Design of Space Shuttle Tile Servicing Robot: An Application of Task Based Kinematic Design

Jin-Oh Kim Pradeep K. Khosla

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA, 15213

Abstract

A framework called Task Based Design (TBD)[1] to design an optimal robot manipulator for a given task is proposed. While the ultimate goal is to design all kinematic and dynamic parameters of a manipulator, we consider only kinematic parameters in this paper. Optimal design of a manipulator even for a simple task is difficult because it involves implicit and highly nonlinear functions of many design variables. We design an optimal manipulator which performs a given task best, by using a framework called Progressive Design that decomposes the complexity of the task into three steps: Kinematic Design, Planning and Kinematic Control. As an example problem of TBD, we design a manipulator for space shuttle tile servicing task.

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 [15].

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 provides a special manipulator for a given specific task. For fully exploiting the capability of RMMS, we need a framework for designing manipulators based on a given task specification.

The goal of Task Based Design is to determine the kinematic and dynamic parameters and therefore specify a manipulator to accomplish the given task. In this paper, we describe our methodology for determining the kinematic parameters which include DOF (Degrees of Freedom), Denavit-Hartenberg parameters (type, dimension and pose) and the base position of the manipulator.

To develop a solution (or a manipulator), we pose the problem as an optimization problem. The function that must be optimized (described later in the paper) is highly nonlinear. Further, due to the large number of design variables, the optimization is very complex. In order to reach a solution it is necessary to have a framework that guides the search for an optimal manipulator. Within Task Based Design, we call the framework Progressive Design which decomposes the design process into three steps: Kinematic Design, Planning and Kinematic Control.

The kinematic design problem was formulated and solved as an optimization problem in [13]. In this work, tasks were described by reachability, joint limits, obstacles in the workspace and dexterity. The problem of synthesizing the dynamic parameters was also formulated as an optimization problem and the solution obtained in [11]. The formulation used bounds on the joint velocities and accelerations to choose joint modules with appropriate torque specifications. The kinematic design in [13] was extended to design fault tolerant manipulators in [1].

The approach in this paper is very different from the one in [1][11][13]. We decompose Task Based Design into three steps of optimization in which we progressively include more task description as we move from the first step of Kinematic Design to the last step of Kinematic Control. This decomposition allows us to design more complex manipulator (say, more DOF) as well as more complex task specification due to the reduced complexity of each step. In this paper, our framework for Task Based Design is applied to the design of an optimal manipulator for the space shuttle tile servicing task.

This paper is organized as follows. In Section 2, we describe space shuttle tile servicing task. In Section 3, we define the problem of Task Based Design with assumptions. Next in Section 4, we introduce our design strategy and framework in detail. We present an example of TBD in Section 5. Finally, we summarize the paper in Section 6.

2. Task Description

The space shuttle base is covered with more than 15,000 tiles which need waterproofing before every launch. This tile servicing task is very laborious and tedious, and is dangerous for a human because of the toxic waterproofing chemical. In order to enhance the quality, efficiency and safety, it is necessary to automate this task and therefore requires the design of a robot manipulator with a mobile base.

A manipulator is required to reach any point of the shuttle base with its tool orientation normal to the surface of every tile. According to the task scenario of tile servicing [3], the shuttle base is tessellated into many regular areas as shown in Figure 1(a). One tessellated area corresponds to one movement of a mobile base and includes about 180 tiles.

In this paper, we design the manipulator part for an imaginary tessellated surface that includes extreme design conditions (tiles

1. Henceforth, Task Based Design in this paper refers to Task Based Kinematic Design.

1050-4729/93 $3.00 © 1993 IEEE

with the lowest altitude and the lowest slope along the center line of the base and tiles with the highest altitude and the highest slope at the edge of the base) as shown in Figure 1(b). We do not design the mobile base, but the position of a mobile base is a design variable.

