Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems Yokohama, Japan July 26-30, 1993

A Formulation for Task Based Design of Robot Manipulators

Jin-Oh Kim
SECOM Intelligent Systems Lab.
6-1-1, Sakae, Tachikawa
Tokyo, 190, Japan

Pradeep K. Khosla
The Robotics Institute
Carnegie-Mellon University
Pittsburgh, PA, 15213, U.S.A.

The authors discuss

ABSTRACT

How to design a manipulator from a given task ~~is a main theme of this paper~~. While the ultimate goal is to design all kinematic and dynamic parameters of a manipulator, ~~we~~ they consider only kinematic parameters ~~in this paper~~. Optimal design of a manipulator even for a simple task is difficult because it involves a complex task description and constraints on possible manipulators and many design variables that lead to a combinatorial explosion. ~~With an~~ For the example of designing a space shuttle tile servicing robot, ~~we~~ they introduce a formulation of a given task and manipulator constraints ~~in this paper~~.

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. This approach is called Task Based Design (TBD) of a robot manipulator [7][10].

To develop a solution (or a manipulator) in TBD, we pose the problem as an optimization problem. The output of this optimization problem is the kinematic parameters which include DOF (Degrees of Freedom), Denavit-Hartenberg parameters (type, dimension and pose) and the base position of the manipulator. The input to this optimization is composed of task description, constraints on the above kinematic parameters and an optimization criterion. In this paper, a focus is given on the formulation of these input elements.

Our approach of Task Based Design is very different from the one in [1][12][13]. One of important differences is the formulation of input of the optimization problem which is the main issue of this paper. Other differences include the optimization algorithm [9] and the design framework [10]. Our approach is applied to the design of an optimal manipulator for the space shuttle tile servicing task and Other examples can be seen in [9][10]

The importance of formulation of the design input of TBD cannot be overlooked because a wrong formulation will lead to a manipulator which is not appropriate for a given task or may fail to perform it. While the accurate formulation is desirable, its complexity should be manageable with the current technology of optimization as well as some time constraint. Thus, a reasonable task modeling as a compromise between accuracy and complexity is important to develop an optimal solution. In addition, any limitation on kinematic parameters such as joint limit should be clearly defined and correctly formulated before design. Otherwise, a designed manipulator could not be built with existing joint/link modules of the RMMS. Notice that TBD is to design kinematic parameters that match a given task and constraints on kinematic parameters. When multiple solutions exist, an optimality criterion helps to select the best among them.

This paper is organized as follows. In Section 2, we define the problem of TBD with assumptions. In Section 3, our design framework is discussed briefly. In Section 4, we describe the space shuttle tile servicing task. Next in Section 5, DOF Selection and Type Generation from a given task are outlined briefly. In Section 6, constraints on kinematic parameters, as an input of TBD, are described. In Section 7, a dexterity measure as an optimality criterion is explained. In Section 8, an optimization algorithm is explained. In Section 9, all design inputs are expressed mathematically. Design result for the tile servicing comes in Section 10. Finally, we summarize the paper in Section 11.

2. 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. 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 manipula-

0-7803-0823-9/93/$3.00 (C) 1993 IEEE

tor specification is generalization of the workspace.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**), which is focused on in this paper. A schematic diagram of Task Based Design is shown in Figure 1.

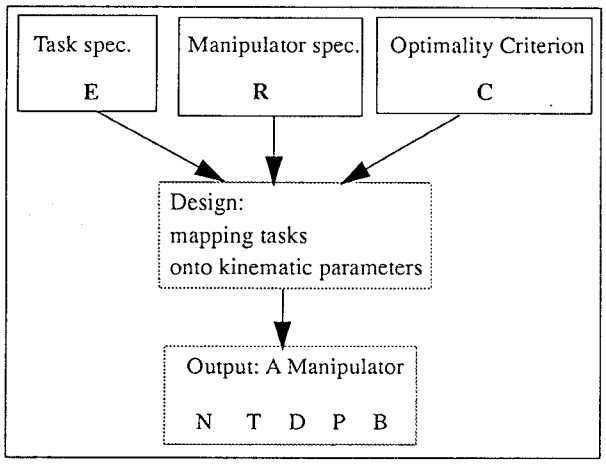


Figure 1: Task Based Design

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.

3. 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 proposed 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. The second step (Planning) optimizes poses at sub-task points between two adjacent task points of Kinematic Design. 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. 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 [10].

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.

