'92 KACC 1992. 10. 19~21

Framework for Task Based Design of Robot Manipulators

Jin-Oh Kim and Pradeep K. Khosla
The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, 15213

ABSTRACT

In this paper, a new design technique called Task Based Design (TBD) is proposed to design an optimal robot manipulator for a given task. Optimal design of a manipulator is difficult because it involves implicit and highly nonlinear functions of many design variables for a complex task. TBD designs an optimal manipulator which performs a given task best, by using a framework called Progressive Design which decomposes the complexity of the task into three steps: kinematic design, planning and kinematic control. An example of TBD is presented to demonstrate the efficiency and effectiveness of our framework.

1. INTRODUCTION

In theory, a robot's task can be changed by simply loading a new program into its controller; however, in practice, this is rarely the case. Each robot has a specific configuration and limited sensing capabilities that support only the applications for which the system was designed. For example, SCARA type manipulators are suitable for table-top assembly operations requiring selective compliance and accuracy but are not good for tasks requiring large workspace. The CMU Reconfigurable Modular Manipulator System (RMMS) was conceived to address the problems associated with conventional fixed-configuration manipulators [11].

The RMMS utilizes a stock of assemblable joint and link modules of different size and performance specifications. The modularity in mechanical, electrical and electronic design allows the user to design a manipulator that is appropriate for a given task. As opposed to existing commercial manipulators which are made to perform as many tasks as possible, the RMMS has a feature that can provide a special manipulator for a given specific task. For a fully exploiting the above features of the RMMS, we need a framework for design of manipulators based on a given task specification.

Task Based Design is an endeavor to map a given task onto a manipulator that can perform the task. The ultimate goal of TBD is to obtain *kinematic parameters* of a manipulator from a given task. The kinematic parameters are DOF (degrees of freedom), the Denavit-Hartenberg parameters (type, dimension and pose) and the base position of a manipulator. The complexity with lots of design variables, and highly nonlinear and implicit functions are characteristic of a general manipulator design. For TBD, another complexity is related to a task. Even a simple two dimensional trajectory can hardly be mapped onto a manipulator that can perform the task best. The goal of this paper is to introduce our design methodology to solve these two complexities (manipulator and task complexities).

This paper is organized as follows. In Section 2, we define the problem of Task Based Design with three assumptions. Next in Section 3, we introduce our design strategy and framework in detail. We present an example of TBD in Section 4. Finally, we summarize our research in Section 5.

2. PROBLEM DESCRIPTION AND ASSUMPTIONS

Design variables for robot manipulators include

1. Degrees of freedom (N),
2. Denavit-Hartenberg (D-H) parameters (α_i, a_i, d_i, θ_i) [4] and
3. Base position (B),

where α_i is the i-th link twist angle, a_i the i-th link length, d_i the i-th joint offset distance and θ_i the i-th joint variable (angle). The D-H parameters is composed of type (T), dimension (D) and pose (P) of the manipulator. The type of n DOF manipulator is represented by n sets of triple (α_i, a_i, d_i). The magnitudes of a_i and d_i are determined by dimensional synthesis and the magnitude of θ_i is determined by pose synthesis. But both are made at the same time. The design problem addressed in this paper is to map the specified task onto the above kinematic parameters. The problem in which the above five variables are unknown can be stated as follows.

> Design a manipulator (M) of DOF N, type T, dimension D, pose P and base position B subject to the given task specification **E**, manipulator specification **R** and some optimality criterion **C**.

For this problem, design variables are 5 tuples of (N,T,D,P,B). The input to this problem is a triple (**E**, **R**, **C**). A schematic diagram of Task Based Design is shown in Figure 1. The task specification is related to only a given specific task and the manipulator specification is related to the above five design variables. We use a dexterity measure as an optimality criterion.

Even though our ultimate goal is to solve this problem, our optimization algorithm does not solve this problem directly because it is highly nonlinear and tremendously complex. Instead, we solve one of the subproblems in which DOF and type are given, and dimension, pose and base position are unknown. We repeat solving this subproblem with other type and/or DOF until a manipulator that satisfies the task/manipulator specifications is obtained.