3. Problem statement

Kinematic design variables for robot manipulators include

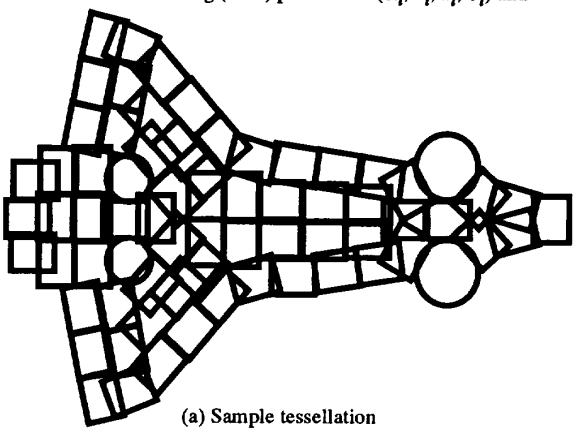
1. Degrees of freedom (N),
2. Denavit-Hartenberg (D-H) parameters (α_i, a_i, d_i, θ_i) and
3. Base position (B),

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-1) sets of (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 determined at the same time. The design problem addressed in this paper is to map the specified task onto the above kinematic parameters.

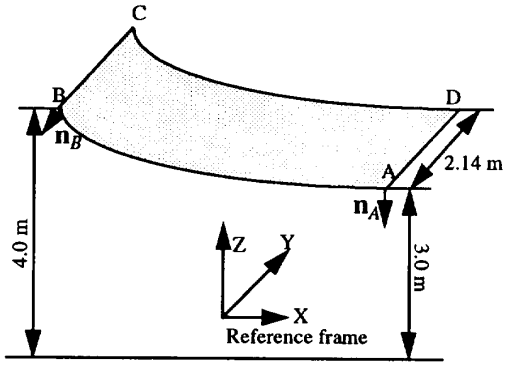
A given task and constraints on link/joint modules are formulated as task specification and manipulator specification, respectively. In other words, the task specification is related to only a given specific task and the manipulator specification is related to the above design variables. For instance, a given trajectory belongs to the task specification and the joint limit of an joint module belong to the manipulator specification. The problem in which the above variables are unknown is 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**.

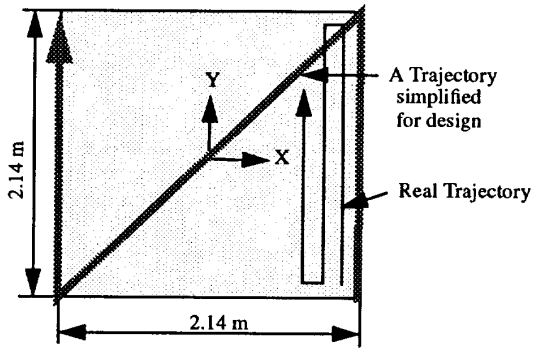
We use a dexterity measure as an optimality criterion to choose one optimal manipulator when there are many solutions that satisfy both task and manipulator specifications. For this problem, design variables are 5 tuples of (N,T,D,P,B). The input to this problem is a triple (**E**, **R**, **C**). A schematic diagram of Task Based Design is shown in Figure 2.



(a) Sample tessellation



(b) Task space



(c) Projection of task space into XY plane

Figure 1: Space shuttle tile servicing task

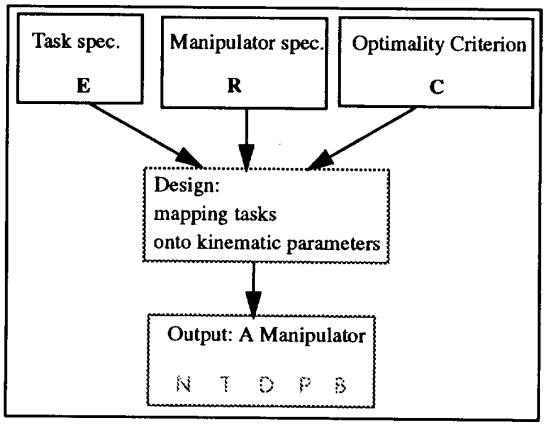


Figure 2: Task Based Design

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.