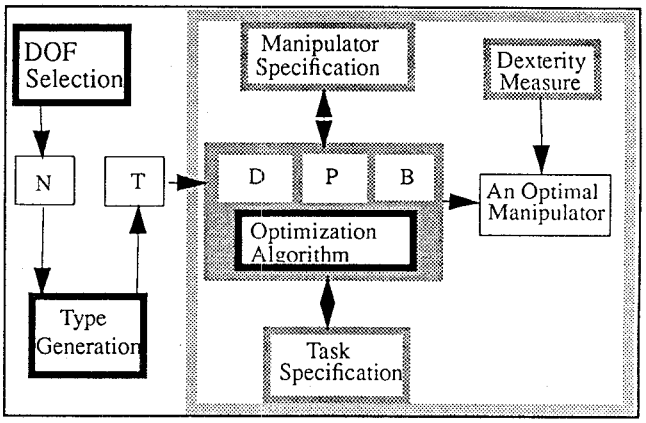


Figure 2: Framework for Kinematic Design

Figure 2 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.

With the above framework, TBD requires a correct but simplified modeling of a given task. In the following section, we discuss task modeling of the space shuttle tile servicing.

4. Task Specification - Space Shuttle Tile Servicing

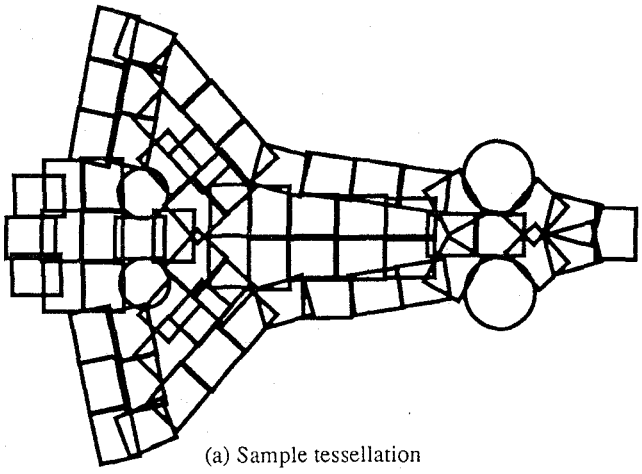
4.1 Task description

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.

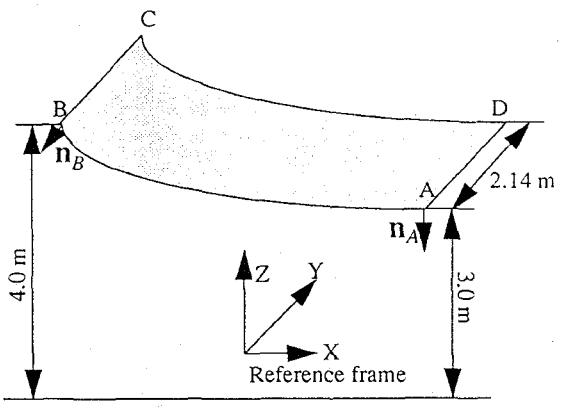
A manipulator is required to reach any point of the shuttle base with its tool orientation normal to the surface of every tile. According to the task scenario of tile servicing [3], the shuttle base is tessellated into many regular areas as shown in Figure 3(a). One tessellated area corresponds to one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

4.2 Task modeling

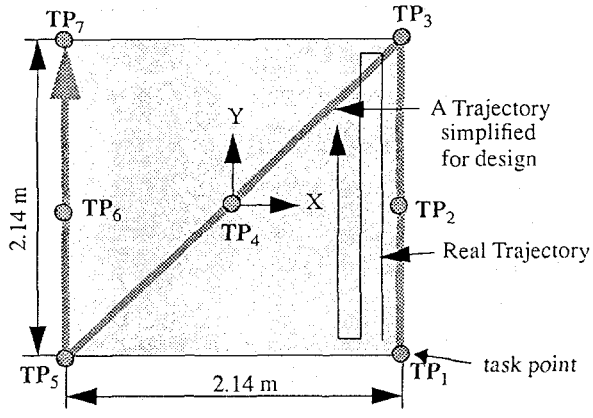
For the space shuttle tile servicing task, three successive task modelings are performed. First, we derive one representative tessellated as shown in Figure 3(b) because the design cannot be performed for all different tessellated areas. This representative and imaginary tessellated surface includes extreme design conditions



(a) Sample tessellation



(b) Task space



(c) Projection of task space into XY plane

Figure 3: Space shuttle tile servicing task

(tiles 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). Second, an inverted "N" shaped trajectory shown in Figure 3(c) is modeled from the complex real trajectory of the tessellated area. Finally, this trajectory is sampled by the seven task points as in Figure 3(c). In fact, design of a manipulator based on the selected task points, not on entire task space, may lead to an incorrect design result. Within the Progressive Design, any incorrect result is fed back to this task modeling to select a new set of (probably more) task points.

