.

for the lengths of the link modules and two variables for the location of robotics.

The probability of a crossover operation equals 0.5 and the probability of mutation equals 0.02. The best selections of design variables at each generation are shown in Figure 4. The average and maximum fitness in each generation are shown in Figure 5. The optimal result is presented as follows:

Optimal configuration:

First joint	Rotational type
Second joint	Rotational type

The length link module:

First link	0.14886
Second link	0.11999

Assemble ports of the link module:

First link	$\langle 1,2\rangle$
Second link	$\langle 1,3\rangle$

Location of robotic configuration:

$$(0.04667,\ -0.07675)$$

Evaluation index:

Manipulability	0.22905
Errors measure	0.48143
Energy index	0.02773
Joint movement	0.25032
Global index	−0.53044

6. CONCLUSION

This paper suggests classification of modular robotic systems into module-level, assembly-level, and configuration-level reconfigurable systems in terms of the way they achieve flexibility in applications. This three level architecture helps develop a new design methodology for modular robots. At this point, a concurrent design procedure based on multiobjective optimization has been developed for the mod-

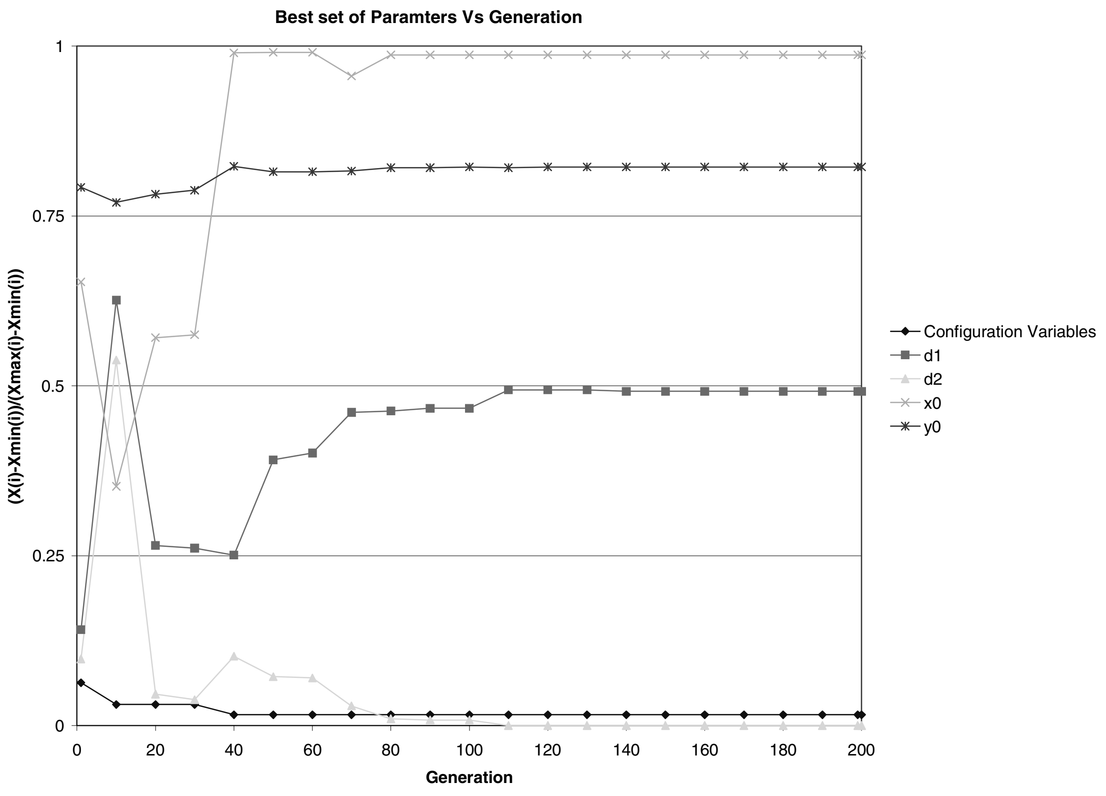

Figure 4. The best set of design variables vs GA generation.