In addition, we assume that the base position of a manipulator is not movable. A robot for the tile servicing is a manipulator with a mobile base. For one tessellated surface in Figure 1(b), however, the mobile base is supposed to remain stationary.

4. Design Framework

The optimization function of Task Based Design is composed of highly nonlinear, implicit and complex functions and includes a large number of design variables. To derive a good solution, it is necessary to construct a good design framework that decomposes the complexity.

We propose a high level framework called "Progressive Design" that decomposes the design into several steps in which we progressively include 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. For the tile servicing task, an inverted "N" shaped trajectory in Figure 1(c) is used for the design of an optimal manipulator. The first step (Kinematic Design) optimizes all design variables at the seven task points in this example as shown in Figure 3. The second step (Planning) optimizes poses at the five sub-task points between two adjacent task points of Kinematic Design. The total number of sub-task points is equal to 30 (=5x6). Finally, all task points are connected by the third step (Kinematic Control) along the given trajectory.

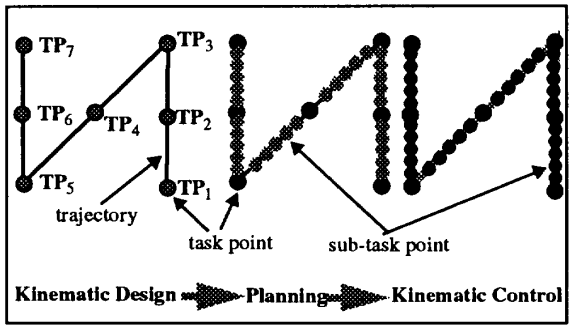


Figure 3: Progressive Design

Each step of Progressive Design is still complex with many variables and nonlinear functions. Kinematic Design is the most complex step because it involves all design variables such as DOF, type, dimension, pose and base position. This requires a framework to decompose the complexity of Kinematic Design.

In 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 for the decomposition is to reduce the complexity of Task Based Design. Due to this decomposition, the number of variables, in an optimization function is reduced by almost half, compared to an optimization function when the type is included as a continuous variable.

Figure 4 shows the framework of Kinematic Design which 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.

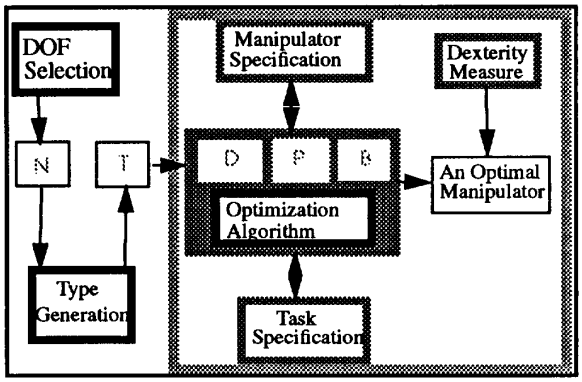


Figure 4:Framework for Kinematic Design

The framework of Planning step is obtained from the framework of 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 Kinematic Design.

The framework of Kinematic Control step is based on the Resolved Motion Rate Control (RMRC)[17]. For redundant manipulators, we use the pseudoinverse approach[12]. Our approach differs from [12][18] in that our Kinematic Control is based on a two point boundary value problem where poses at two adjacent task points become end boundaries. In the following, we describe in detail the three inputs and the three modules of the framework of Kinematic Design.

4.1 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 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.

In fact, 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.

For the tile servicing task, we assume that the wrist structure is a spherical wrist with the hand size equal 20 cm because spherical wrist is most dextrous in terms of singularity avoidance [14]. The tool for tile servicing has an orientational symmetry, but the task needs 3 DOF wrist to ease the visual processing (alignment of old and new images). For the regional (positioning) structure, at least four DOF are necessary considering the singularity cylinder along the axis of the first joint [7]. Therefore, we begin with the 7 DOF manipulator as a candidate minimum DOF.

