Proceedings of the 11th
KACC, October 1996

Task Based Design of Modular Robot Manipulator using Efficient Genetic Algorithms

°Jeongheon Han* Wankyun Chung† , Youngil Youm‡ and Seungho Kim§

School of Mechanical Engineering
Pohang University of Science & Technology(POSTECH), Pohang, KOREA
E-mail: jhhan@risbot.rist.re.kr or wkchung@vision.postech.ac.kr

Abstracts Modular robot manipulator is a robotic system assembled from discrete joints and links into one of many possible manipulator configurations. This paper describes the design method of newly developed modular robot manipulator and the methodology of a task based reconfiguration of it. New locking mechanism is proposed and it provides quick coupling and decoupling. A parallel connection method is devised and it makes modular robot manipulator working well and the number of components on each module reduced. To automatically determine a sufficient or optimal arrangement of the modules for a given task, we also devise an algorithm that automatically generates forward and inverse manipulator kinematics, and we propose an algorithm which maps task specifications to the optimized manipulator configurations. Efficient genetic algorithms are generated and used to search for a optimal manipulator from task specifications. A few of design examples are shown.

Keywords Optimal Design, Task Based Design,Modular Manipulator,Genetic Algorithms

1. Introduction

Today many robot manipulators have been developed for practical use. But most of them cannot change their structure, so it is difficult to adapt such a system to variable tasks and environments. Easily adaptable robotic system has long been desired to accomplish different tasks. Alternative for this problem is a modular robot manipulator design. The term modular robot manipulator is referred to a robot manipulator assembled from discrete mechanical joints and links into one of many possible manipulator structures [6]. Modular robot manipulator utilizes a stock of interchangeable link and joint modules of various sizes and performance specifications. By recombining the modules, different robots can be created so as to suit a variety of task requirements. Each module is connected through standard mechanical and electrical interfaces. Such a manipulator has several advantages over conventional manipulators; low cost, easy maintenance, easy modification and durability against system malfunctions. The feasibility of the modular robot manipulator has been carried out in the prototype systems built in several research institutes [1] [5] [6] [8], and kinematic analysis of modular robot manipulator have also been researched in [4].However, most of these systems lack the property of reconfigurability, which is the most important feature to the concept of modular manipulator.

Effective use of the modular manipualtor requires task based design software. This software generates optimal available modular assembly as input descriptions of the task. Several different approaches of module assembly planning and task optimal configurations were shown in [9] [2].

The objective of our research in the area of mechanical design of modular robots is to develop an inventory of basic modular units, which will allow a user to configure the most suitable robot geometry for a set of tasks at hand without vast modification to the concept of conventional robot manipulators. This paper describes the design method of newly developed modular robot manipulator and the methodology of a task based reconfiguration of it.

The outline of this paper is as follows : Section 2 describes the design of newly developed modular robot manipulator. Section 3 explains what type of the modular manipulator can be selected from task specifications. Section 4 introduces genetic algorithm and the three commonly used operators and the determination method of mutation probability and the ranges of parameters. Section 5 presents the results for a few of examples and finally, a discussion follows.

2. Mechanical Design

The modular robot manipulator is composed of a wide variety of hardware modules available. We built three kinds of modules: the manipulator base, a link module, two pivot modules. The design of each module is independent of other modules except for the module interface which is standardized. The base module and link module have no degrees of freedom while each joint module has one degree of freedom.

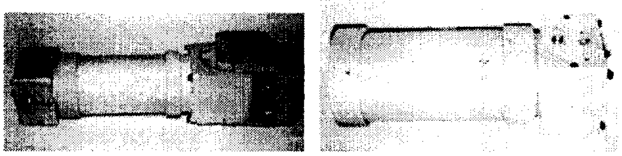


Fig. 1 (a) Joint Module and (b) Link Module

2.1. *Joint Module*

For modular robots, the actuator has to be located in the joint module. Generally three types of 1-DOF joint modules are considered; revolute(rotating and pivoting) and prismatic joints. A rotate type joint has link axes which are co-linear with each other and with the joint axis. A pivot one has link axes which are both perpendicular to the joint axis. For simplicity and convenience, we built only two pivoting type joint modules. Our prototype design for the pivot joint is shown in Fig. 1(a). It compactly fits a DC motor with encoder and gear transmission(the reduction ratio is 1:100) and is rated at a regular speed of 30 rpm and regular torque of 30Nm. Each module has a mass of approximately 6.5kg. Of course there are a pair of locking element at both sides of joint module.