We make the following assumptions in this paper:

1. Only kinematic and static task descriptions are considered.
2. Only serial manipulators with revolute joints are considered.
3. The base position of a manipulator is not movable.

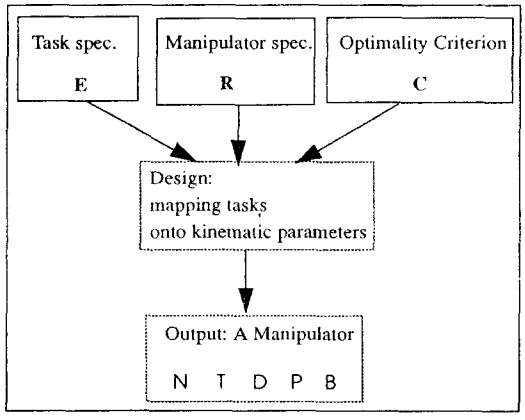


Figure 1: Task Based Design

3. DESIGN FRAMEWORK

The optimization function of Task Based Design is composed of highly nonlinear and complex functions and includes a large number of design variables. A solution can hardly be obtained by running an optimization algorithm. The complexity of design can be divided into a task complexity and a manipulator complexity. Most of task complexity is related to a trajectory to follow in space, and most of manipulator complexity comes from the large number of design variables. To derive a good solution, it is necessary to construct a good design framework that decomposes these two complexities.

First, we propose a high level framework called "Progressive Design" that decomposes the design into several steps in which we include progressively more and more task specifications. Progressive Design decomposes the task complexity and is composed of three steps: kinematic design, planning and kinematic control. In the first step (kinematic design), assuming that a finite number of task points can approximate a given trajectory of the end-effector, we design the optimal values of DOF, type, dimension, pose and base position for the given task. Suppose, for instance, that a trajectory for a task is given as a circle as shown in Figure 2. The first step of kinematic design optimizes all design variables at the four task points in this example. The second step (planning) tries to find the optimal poses at the finite number of sub-task points between the task points of the kinematic design. In the example, there are four sub-task points between task points, totalling 16 sub-task points. Finally, all task points are connected by the third step (kinematic control) along the given trajectory.

Each step of Progressive Design is still complex with many variables and nonlinear functions. The kinematic design is the most complex step because it involves all design variables of DOF, type, dimension, pose and base position. This requires a low level framework to decompose the complexity of the kinematic design. The following steps (planning and kinematic control) also need their frameworks. Figure 3 shows both high and low level frameworks.

In the kinematic design, we propose to decompose design variables into two groups to reduce complexity. The first group includes DOF and type, while the second group includes dimension, pose and base position, of which the optimal values are obtained by an optimization algorithm. With a given DOF and type, our optimization algorithm works on the space of 3-tuples (dimension, pose and base position). We have assumed that type is a discrete variable because of the above decomposition. The main reason of the decomposition is to reduce the complexity of Task Based Design. Due to