4.2 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: Kinematic simplicity constrains connection between two adjacent joints. 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 [6]: 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 manufacture. 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.

When the above two rules applied to the 7 DOF spatial manipulator with the 3 DOF spherical wrist, we come up with the 29 candidate types as shown in Figure 5.

4.3 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 task space and workspace are extended and generalized in our design problem. The task specification is generalization of the task space while the manipulator specification in the following subsection is generalization of the workspace. The task specifications that are used in our design are summarized below:.

(1) **Reachability constraint** (RC) must be satisfied over a given trajectory. During Kinematic Design, a manipulator must reach finite number of task points, which are assumed to approximate the given trajectory. Typically, the closest and farthest points from the given base space should be included in the set of task points. Selection of task points affects the performance of a designed manipulator. Since our Kinematic Design based on a set of task points is further investigated in the following steps of Planning and Kinematic Control, a risk that may result in a failure of design due to the selection of task points is eliminated by our interactive design.

The first step of Progressive Design (Kinematic Design) is based on the seven task points in Figure 3; the origin of the hand coordinate system at the tool tip of the end-effector must be at the seven task points, the **z** direction must be orthogonal to the shuttle base surface, and the **y** direction must be parallel with the **Y** direction of the reference frame.

(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[2].

For the tile servicing task, TC is composed of three different task constraints; (a) $\det(J_{OO}) > 0.5$ to avoid the singularity of the wrist structure where J_{OO} is the Jacobian matrix of only the wrist structure, (b) the static criterion that requires that $\beta > 1$, where

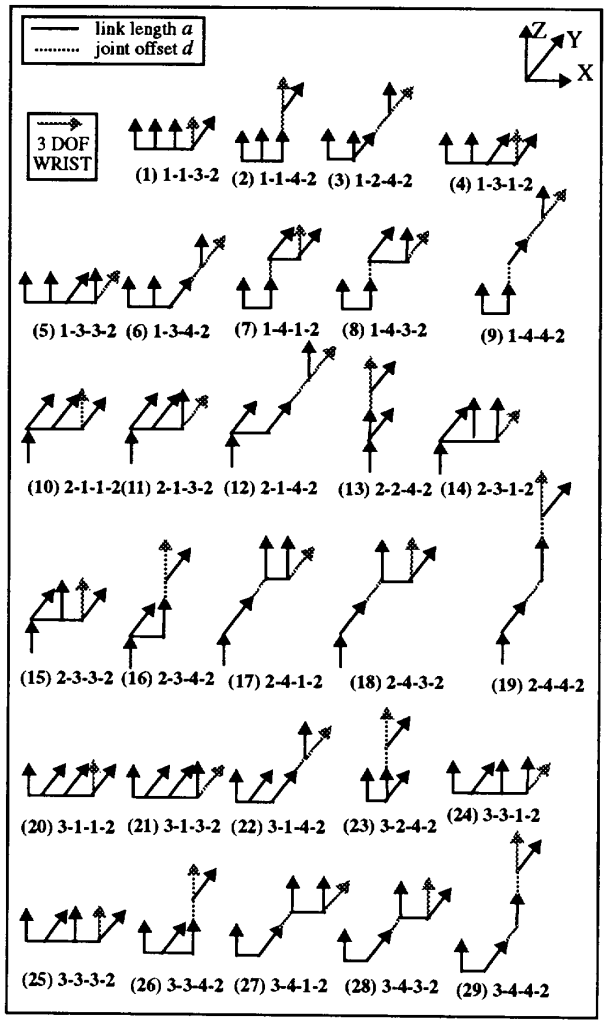


Figure 5: 29 candidate types of manipulators for the tile servicing

$\beta = (\mathbf{u}^T (J_C J_C^T)\, \mathbf{u})^{-1/2}$ and $\mathbf{u} = [0,0,1]$, to sustain the gravitational force at the end-effector, and (c) obstacle avoidance that requires a manipulator to move below the shuttle base.