2.2. *Link Module*

The purpose of the link module is to change the distance between the rotational axis of the adjacent joint modules. The link module is shown in Fig. 1(b). It consists of two cylinders, one inserted in the other. Two cylinders are linearly moved manually relative to each other in the direction of the link, and are locked by adjusting ball into the hole. There should be no offset and twisting between the two sides.

2.3. *Locking Mechanism*

The major difference between modular manipulator and a conventional manipulator is the standardized interface component. In order to assemble the joint and link modules into a manipulator, locking mechanism is required. This connection must both

---

*Graduate Student in Ph.D Program, POSTECH
†Associate Professor, POSTECH & ARC
‡Professor, POSTECH
§Reasearcher,KAERI

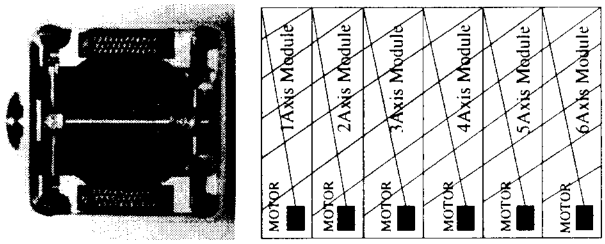


Fig. 2 (a) Locking Mechanism and (b) Electrical Connection

align the modules and lock them together with sufficient strength to transmit the internal forces generated by the movement of the manipulator. We developed a quick-coupling mechanism with which a secure mechanical connection between modules can be achieved by simply plugging by hand; no tools are required. It is shown in the photograph in Fig. 2(a). The mechanical coupling is accomplished using kinematic linkage and spring. As the mechanical connection is made, electrical connections are also established. Inside the locking mechanism are two fair of modular connectors which have 34 electrical pins on both sides. These correspond to matching female components on the mating connector. Sets of pins are wired in parallel to carry the 75V - 2A power for each motor and 5V power for each encoder. Additional pins carry signals for control. An internal wiring method is selected to provide the neat shape. The parallel connection method is developed for better communication. Principles of the parallel connection method is shown in Fig. 2(b) . In Fig. 2(b) electrical connection of each module is same, so any connection of module has same results. Owing to the parallel connection method, the control system is concentrated in the control unit, and the jacobian matrix is applied to the control software to be independent from the configuration of the assembled manipulator without special modification [8]. Of course, the parallel connection method gives rise to connect many electrical lines. There is a trade off between complexity of joint module and complexity of pin connection. Another important advantage of the parallel connection is that the number of components on each module is reduced, and it becomes easy to control and achieve a small and lightweight module, which is extremely effective for improving the payload and tip speed.

2.4. Prototype System and Experiment

Newly developed module is composed of two pivoting joint modules, a link module and a base module. These modules can be assembled into any desired configuration and shape, for instance, an horizontal type or an vertical type. Fig. 3 shows two DOF planar modular robot manipulator.

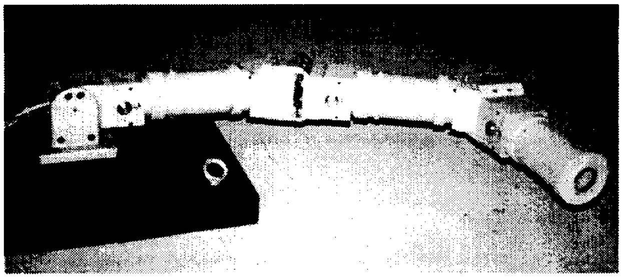


Fig. 3 Planar Two D.O.F Modular Robot

The main experimental device used in this paper is planar two DOF modular robot manipulator shown in Fig. 3. Joint position is measured by an incremental type encoder which is directly attached to the motor shaft included in each joint module. It is controlled by independent joint control algorithm and the controller used for experiments is a pentiup chip based personal computer with real time capabilities. Fig. 4 shows arm motion for following square trajectory and joint motion error. Joint position error is small enough to apply for industrial purposes.