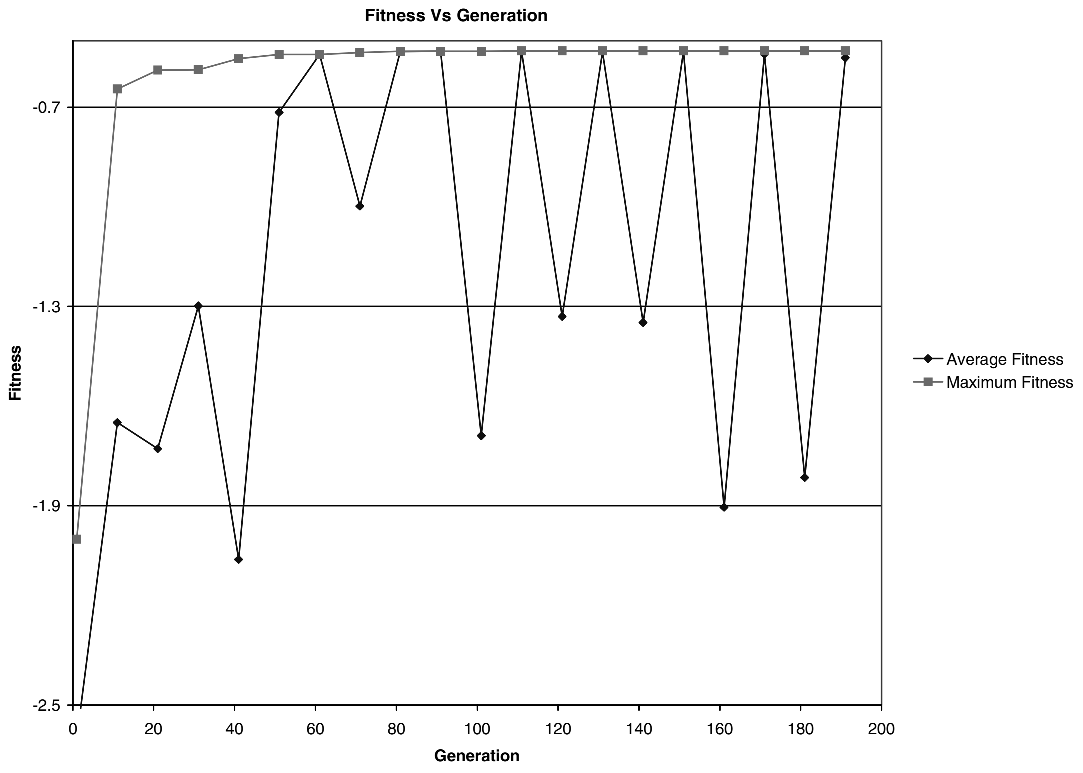

Figure 5. The average and maximum fitness vs GA generation.

ule-level and assembly-level architecture. This methodology is based on our observation that, in modular robots, the classification of parameters into kinematic and dynamic is not a primary concern: parameters are more associated with a complete description of modules (link module and joint module). This feature may be called module-oriented description, and it certainly leads to some dependencies developed among kinematics and dynamical parameters (from a conventional robot design viewpoint). Such dependencies result in reduction of the number of independent design variables and also coupling of design variables in terms of kinematic and dynamic objectives.

This coupling nature naturally requires kinematic and dynamic performance to be considered concurrently. The main benefits of this proposed concurrent design procedure are (1) achieving a global optimal solution and (2) avoiding repeated calculations in kinematics and dynamic analysis and synthesis.

The authors thank the Natural Science and Engineering Research Council (NSERC) of Canada and Atomic Energy Canada Limited (AECL) for financial support through a Cooperative Research and Development (CRD) grant awarded to the second author.

REFERENCES

1. J.J. Paredis Christiaan and K. Khosla Pardeep, Approach for mapping kinematics task specification into a manipulator design, 5th Int Conf on Advanced Robotics, Piscataway, NJ, 1991, pp. 556–561.
2. J.J. Paredis Christiaan and K. Khosla Pardeep, Serial link manipulators from task specifications, Int J Robotics Res 12 (1993), 274–286.
3. T. Matsumaru, Design and control of the modular robot system: TOMMS, IEEE Int Conf on Robotics and Automation, Nagoya, Japan, 1995, pp. 2125–2131.
4. R. Cohen, Conceptual design of a modular robot, ASME J Mech Design 112 (1992), 117–125.
5. R. Hooper and D. Tesar, Computer-aided configuration of modular robotic systems, J Comput Control Eng 5 (1994), 137–142.
6. A.J. Fryer, G.T. Mckee, and P.S. Schenker, Configuring robots from modules: An objected oriented approach, 8th Int Conf on Advanced Robotics, Monterey, CA, 1997, pp. 907–912.
7. I-Ming Chen and J.W. Burdick, Enumerating the non-isomorphic assembly configurations of modular robotic systems, Int J Robotics Res 17 (1998), 702–719.

8. I-Ming Chen and J.W. Burdick, Determination of task optimal modular robot assembly configurations, IEEE Int Conf on Robotics and Automation, Nagoya, Japan, 1995, pp. 132–137.
9. T. Fukuda and Y. Kawauchi, ''The cellular robotic system (CEBOT), a self-organized system,'' Intelligent robotic systems, S.G. Tzafestas (Editor), Dekker, New York, 1991, pp. 137–163.
10. J. Han, W.K. Chung, Y. Youm, and S.H. Kim, Task based design of modular robot manipulator using efficient genetic algorithm, IEEE Int Conf on Robotics and Automation, Albuquerque, NM, 1997, pp. 507–512.
11. O. Chocron and P. Bidaud, Genetic design of 3D modular manipulators, Proc IEEE Int Conf on Robotics and Automation, Albuquerque, NM, 1997, pp. 223–228.
12. B.O. Nnaji, Computer-aided design, selection, and evaluation of robot, Ph.D. Thesis, University of Massachusetts, 1986.
13. Z.M. Bi, Y.F. Li, and W.J. Zhang, A new method for dimensional synthesis of robotic manipulators, 5th National Applied Mechanisms and Robotics Conf, Cincinati, 1997, AMR97-041(01-07).
14. A.C. Nearchou and N.A. Aspragathos, Application of genetic algorithms to point-to-point motion of redundant manipulators, Mech Mach Theory 31 (1996), 261–270.
15. D.L. Carroll, Chemical laser modeling with genetic algorithms, AIAA J 34 (1996), 338–346.