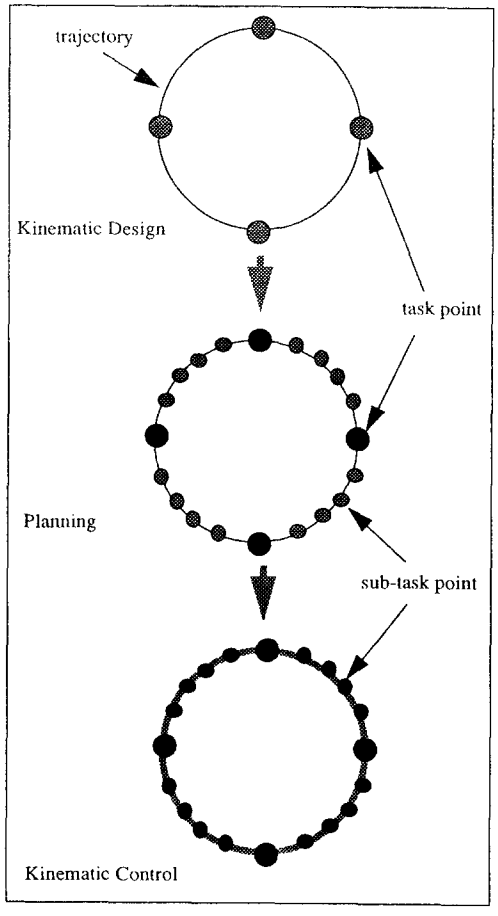


Figure 2: Progressive Design

this decomposition, the optimization function in our framework has a number of design variables reduced by almost half, compared with an optimization function when the type is included as a continuous variable.

The framework of the kinematic design consists of three inputs (task specification, manipulator specification and dexterity measure) and three modules (DOF selection, type generation and optimization algorithm). The task/manipulator specifications are formulated as design constraints and the dexterity measure is used as an optimality criterion. DOF selection is based on the dimensional analysis of a given task; more specifically, a trajectory. Type generation creates all possible types for a given DOF and is based on simple rules derived from existing manipulators. As an optimization algorithm, a Multi-Population Genetic Algorithm (MPGA) is proposed. This algorithm implements an existing Genetic Algorithm (GA) in parallel and exploits a parallel nature of the task specification.

The framework of the planning is obtained from the framework of the kinematic design by turning off DOF selection and type generation. Pose (joint angles) is the only remaining variable, since the other variables are determined in the first step of the kinematic design.

The framework of the kinematic control is based on the Resolved Motion Rate Control (RMRC)[13]. For redundant manipulators, we use the pseudoinverse approach[10]. A difference from others[10][14] is that our kinematic control is two point boundary value problem where poses at two adjacent task points become end boundaries.

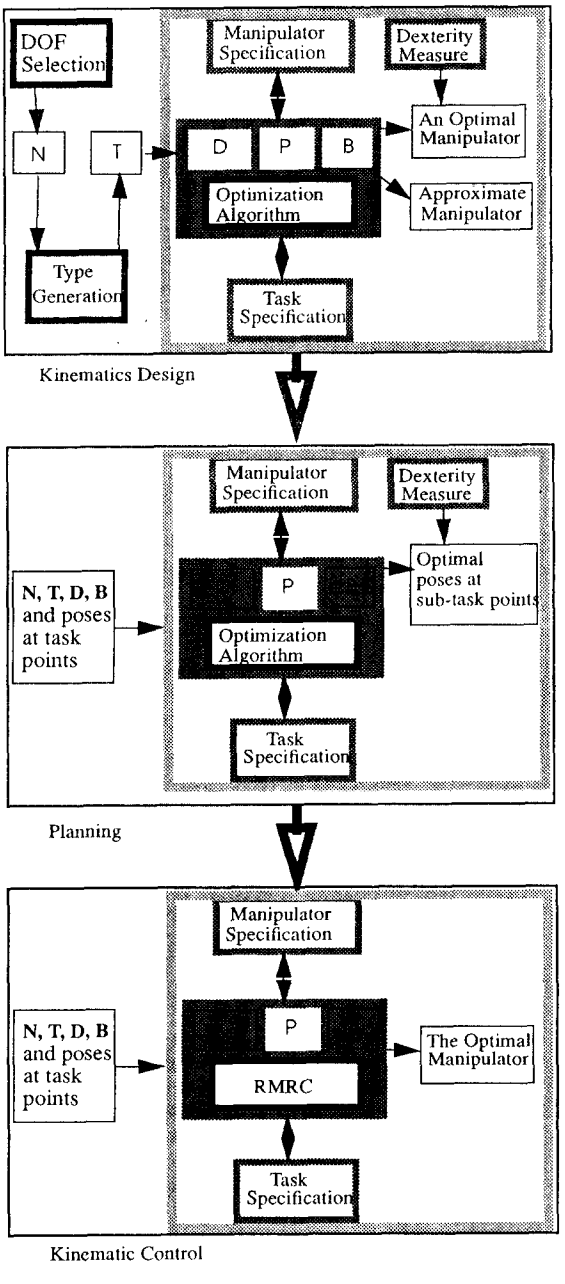


Figure 3: Schematic Diagram of design framework

In the following subsections, we describe in detail the three inputs and the three modules of the framework of the kinematic design.