(3) **Joint angle change constraint** (JAC) is based on the observation that the joint angle change between two adjacent task points must be kept small. This constraint is important to prevent local optima in poses in Kinematic Design. The local optima in poses result from the multiple solutions of inverse kinematics. For example, a 2 DOF planar manipulator has two solutions (elbow up and down) to reach one task point. If one task point is reached by the pose of elbow up and the next task point is reached by the pose of elbow down, a trajectory between these two task points cannot be followed continuously because of the singularity between elbow up and down.

Figure 6 shows four global optima obtained when a 2 DOF planar manipulator is designed without JAC. When JAC is applied, (a) and (c) become local optima, while (b) and (d) remain as global optima. In general, when there are I inverse kinematic solutions and t task points, there are I^t global optima in pose if JAC is not en-

forced. JAC makes I solutions remain as global optima and the remaining become local optima.

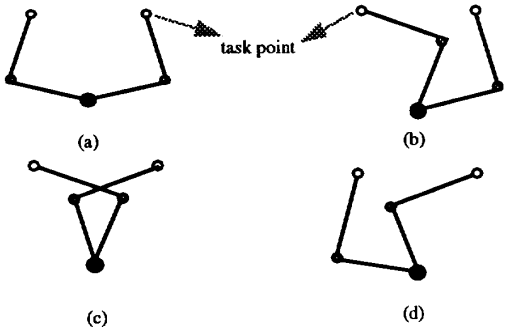


Figure 6: Four global optima in pose for a 2 DOF manipulator

4.4 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 task specification, manipulator specification is expressed in the form of constraints.

(1) **Dimension constraint** (DC) is a constraint on distances l_i between two adjacent joints which are defined as $l_i = \sqrt{a_i^2 + d_i^2}$. In this paper, we require that an inner distance be always longer than an outer distance; $l_i \geq l_j$ when $i < j$. This constraint is based on the following observations; outer links are more appropriate for fine motion and inner links for gross motions, and the dextrous workspace[2] requires outer links to be shorter than inner links. This heuristic constraint may change depending on a given task. Therefore, the use of the above constraint is up to the user, who may propose to use some other form of a dimension constraint like $l_{min} < l_i < l_{max}$ depending on the shortest and the longest link modules of the RMMS.

(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 the 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).

For the tile servicing task, 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.

(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 $[X_{min}, X_{max}]x[Y_{min}, Y_{max}]x[Z_{min}, Z_{max}]$. For the tile servicing, it is [-0.5, 0.5]x[-0.5, 0.5]x[0., 2.5].

4.5 Dexterity measure - optimality criterion

The need for a 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 [8].

We use the relative manipulability for the regional structure only. The relative manipulability for the 4 DOF regional structure is defined as

$$M_r = \frac{\sqrt[3]{\det(J_C J_C^T)}}{l^2} \qquad (1)$$

where J_C is the 3x4 Jacobian matrix of the regional structure and l is the total link length ($l = \sum_{i=1}^{4} \sqrt{a_i^2 + d_i^2}$).

The singularity of a spatial manipulator with a spherical wrist can be decomposed into singularities of regional structure and orientational structure. For redundant manipulators with a 3 DOF wrist, singularity of the regional structure does not necessarily mean singularity of the whole structure. When considering the role of each structure - positioning by the regional structure and orienting by the orientational structure - the singularity of the regional structure is not desirable. The singularity of the Jacobian matrix of the 7 DOF spatial manipulator with a 3 DOF wrist can be decomposed as follows:

Let $A = \{\theta: \det(J) = 0\}$, $B = \{\theta: \det(J_C) = 0\}$ and $C = \{\theta: \det(J_{OO}) = 0\}$ where J is the 6x7 non-uniform[3] Jacobian matrix, J_C the 3x4 uniform wrist Jacobian matrix and J_{OO} the uniform 3x3 Jacobian matrix of the wrist structure. The dimension of all elements of J_C is [length] and that of J_{OO} is non-dimensional. Thus, they are uniform and the determinant of the uniform Jacobian matrix has physical meaning. Then, the singularity A of the 7 DOF spatial manipulator is a subset of the union of the singularity B of the regional structure and the singularity C of the orientational structure. That is, $A \subset (B \cup C)$.