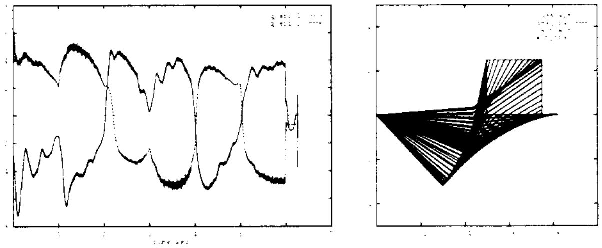


Fig. 4 Robot Motion and Joint Motion Error

3. Modular Robot Kinematics

To automatically generate a modular robot assembly for a given task, structure-decision method has to be developed. For simplicity and convenience, we considered only two common types of revolute joint in our modular manipulator system. These two types are rotate and pivot(or bending) and are distinguished by the orientation of the joints link axes with the joint axis as mentioned before.

Each module has to be connected sequentially and there are no manipulator offsets. Two coordinate systems for each work point with two straight lines along the z axis of each coordinate system are defined. One is determined as position and orientation of base, the other is those of end-effector. This coordinate system is shown in Fig. 5. A simple three-link manipulator(BBB)[1] approach is considered. Minimal DOF, structure of manipulator and their link length can be determined by checking torsion angle between basic links and distance of PQ. First we must determine the position of P and Q. This requires the knowledge of position and orientation of robot base and end-effector which are given from task specifications. Then the position of Q and P is determined from Eq. (1), (2). If basic links cannot satisfy the calculated angle of torsion given by Eq. (3), (4), (5), rotating joint 'R's is needed for correction the torsion angle.

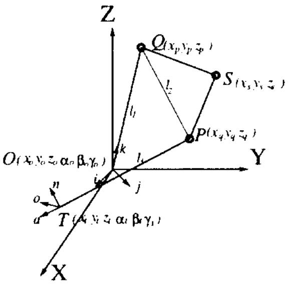


Fig. 5 Coordinate System of Modular Assembly

$$Q = l_1\vec{k} \quad (1)$$

$$P = T - l_3\vec{a} \quad (2)$$

$$\vec{j}\cdot\overrightarrow{PQ} \overset{?}{=} 0 \quad (3)$$

$$\vec{a}\cdot(\vec{k}\times\overrightarrow{QP}) \overset{?}{=} 0 \quad (4)$$

$$\vec{o}\cdot(\overrightarrow{PQ}) \overset{?}{=} 0 \quad (5)$$

We concluded that type of manipulator can be determined from submanipulator of RBBRBR [2]. So we can obtain selection vector which is whether the joint is selected or not. From the selection vector, we can obtain forward kinematics and inverse kinematics and forward dynamics and inverse dynamics. With these we can optimize the link length of the obtained manipulator structure.

---

[1] B means bending or pivoting joint

[2] RBBRBR is a manipulator which is offset free puma-type-manipulator, and this is minimal representation of the structure of nonredundant modular robot manipulator.

4. Genetic Algorithms

Genetic algorithms are stochastic optimization algorithms that were originally motivated by the mechanisms of natural selection and evolutionary genetics [3].

GAs are inherently parallel, because they simultaneously evaluate many points in the parameter spaces(search spaces). In the most conventional search techniques, a single point is considered based on some decision rules. However GAs work with a population of binary strings, searching many peaks in parallel. By employing genetic operators, they exchange information between the peaks, hence reducing the possibility of ending at a local optimum and would be more likely to converge to the global optimum. More details about GAs can be found in Goldberg [3].