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 3(b), however, the mobile base is supposed to remain stationary.

4.3 Task specification

This step is to derive a design input called task specification from the above task modeling of a given task. The task specification for the space shuttle tile servicing is summarized below:

(1) **Reachability constraint** (RC) must be satisfied at all task points. With tile servicing task, Kinematic Design is based on the seven task points in Figure 3(c); 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. A risk that may result in a failure of design due to the selection of task points can be detected by the following steps of Planning and Kinematic Control.

(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.

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 [14], (b) 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 at the end-effector [2], 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.

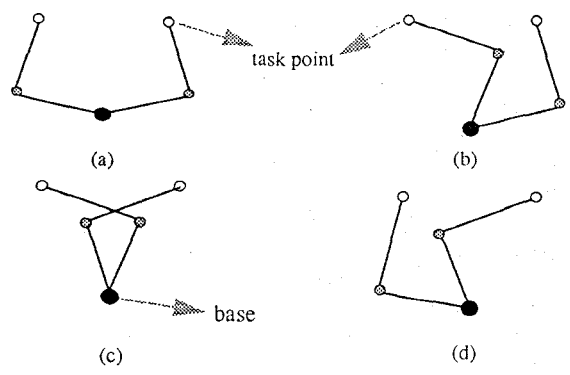


Figure 4: Four global optima in pose for a 2 DOF manipulator

Figure 4 shows four global optima obtained when a 2 DOF planar manipulator for two task points 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 unless JAC is enforced. JAC makes I solutions remain as global optima and the others become local optima.

5. DOF Selection and Type Generation

In the module of DOF Selection, we select the minimum DOF which can perform a given task. Because the minimum DOF cannot be determined independently of other design variables (T, D, P and B), DOF Selection module just 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 inputs (constraints) are satisfied.

For the tile servicing task, we assume that the wrist structure is a spherical wrist with the hand size of 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.

For this candidate DOF manipulator, all possible candidate types of manipulators are generated and fed forward to an optimization algorithm with unknown dimension, pose and base position. Type Generation is based on two heuristic rules. First, *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). Second, *Mechanical simplicity* [6] 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.

The above two rules were applied to the 7 DOF spatial manipulator with the 3 DOF spherical wrist and resulted in the 29 candidate types, all of which were tested to derive an optimal manipulator for the tile servicing task shown in Section 10.

6. Manipulator Specification

Manipulator specification, an extended concept of the workspace of a manipulator, includes all constraints on design variables of dimension D, pose P and base position B. Like task specification, manipulator specification is an important design input for TBD.

(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[1] 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 [θ_{min}, θ_{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 (Joint Limit Interval = JLI) 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(θ_1) = 350°, JLI(θ_2) = 350°, JLI(θ_3) = 270°, JLI(θ_4)= 270° and JLI(θ_5) = 270°, θ_6 = [-100°, 100°] and θ_7 = [-266°, 266°].

(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].

7. 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

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

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

8. 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

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

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.

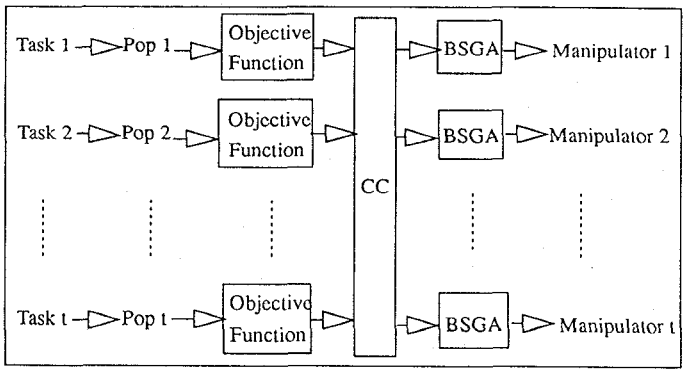


Figure 5: MPGA structure

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.