Task specification

In general, *task space* is defined as a space of a given task that is required to be reached and *workspace* as a volume reachable by a manipulator. Concepts of the task space and the workspace are extended and generalized in our design problem. The task specification is a generalization of the task space while the manipulator specification in the following subsection is that of the workspace. Below is summarized the task specifications that are used in our design.

(1) Reachability constraint (RC) must be satisfied at the finite number of task points, which are assumed to approximate a given trajectory. The reachability constraint enables us to dispense with inverse kinematics that exists for only special class of manipulators[12].

(2) Task constraint (TC) varies depending on the given task. Tasks such as bracing, obstacle avoidance, singularity avoidance and dexterity may be included as a task constraint. Among these, we consider dexterity and singularity avoidance, which can be represented by features of the velocity and force ellipsoids[1].

(3) Joint angle change constraint (JAC) is based on the fact that the joint angle change between two adjacent task points must be kept small. The JAC is important to prevent local minima in poses that result from the multiple solutions of inverse kinematics.

Manipulator specification

Manipulator specification, an extended concept of the workspace of a manipulator, constrains design variables of dimension D, pose P and base position B. Like the task specification, the manipulator specification is expressed in the form of constraints.

(1) Dimension constraint (DC) is a constraint on link (link length and joint offset). The constraint is that an inner link is always longer than an outer link. This constraint is based on the following observations; outer links are more appropriate for fine motion and inner links for gross motion, and the dextrous workspace[1] requires outer links to be shorter than inner links.

(2) Pose constraint (PC) or joint constraint (JC) assumes that the range of motion of an actuator is limited. For a given manipulator (structure), this constraint is expressed by two absolute values link $[\theta_{min}, \theta_{max}]$. For a design problem in which a manipulator is not given but only joint modules are given, this constraint is expressed by an interval between two limits, since a joint module can be assembled to make any arbitrary limits with the interval. Suppose that the range of a joint module is 100°. When it is assembled, the joint limit can be [0°, 100°] or [20°, 120°]. Only the interval matters. In this paper, the joint constraint expressed by its interval before a structure is decided, is called the joint interval constraint (JIC), and that expressed by two absolute values when a structure is decided, is called the joint limit constraint (JLC).

(3) Base space constraint (BSC) is a constraint on the possible base position. This is expressed by a bounded space in a 3 dimensional space. That is, this constraint requires that (X_0, Y_0, Z_0) be inside $[X_{min}, X_{max}] \times [Y_{min}, Y_{max}] \times [Z_{min}, Z_{max}]$.

Dexterity measure - optimality criterion

The need of dexterity measure is obvious. It is useful when there exist multitude of solutions that satisfy all task/manipulator constraints. Dexterity measures quantize performance and thus can be called "performance index". There are two requirements that must be satisfied by a well defined dexterity measure; physical meaning and scale independence. Without physical meaning, optimality of a designed manipulator is hard to interpret. If it is scale dependent, we may come up with an awkward design like an infinite link length. Dexterity measures that satisfy these two requirements are the relative manipulability, the condition number and the measure of isotropy [7].

DOF selection

In this module, we select the minimum DOF which can perform a given task. In general, 6 DOF is necessary for a spatial positioning

1. The dextrous workspace is a space in which the manipulator's hand can rotate fully about all axes through any point.

and orientating. However, not all tasks require 6 DOF. For example, when a tool at the end-effector has an orientational symmetry as in arc welding and grinding tasks, the minimum DOF becomes five. On the other hand, when more dexterity is required, or there are obstacles in task space, the minimum DOF may become more than six. In this paper, we define the minimum DOF as the number of joints that are *just enough* for a given task. Large DOF may help increase the dexterity at the end-effector, but the control and planning become much more complex.