Table 1 is the pseudo code of simple genetic algorithmexplains the operation principles of GAs. Table 1 is the pseudo code of simple genetic algorithmexplains the operation principles of GAs. To initialize the procedure, the population is created by randomly selecting members from the allowable space. Then each of the individuals in the population are evaluated. Next an iterative loop is entered in which some members of the population are selected for the next generation based on their fitness. These members are recombined to form the population of the next generation, and finally the members of the new generation are evaluated. If the search goal is achieved, or an allowable generation is attained, algorithm is stopped.

```
    procedure                  Genetic Algorithm
begin
        initialize   Population(k(generation) = 0)
        evaluate     Individuals in Population(k)
        while        Termination Condition not satisfied, do
        begin
                     k = k + 1
                     select Population(k) from Population(k-1)
                     recombine Individuals in Population(k)
                     evaluate Individuals in Population(k)
        end
end
```

Table 1 pseudo code of GA

4.1. *Chromosome coding and decoding*

GAs work with a population of binary strings, not the parameters themselves. For simplicity and convenience, binary coding is used in this article. With the binary coding method, the link length l_1, l_2 and l_3 would be coded as binary strings of 0's and 1's with the bit length B_1, B_2 and B_3 for the parameters concerned with the resolution R_i specified by the designer in the search space. The bit length B_i and the corresponding resolution R_i is related by

$$R_i = \frac{U_{max} - U_{min}}{2^{B_i} - 1} \tag{6}$$

where U_{max} and U_{min} are the upper and lower bounds of the link length l_i.

4.2. *Basic Operators of GA*

There are three basic operators which are reproduction, crossover and mutation in GA.

Reproduction is based on the law of survival of the fittest. There are many ways to achieve effective reproduction. One simple scheme is selecting individual strings for reproduction according to their fitness. Individuals with higher fitness values have a higher probability of being selected for mating and subsequent genetic action according to the distribution.

In crossover, two individual strings, selected using the reproduction operator, create two new individuals by exchanging partial strings with each other. A crossover site along the string length is selected uniformly at random. Next step is to exchange all characters following the crossing site. This crossover allows combination of advantageous substructures into individuals more fit than either parents.

Mutation operator enhances the ability of GAs to find a near-optimal solution. When used sparingly in combination with reproduction and crossover, mutation is an insurance policy against the loss of important genetic material at a particular position. In the case of binary coding, the mutation operator simply flips the state of a bit from 0 to 1 or vice versa.

4.3. *Fitness and Objective Function*

The main role of an objective function in design is to choose one optimal design from the candidate which satisfies given constraints. For a dexterity measure to be used for design, it must be independent of the scale of a manipulator. In this paper, we use the relative manipulability as a dexterity measure for kinematic optimization problem. The relative manipulability is a dimensionless scale independent measure of manipulability and is defined as [2]

$$M_r = \frac{M}{f_M} \tag{7}$$

where M is the order independent manipulability and f_M is a function of dimension ([length]^2). the measure M is obtained as

$$M = \sqrt[m]{JJ^T} \tag{8}$$

where J is the Jacobian matrix of instantaneous kinematics and m is the order of task space. For a 3 DOF planar manipulator with 2 dimensional task space, m = 2. We use l^2 for f_M, where l is a total length of a manipulator. In general, link length is defined as $(l_i = \sqrt{(a_i^2 + d_i^2)})$, where link length a_i,and offset d_i are D-H parameters. Since the task is given along trajectory, objective function is as follows.

$$\begin{aligned} \text{objective function} &= M_{rms} \\ &= \sqrt{\frac{\sum_i [{M_{ri}}^2 T_{sampling}]}{T_{completion}}} \end{aligned} \tag{9}$$