The singularity B can be prevented by defining a dexterity measure for the regional structure. As the dexterity measure, we use the relative manipulability. This relative manipulability is used to design dimension, pose and base position of the regional structure. The constraint that helps avoid the singularity C is formulated as one of the task constraints (TC).

2. The dextrous workspace is a space in which the manipulator's hand can rotate fully about all axes through any point [5].

3. Non-uniform implies that all elements of the Jacobian matrix do not have the same unit. For the 6x7 Jacobian matrix in this example, elements in the upper three rows has the dimension of [length], but the elements of the low three rows are non-dimensional. Dexterity measures such as the manipulability [18] derived from this non-uniform Jacobian matrix does not have any physical meaning, but only can be used to avoid singularities because the singularity is a geometrical feature.

4.6 Optimization algorithm

This optimization algorithm derives 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. Our goal here is to develop an algorithm that is efficient and effective for our Task Based Design (in other words, we do not develop a general purpose optimization algorithm). To this end, we propose a MPGA for Task Based Design that has several Boundary Search GAs (BSGA) in parallel.

Our MPGA [9][10] is a parallel implementation of Boundary Search Genetic Algorithm (BSGA) which is an augmented Simple Genetic Algorithm (SGA)[4] 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 keeps healthy competition among individuals and the difference between individuals 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 constant and does not depend on the number of task points. Each optimization function consists of the objective function and connecting constraints (CC). Constraints in the objective function include 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 length constraint (LC) and base position constraint (BPC). Link length 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.

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. In contrast, 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.

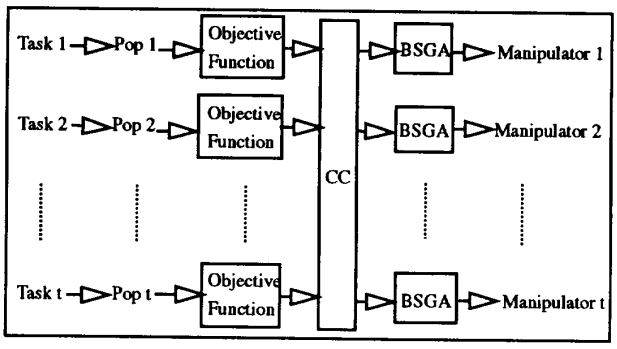


Figure 7: MPGA structure

5. Example: Space Shuttle Tile Servicing Robot

In this section, we present results of TBD for the space shuttle tile servicing task. Because of the limited space, we show only results of Kinematic Design (the first step of Progressive Design).

Our type generation module generates 29 candidates of 7 DOF spatial manipulators. Results of the first step over the 29 types are shown in Figure 8; the relative manipulability M_r, the task constraint (TC), and the reachability constraint (RC). It is very difficult to derive one figure that includes all information necessary to choose the optimal manipulator. Thus, we present results necessary for choosing the optimal manipulator in the three figures. Then, how can we choose the optimal type of manipulator for the given task from the three figures? Each figure gives different information but is used to filter out undesirable candidates. The types of manipulators that survive from the three filtering actions can be said to be the "good" candidates, from which we can choose one.