On the other hand, the minimum DOF cannot be determined independently of other design variables (T, D, P and B). Thus, DOF selection module proposes a candidate minimum DOF for a given task based on the reachability of a given trajectory. When this candidate minimum DOF fails to satisfy all design constraints, we increase the DOF by one. This process is repeated until all design constraints are satisfied.

Type generation

All possible candidate types of manipulators for a selected DOF are generated and fed-forward to an optimization algorithm with unknown dimension, pose and base position. Type generation is based on some heuristic rules that have been applied to almost all existing manipulators. The rules we use in this unit are as follows;

(1) Kinematic simplicity: The kinematic simplicity constrains connection between two adjacent joints. The kinematic simplicity implies that link twist angle (α) is either 0 or $\pi/2$, and link length (a) and joint offset (d) are zero or nonzero and at least one of two is zero. Then, there exist four possible connections between two joints; **1**(α=0, a=nonzero, d=0), **2**($\pi/2$, 0, 0), **3**($\pi/2$, nonzero, 0) and **4**($\pi/2$, 0, nonzero).

(2) Mechanical simplicity [5]: A connection is called mechanically complex when two or more actuators are located at the same place. The rule of mechanical simplicity requires that two actuators at the same place be eliminated because it is hard to be manufactured. In terms of the above four possible connections derived from kinematic simplicity, this can be restated as "Connection **2** must be followed by Connection **4** only". Otherwise, a manipulator is said to be mechanically complex.

Optimization algorithm

This optimization algorithm is supposed to derive one optimal solution that satisfies all design constraints (task and manipulator specifications). We propose an optimization algorithm called "Multi-Population Genetic Algorithm" (MPGA) which is based on an existing Genetic Algorithm (GA), called Simple GA (SGA). Genetic Algorithms are adaptive search techniques based on mechanics of natural genetics, and they are efficient and effective for highly nonlinear problems. Since our goal is to develop an algorithm that is more efficient and effective for our Task Based Design (in other words, we do not develop a general purpose optimization algorithm), we investigate a better use of SGA. To this end, we propose a Boundary Search GA (BSGA) in contrast to the point search of SGA, and a MPGA for Task Based Design.

Our MPGA [8][9] is a parallel implementation of Boundary Search Genetic Algorithm (BSGA) which is an augmented Simple Genetic Algorithm (SGA)[3] with two additional operators called Moving Boundary (MB) and Coarse-To-Fine (CTF). The two operators of BSGA are added to enhance the performance of SGA which becomes slow when the difference among individuals becomes small. Every N generations, MB moves the boundary of search so that the center of the new search space becomes equal to the best individual obtained, and CTF shrinks the boundary toward the center. After application of MB and CTF, BSGA generates new random individuals except two individuals corresponding to the center point. This way, BSGA keep the healthy competition among individuals of which the difference is kept almost constant over the whole generation.

Different from SGA and other GAs, MPGA uses multiple populations. Each population searches an optimal manipulator for one task point and are connected to each other by connecting constraints. Therefore, the complexity of each optimization function is fixed without depending on the number of task points. Each optimization function consists of the objective function and connecting constraints (CC). Constraints in the objective function included RC, DC, JLC and TC, which depend on a task point only; Constraints in the connecting constraints are JIC and JAC which depend on other task points. The cost of the simplicity of MPGA is two additional connecting constraints - link constraint (LC) and base position constraint (BPC). Link constraint enforces link lengths for all optimal manipulators corresponding to all task points to be the same, and the base position constraint makes all optimal manipulators have the same base position. Without these two constraints, our MPGA is simply a collection of BSGAs each of which will design an optimal manipulator for the corresponding task point.