4.4. *Determination of mutation probability and ranges of parameters*

Mutation probability and ranges of parameters affect significantly the efficiency of GAs. So they are important , we proposed a determination method of mutation probability and ranges of parameters as:

$$p = \frac{i}{i_{max}} p_{max} \tag{10}$$

$$\begin{aligned} x^j_{min} &= \bar{x}^j - (\bar{x}^j - x^j_{min}) \exp(\eta \frac{i}{i_{max}}) \\ x^j_{max} &= \bar{x}^j + (-\bar{x}^j + x^j_{max}) \exp(\eta \frac{i}{i_{max}}) \end{aligned} \tag{11}$$

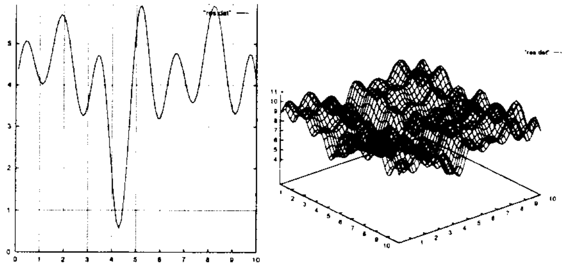
where η is the reducing rate, x^j is j^{th} parameter and $\bar{x}^j$is j^{th} fittest. p is the probability of mutation, i is a generation number and i_{max} is the maximum generation number. Eq. (10) shows how the mutation probability determines. We know that the probability of mutation increases linearly as iteration proceeds. This equation enhances the ability of GAs to find a near-optimal solution. Range of parameters are determined by Eq. (11). The role of this equation is to reduce searching space efficiently to find an optimal solution.

5. Design Examples

It is needed to show how the proposed genetic algorithm operates well. Its performance is tested by using highly nonlinear and difficult problem to find a maximized parameter of object function shown in Eq. (12).

$$\text{Objective Function} = f(x) + f(y) \qquad 0 < x, y < 10 \tag{12}$$

$$\begin{aligned} f(x) = {} & 4 + \sin(x)[\sin^2(x) + (\frac{\cos(x)}{\sin(2x)})\sin(4x) \\ & + \sin(2x)\arctan(x) + \cos(3x)\arctan(\sin(x))] \\ f(y) = {} & 4 + \sin(y)[\sin^2(y) + (\frac{\cos(y)}{\sin(2y)})\sin(4y) \\ & + \sin(2y)\arctan(y) + \cos(3y)\arctan(\sin(y))] \end{aligned}$$

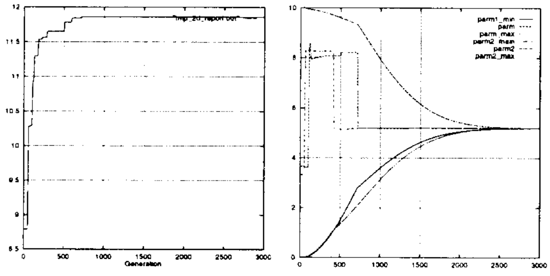


(a)One dimension (b)Two dimension

Fig. 6 Objective function

Then the optimization problem is to find parameter which makes object function shown in Eq. (12) maximized.

Fig. 6 shows one-dimensional and two-dimensional objective function. There are so many local maximum that gradient descent method ,simple GA, or other tool may not find the global maximum. Fig. 7 shows that fitness and parameter values for two dimensional problem. We set the maximum mutation probability and reducing rate η as 0.1 and 0.0045 respectively, and the intial ranges of parameters is shown in Fig. 7.



(a) Fitness (b) Parameter

Fig. 7 Fitness and Parameter

From Fig. 7, fitness value is exponentially increaing and searching spaces of parameters are decreased effectively by using the proposed method. After 700 generations we could know that exact values are found.

Now, by using the proposed GAs, we try to find the kinematic optimized modular manipulator whose kinematics is described in secton 3, and the objective function given in section 4.3. Table 2 are the task specifications including position and orientation, and the number of task is 4. The manipulator is chosen as a 'RBBRBR' whose DOF is six. [3]

Fig. 8 shows that the fitness(objective function) of parameter has grown as generation increases and optimized link length. Fig. 9 shows maximized manipulability and kinematically optimized arm motion.