For the two constraints (TC and RC), the desired value is zero at the seven task points. The small horizontal bar corresponds to one task. To find the good candidates, we start with TC in Figure 8, because the number of surviving types of manipulators is minimum among the three filters (figures). When we accept only types of manipulators with TC < 0.02, the surviving types from the TC requirement are 8, 18, 19 and 21. Next, we use the RC to filter out undesirable types. Among the four types, 18 and 21 survive from the RC filter.

Finally with M_r, we can choose one between 18 and 21. Short horizontal lines for each type of manipulator represent the magnitudes of the relative manipulability at a task point. The vertical line connected to the abscissa ($M_r = 0$) implies that the type of manipulability can hardly satisfy all constraints. We set M_r for a task point equal to zero when the manipulator cannot satisfy any constraints except TC and RC. Candidate 21 is connected to the abscissa, but 18 is not. This implies that 18 is the only surviving type of manipulator from all three filters. In other words, the only good candidate remaining from the three filters is 18 in this example. This solution is an answer to the question "what kind of redundancy helps most to remove singularity?".

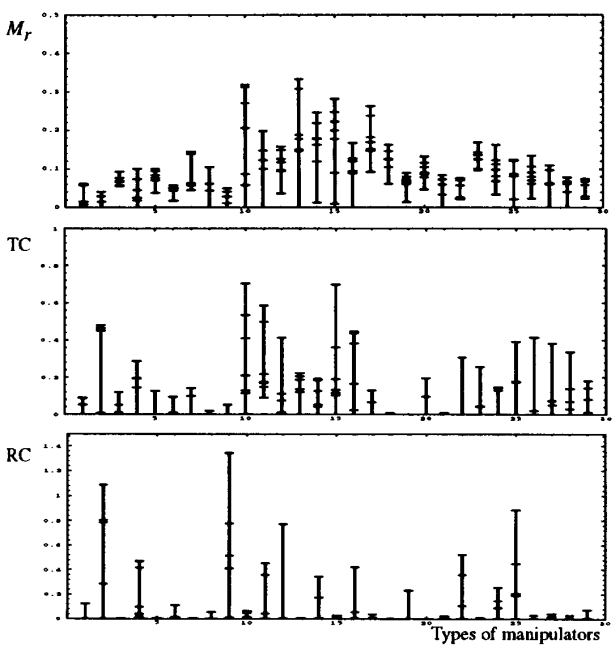


Figure 8. Results of Kinematic Design for 29 candidate types

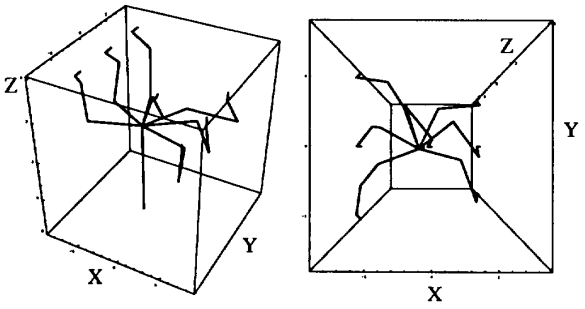
Magnitude of the relative manipulability should not be interpreted as absolute mobility because the magnitude is normalized by the total link length. A larger relative manipulability does not necessarily mean a better mobility at the end-effector, but means a smaller total link length. Thus, a small relative manipulability and large link length can result in a large mobility at the end-effector. Only the relative difference between the maximum and minimum relative manipulabilities is important because it can be used to predict the closeness to singularities [10].

One important observation from this result is that the initial connection has the most significant effect on the performance of the type of manipulator. For types from 1 to 9, the initial connection is **1**. For types from 10 to 19, the initial connection is **2**. For types from 20 to 29, the initial connection is **3**. From the M_r, we can see that types of manipulators with the initial connection equal to **2** are larger relative manipulability than the others. This is because the axis of the second joint for types beginning with **2** passes through the origin of the base coordinate frame. This implies that the distance between the second joint and a task point is longer than the other types. As a result, the velocity transmission ratio due to the second joint of types beginning with **2** is the largest, in general.