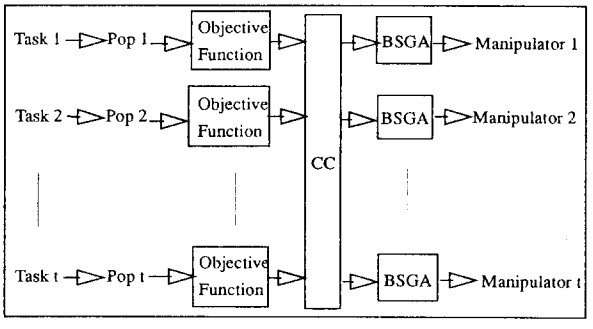


Figure 4: MPGA structure

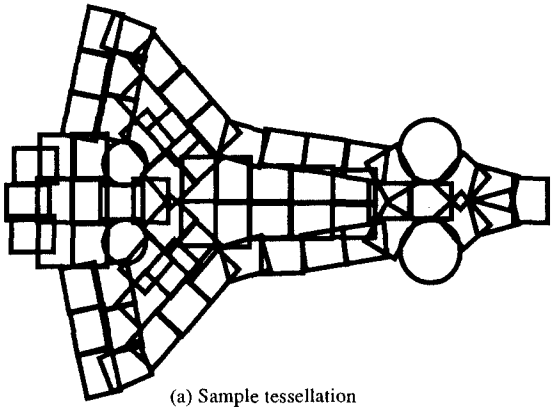
All design constraints except BSC are expressed as penalty functions. BSC is a constraint that can be treated by GA coding directly. It is not necessary to express is as a penalty function. A variable is coded by binary digits inside GA. When we use seven digits for X_0 of the base position, the two extreme values of [000000] and [111111] are mapped onto $[X_{min}, X_{max}]$. This way, BSC is automatically satisfied by the GA coding.

The advantage of MPGA is that the complexity of each BSGA is almost constant regardless of the number of task points. This advantage is obtained by decomposing the complex optimization function into t sub-functions for the t task points. On the contrary, if one optimization function is solved by other search techniques like SGA and simulated annealing, the search space of Task Based Design increases exponentially as the number of variables increases. This implies that with the same effort the quality of the solution decreases rapidly as the search space increases. For our MPGA, as task points increases, only the number of populations increases. Another advantage is that MPGA can provide a general search technique for a manipulator either with or without a prismatic joint by just turning on or off LC and for a manipulator with mobile base by adjusting BPC.

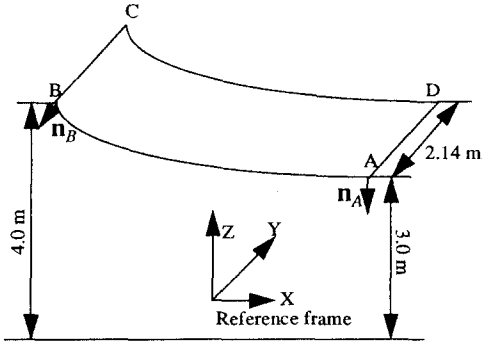
4. AN EXAMPLE OF TASK BASED DESIGN

In this section, we present an example of TBD, in which the relative manipulability [7] is used as a dexterity measure. As mentioned earlier, our Progressive Design consists of three steps (kinematic design, planning and kinematic control). Among these, the kinematic design is the most important because DOF, type, dimension and base position are determined in this step. Because of the limited space, we show only results of the kinematic design.

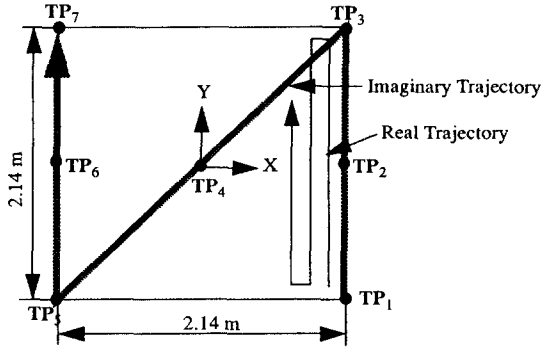
We design an optimal 7 DOF spatial manipulator with a mobile base for space shuttle tile servicing task[2]. The space shuttle base covered with tiles are tessellated into many regular areas as shown in Figure 5(a). One tessellated area corresponds to one movement of a mobile base and includes about 180 tiles. We do not design the mobile base, but design a manipulator with the position of a mobile base as a variable. Figure 5(b) shows an imaginary tessellated area that includes the lowest task point with the lowest slope and the highest task point with the highest slope. Its projection into XY plane and the selected seven task points are shown in Figure 5(c).