TaskNum	p_x	p_y	p_z	θ_p	ϕ_p	ψ_p
1	0.2	0.4	-0.1	-300.0	200.0	300.0
2	0.1	0.3	0.1	-200.0	200.0	300.0
3	-0.1	0.2	0.2	-200.0	100.0	300.0
4	-0.2	-0.3	0.4	-200.0	100.0	200.0

Table 2 Task Specification

6. Concluding Remarks

This paper describes the design method of newly developed modular robot and the method of task based reconfiguration of it. We developed a prototype modular manipulator and described the design method of this prototype. The prototype includes two pivoting joint modules and a link module, and a controller consisting of a pentium chip based computer with real-time capabilities. New locking mechanism is proposed and it provides quick coupling and decoupling. The most important point for the parallel connection is that it is easy to achieve a small and lightweight module, and as a result it improves the payload and tip speed. It is controlled by independent joint control algorithm and the experimental results show that it is operated well.

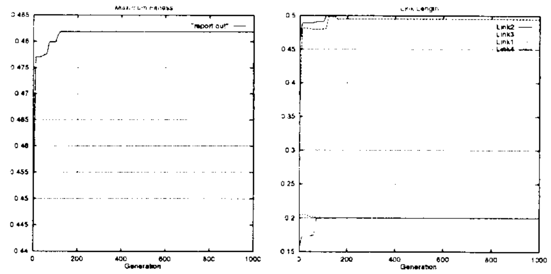


Fig. 8 Fitness and Link Length

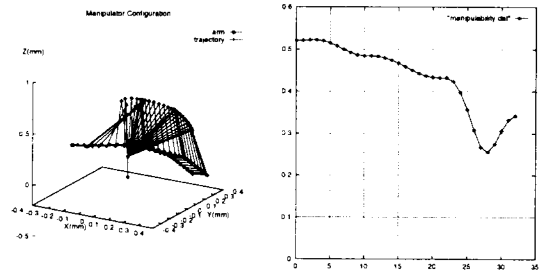


Fig. 9 Simulation and Manipulability

We also proposed an algorithm that automatically generates forward and inverse manipulator kinematics, and maps task specifications to manipulator configurations. Proposed genetic algorithms is used to search for an optimal manipulator from task specifications.

---

[3] 'RBBRBR' is like a offset free puma arm.

REFERENCES

[1] D. Schimitz,P. K. Khosla and T. Kanade, "The CMU Reconfigurable Modular Manipulator System", Technical Report,Carnegie Mellon University,CMU-RI-TR-88-7,16pages,1988.

[2] J. O. Kim and P. K. Khosla, "A Multi-Population Genetic Algorithm and Its Application to Design of Manipulators", *IEEE/RSJ int. workshop on Intelligent Robots and Systems(IROS'92)*, July,1992.

[3] D. Goldberg,"Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley,1989.

[4] L. Kelmar and P. K. Khosla, "Automatic Generation of Kinematics for a Reconfigurable Modular Manipulator System", *IEEE Proc. of Int. Conf. on Robotics and Automation*,pp. 663–668,1988.

[5] T. Fukuda and S. Nakagawa, "Dynamic Reconfigurable Robotic System", *Proc. IEEE Conf. on Robotics and Automation*,Vol. 3,pp. 1581–1586,1988.

[6] K. H. Wurst, "The Conception and Construction of a Modular Robot System", *IFS,proc. Int. Symp. on Industrial Robotics(Belgium)*,pp. 37–44,1986.

[7] R. Cohen, M. G. Lipton,M. Q. Dai and B. Benhabib, "Conceptual Design of a Modular Robot", *Journal of Mechanical Design*,Vol. 114,pp. 117–125,1992.

[8] T. Matsumaru, "Design and Control of the Modular Robot System : TOMMS", *IEEE Proc. of Int. Conf. on Robotics and Automation*, pp. 2125–2131, 1995.

[9] C. J. Paredis, P. K. Khosla, "Kinematic Design of Serial Link Manipulators From Task Specifications", *Int. J. Robotics Research*, 12, no. 3, pp 274–287,1993.