9. Mathematical Formulation of Design Input

The optimization (fitness) function for the j-th individual of the i-th population is composed of the following functions:

$$^{ij}\mathrm{F} = {}^{ij}\mathrm{DM} + {}^{ij}\mathrm{RC} + {}^{ij}\mathrm{DC} + {}^{ij}\mathrm{TC} + {}^{ij}\mathrm{JLC} + {}^{ij}\mathrm{JIC} + {}^{ij}\mathrm{LC} + {}^{ij}\mathrm{BPC} + {}^{ij}\mathrm{JAC}. \quad (2)$$

9.1 Dexterity Measure (DM)

When ${}^{ij}J_C$ represents for the 3x4 wrist Jacobian matrix the j-th individual of the i-th population, the relative manipulability of the regional structure as a dexterity measure can be expressed as

$$^{ij}\mathrm{DM} = {}^{i}w_{DM} \left(\left(\sqrt[3]{{}^{ij}J_C \, {}^{ij}J_C^T} \right) \Big/ \left(\sum_{k=1}^{4} {}^{ij}l_k \right)^2 \right) \quad (3)$$

where ${}^{ij}l_k$ the k-th link length[2] of the j-th individual of the i-th population, and ${}^{i}w_{DM}$ is the weight of the dexterity measure for the i-th task point. For the following constraints, we apply similar weights without explicit explanation.

9.2 Reachability Constraint (RC)

In general, for a complete spatial motion including 3 DOF for position and 3 DOF for orientation, the 6 dimensional task space can be used. A vector $\mathbf{TP}_i$ is used to describe the i-th 6 dimensional task point. It is composed of a 3 dimensional position vector and a 3 dimensional orientation vector: $\mathbf{TP}_i = [\mathbf{TP}^P{}_i, \mathbf{TP}^O{}_i]^T$. For the orientational task space, we use the Euler angles (α, β, γ), which give the same orientation as ($\alpha+\pi$, $-\beta$, $\gamma-\pi$).

When the i-th task point (or population) is at $\mathbf{TP}_i$, and the end-effector of the j-th individual is at ${}^{ij}\mathbf{E} = [{}^{ij}\mathbf{E}^P, {}^{ij}\mathbf{E}^O]^T$, the penalty for this individual due to RC is

$$^{ij}\mathrm{RC} = -{}^{i}w_{RC}^{P}\,{}^{ij}\mathrm{RC}^P - {}^{i}w_{RC}^{O}\,{}^{ij}\mathrm{RC}^O \quad , \quad (4)$$

where the RCs for position and orientation can be expressed as

$$^{ij}\mathrm{RC}^P = \left\| \mathbf{TP}_i^P - {}^{ij}\mathbf{E}^P \right\| \text{ and} \quad (5)$$

$$^{ij}\mathrm{RC}^O = \min\left(\left\| \mathbf{TP}_i^O - {}^{ij}\mathbf{E}^O(\alpha, \beta, \gamma) \right\|, \left\| \mathbf{TP}_i^O - {}^{ij}\mathbf{E}^O(\alpha+\pi, -\beta, \gamma-\pi) \right\| \right), \quad (6)$$

where $\| \mathbf{x} \|$ is the Euclidean norm.

9.3 Dimension Constraint (DC)

The dimension constraint compares only nonzero links and can be expressed as

$$^{ij}\mathrm{DC} = -{}^{i}w_{HC} \sum_{k=1}^{n-1} \max\left(0, {}^{ij}l_k - {}^{ij}l_{(k+1)}\right). \quad (7)$$

This constraint may be turned off for special tasks, such as obstacle avoidance in which the outer link may be required to be longer than the inner link to avoid obstacles close to the base.

2. Here, link length implies link length a as well as joint offset d of the D-H parameters, assuming at least one of these two is always zero.

9.4 Link Length Constraint (LC)

The link length constraint (LC) compares the k-th link length ${}^{ij}l_k$ of the j-th individual of the i-th population with the average k-th link length of the other populations and can be written as

$$ {}^{ij}\mathrm{LC} = -\,{}^{i}w_{LC} \sum_{k=1}^{n} \left(\sum_{i=1}^{t} {}^{ij}l_k - {}^{i}\bar{l}_k \right) / t \quad , \qquad (8) $$

where ${}^{i}\bar{l}_k = \left(\sum_{j=1}^{m} {}^{ij}l_k \right) / m$ is the average of the k-th link length for the i-th population.

9.5 Base Position Constraint (BPC)

The base position constraint (BPC) compares the base position ${}^{ij}\mathbf{B}$ of the j-th individual of the i-th population with the average base position of the other populations, and can be written as

$$ {}^{ij}\mathrm{BPC} = -\,{}^{i}w_{BPC} \left(\left(\sum_{i=1}^{t} \left\| {}^{ij}\mathbf{B} - {}^{i}\bar{\mathbf{B}} \right\| \right) / t \right), \qquad (9) $$

where ${}^{i}\bar{\mathbf{B}} = \left(\sum_{j=1}^{m} {}^{ij}\mathbf{B} \right) / m$ is the average base position of the i-th population.