(a) Sample tessellation



(b) Task space

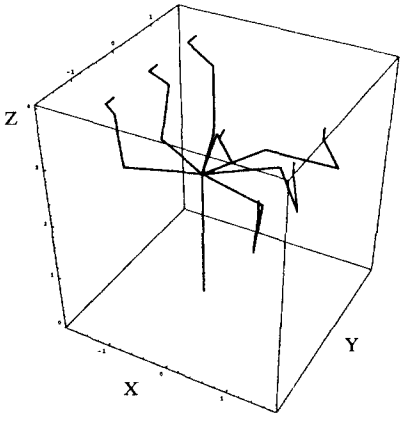


(c) Projection of task space into XY plane

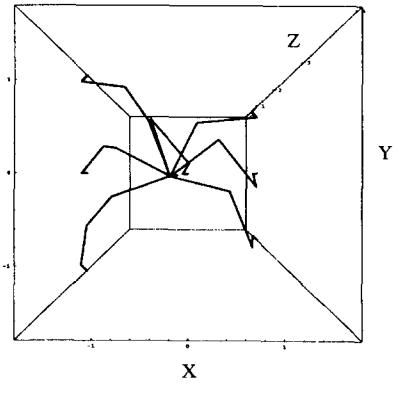
Figure 5: Space shuttle tile servicing task

In this example, we include all design constraints discussed in Section 3. For instance, RC requires that the seven task point are reachable with the **z** direction of the hand coordinate frame orthogonal to the base surface. JC is composed of JIC for the first five joints and JLC for the last two joints because the wrist structure is already determined. They are $JLI(\theta_1) = 350°$, $JLI(\theta_2) = 350°$, $JLI(\theta_3) = 270°$, $JLI(\theta_4) = 270°$ and $JLI(\theta_5) = 270°$, $\theta_6 = [-100°, 100°]$ and $\theta_7 = [-266°, 266°]$, where JLI stands for Joint Limit Interval. TC is composed of three different task constraints; (1) $\det(J_{OO}) > 0.5$ to avoid the singularity of the wrist structure where J_{OO} is the Jacobian matrix of the wrist structure, (2) the static criterion that requires that $\beta > 1$, where $\beta = (\mathbf{u}^T (J_C J_C^T) \mathbf{u})^{-1/2}$ and $\mathbf{u} = [0,0,1]$, to sustain the gravitational force of the end-effector, and (3) the obstacle avoidance that requires a manipulator to move below the shuttle base.

Because of the tool symmetry for the tile servicing task, 5 DOF (3 DOF regional structure and 2 DOF wrist structure) is the minimum DOF for the task. Two additional DOF are required to avoid singularity of the 3 DOF regional structure and to ease alignment of the new and old visual images. Finally, we decide that a 7 DOF spatial manipulator with a 4 DOF regional structure and a 3 DOF spherical wrist is the minimum DOF for the current task. Our type generation module generates 29 candidates of 7 DOF spatial manipulator. Our optimization algorithm evaluates all of these and finally comes up with the optimal manipulator shown in Figure 6. It shows the optimal type (**2-4-3-2-4-2**), dimension, base position and poses at the seven task points. The connectivity of poses are checked in the following steps of planning and kinematic control, but is not shown due to the limited space.



(a) The optimal manipulator



(b) Top view

Figure 6: The optimal 7 DOF manipulator

5. SUMMARY

In this paper, we have introduced a new framework for Task Based Design (TBD) in which we design an optimal manipulator that is most appropriate for a given task. Our research has been motivated mainly by the CMU RMMS and has resulted in a general framework for the design of robot manipulators. This design problem involves very complex, highly nonlinear and implicit functions with a large number of variables which increases with the complexity of a task. Therefore, our framework seeks for a way to reduce the complexity of the design problem by decomposition.