Another observation can be made from the TC. Types beginning with Connection **2** are not desirable for our task constraints (TC), whereas the other types satisfy better. The reason for this is the same as why types beginning with **2** have better relative manipulability. The longer distance between the second joint and a task point implies that the second joint has to sustain a larger static torque. Therefore, our TC is a constraint that competes with the DM. For types from 17 to 19, the beginning connection is **2** and the next connection is **4**. The second connection places the second joint in a way that a large static torque is sustained by the structure. The mobility in the Z direction increases, but not as much as the second joints for types from 10 to 16. Thus, the second connection is the second most significant effect on the performance. Among the three from 17 to 19, type 18 is chosen the best for the current task. The optimal type 18 is selected from this competition, or it can be said to be a compromise of two competing criteria.

The optimal manipulator (type = 18) is shown in Figure 9 with poses at the seven task points and is further investigated in the following steps (Planning and Kinematic Control). It has been found that over the whole trajectory all design constraints are satisfied with the optimal manipulator in Figure 9.

The final design of the optimal manipulator for the space shuttle tile servicing is shown in Figure 10. The joint centers for the first five joints whose joint limit intervals are formulated as JIC are another important result that will guide how to assemble joint module. At the opposite side of the joint center, there is a forbidden region of joint limit.



(a) The optimal manipulator (b) Top view

Figure 9: The optimal 7 DOF manipulator

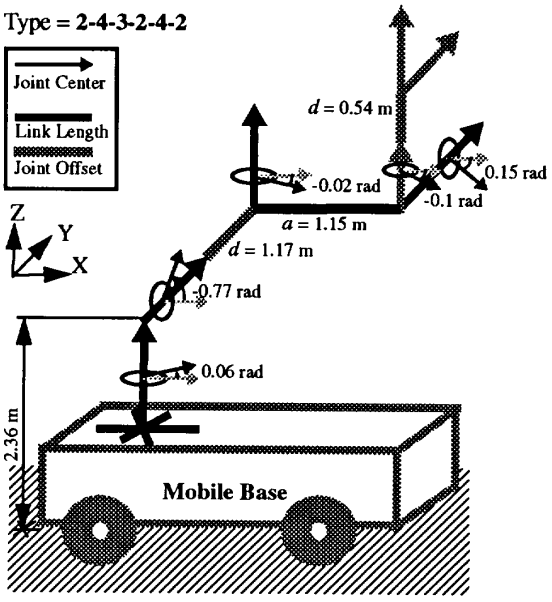


Figure 10: The optimal manipulator designed for tile servicing

6. 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 decomposes complexity into three steps, in which it progressively includes more task description. The framework of 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.

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 link modules 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.

7. ACKNOWLEDGMENT

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

8. REFERENCES

[1] Au, W.K.F., *Fault Tolerant Manipulator Design*, M.S. Thesis, Carnegie Mellon University, Electrical and Computer Engineering Department, May 1992.

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

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

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

[5] Gupta K.C. and Roth B., "Design Considerations for Manipulator Workspace", Transaction of the ASME, J. of Mechanical Design, Vol. 104, pp. 704-711, October, 1982.

[6] 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.

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

[8] 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.

[9] 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.

[10] 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.

[11] Krishnan A. and Khosla P. K. "A Methodology for Determining the Dynamic Configuration of a Reconfigurable Manipulator System", in Proceedings of the 5th Annual Aerospace Applications of AI conference, Dayton, Ohio, October 23-27, 1989.

[12] Li'e 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.

[13] Paredis C.J. and Khosla P.K., "An Approach for Mapping Kinematic Task Specifications into a Manipulator Design", Fifth International Conference on Advanced Robotics, Pisa, Italy, June, 1991.

[14] Paul, R.P. and C.N. Stevenson "Kinematics of Robot Wrists", The int. J. of Robotics Research, 2(1), pp. 31-38, spring, 1983.

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

[16] 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.

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

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