Progressive Design is to decompose the task complexity while the framework of the kinematic design is to decompose the manipulator complexity. The framework of the kinematic design is composed of three inputs (task specification, manipulator specification and dexterity measure) and three modules (DOF selection, type generation and optimization algorithm). The task and manipulator specifications constitute all design constraints, which are divided into two groups (objective function and connecting constraint) depending on the characteristic of the constraint.

Another decomposition can be seen inside of our MPGA, which is a parallel implementation of BSGAs. This decomposition is between decoupling and approximation because there does not exist complete parallelism among task points. That is, we have to connect all BSGAs by introducing connecting constraints. The advantage of MPGA is the efficiency and effectiveness obtained by exploiting the advantages of GAs and the parallelism of task points.

Our approach for Task Based Design has been tested with the example of space shuttle tile servicing task. Extensions to more complex problem domains can be made with a small modification because the framework and optimization algorithm are developed with extensions in mind. For example, the reachability constraint (RC) eliminates the need for inverse kinematics. The link length constraint (LC) and base position constraint (BPC) can be modified to include a link module with prismatic joints and a manipulator with a mobile base. For sub-problems such as path placement and workspace design, our approach can be applied directly by just fixing the related variables. Our approach is not limited to the RMMS, but for design of general manipulators. The example demonstrates the efficiency and effectiveness of our design framework. For more details, refer to [6].

ACKNOWLEDGMENT

This research was funded in part by NASA under grant NAG-1-1075, DOE under grant DE-F902-89ER14042, the Department of Electrical and Computer Engineering, and The Robotics Institute, Carnegie Mellon University.

REFERENCES

[1] Chiu, S., "Task Compatibility of Manipulators Postures", *The International Journal of Robotics Research*, vol.7, No. 5, pp. 13-21, October, 1988.

[2] Dowling K. (Editor), *TPS Robot Design Document*, The Technical Report, The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, fall, 1992.

[3] Goldberg, D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley publishing company, 1989.

[4] Hartenberg, R.S. and Denavit, J., *Kinematic Synthesis of Linkages*, McGraw-Hill Book Company, 1964.

[5] Hollerbach, J.M. "Evaluation of Redundant Manipulators Derived form the PUMA Geometry", Robotics and Manufacturing Automation: The Winter Annual Meeting of the ASME, pp. 187-192, Florida, November, 1985.

[6] Kim, J.-O., *Task Based Design of Robot Manipulators*, Ph.D. thesis, The Robotics Ph.D. program, The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, August, 1992.

[7] Kim, J.-O. and Khosla, P.K., "Dexterity Measures for Design and Control of Manipulators", *IEEE/RSJ int. workshop on Intelligent Robots and Systems (IROS'91)*, Osaka, Japan, November, 1991.

[8] Kim, J.-O. and Khosla, P.K., "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.

[9] Kim, J.-O. and Khosla, P.K., "Task Based Synthesis of Optimal Manipulator Configurations", 1992 Japan-USA Symposium on Flexible Automation - A Pacific Rim Conference- ISCIE, San Francisco, CA, July, 1992.

[10] Li é geois, A., "Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms", IEEE transaction on Systems, Man, and Cybernetics, Vol. SMC-7(12), pp. 868-871, 1977.

[11] Schmitz, D.E., Khosla, P.K., and Kanade, T., "The CMU Reconfigurable Modular Manipulator System", *Proceedings of the 18-th ISIR*, Australia, 1988.

[12] Tsai L.-W. and Morgan A.P., "Solving the Kinematics of the Most General Six- and Five Degree-of-Freedom Manipulators by Continuation Method", J. of Mechanisms. Transmissions, and Automation in Design, Vol. 107, pp. 189-200, June, 1985.

[13] Whitney, D.E., "Resolved Motion Rate Control of Manipulators and Human Prostheses", IEEE Trans. Man-Machine Sys., MMS-10(2), pp. 47-53, 1969.

[14] Yoshikawa, T., "Analysis and Control of Robot Manipulators with Redundancy", Robotics Research: The first int. symp., MIT press, pp. 735-747, 1984.