9.6 Joint Constraint (JC)

As mentioned earlier, 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. JIC is represented by the Joint Limit Interval (JLI), while JLC is by two absolute values of boundary joint angles.

Joint Limit Constraint (JLC):

The JLC requires all joint angles lie inside of joint limits, which are expressed by two absolute values for each joint. When the maximum ($\theta_{k,max}$) and the minimum ($\theta_{k,min}$) joint limits for the k-th joint are given, JLC can be expressed as

$$ {}^{ij}\mathrm{JLC} = -\,{}^{i}w_{JC} \sum_{k=1}^{n} \left[\max(0,\ {}^{ij}\theta_k - \theta_{k,\,max}) + \max(0,\ \theta_{k,\,min} - {}^{ij}\theta_k) \right] , \qquad (10) $$

where ${}^{ij}\theta_k$ is the k-th joint angle of the j-th individual of the i-th population.

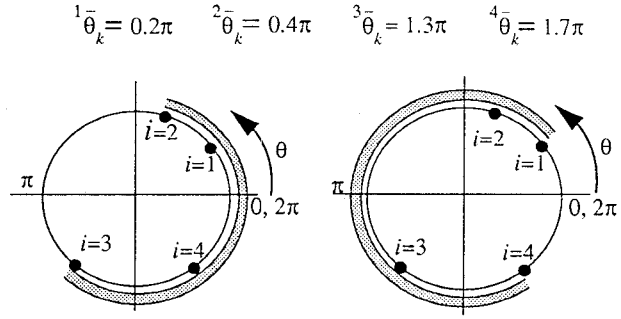
Joint Interval Constraint (JIC):

If a joint has a mechanical limit in its rotation (i.e., less than 2π rad), then only its range interval between two limits has a meaning, but the absolute value of joint angles is meaningless. The JIC in design requires that all joint angles of a joint for all task points must be inside the given joint limit interval (JLI). For instance, suppose that one joint has a limiting interval of π. Then, the joint angle θ may vary from 0 to π or from 0.5π to 1.5π. That is, only its interval matters but the absolute value of θ does not, since the joint limit can be adjusted arbitrarily when we assemble the joint. JIC gives penalty to individuals which are away more than half of JLI from the average k-th joint angle $\bar{\theta}_k$.

For example, let's assume that the joint limit interval JLI_k for the k-th joint is π. For the sake of simplicity, four task points are given (i= 1,2,3,4), and the corresponding average joint angle of the k-th joint for the i-th task point is ${}^{i}\bar{\theta}_k$, which is defined as

$$ {}^{i}\bar{\theta}_k = \left(\sum_{j=1}^{m} {}^{ij}\theta_k \right) / m. \qquad (11) $$

To derive the penalty function, we have to obtain the joint range where all joint angles ${}^{i}\bar{\theta}_k$ lie. Let's assume that four average joint angles are obtained as in Figure 6 where two heuristics of obtaining joint range are applied and compared. First, Minimum Range is based on the minimum joint range in which all average joint angles lie and some penalty will be given to individuals of two populations corresponding to two boundary task points (i=2 and 3). For this joint to reach four task points, the joint has to move (i=1) to (i=2) counter-clockwise, (i=2) to (i=3) clockwise and (i=3) to (i=4) counter-clockwise. This total travelling distance is defined as Connection Length. We have observed that this heuristic can hardly provide a good connection for all task points. In many designs, we found that this heuristic create local minimum problem on joint angles.



(a) Minimum Range (b) Minimum Connection Length

Figure 6: Two different joint range

Our second heuristic is based on the Minimum Connection Length which was found to provide local minimum free joint connection. This heuristic results in totally different result as shown in Figure 6(b). Some penalty is given to individuals of two populations corresponding to two boundary task points (i=1 and 4). For this case, the joint travels from (i=1) to (i=4) counter-clockwise.

Derivation of the Minimum Connection Length and the corresponding penalty function is complex because of periodicity (e.g., $0 = 2\pi = 4\pi$...) of joint angle but won't be explained more due to the limited space. For more detail, refer to [7]. In this paper, assuming that we have derived the joint range corresponding to the Minimum Connection Length and all joint angles transformed to be within the interval of 2π, we can express the penalty function for individuals whose joint angles are outside of joint range larger than JLI:

$$ {}^{ij}\mathrm{JIC} = -\,{}^{i}w_{JLC} \sum_{k=1}^{n} f({}^{ij}\theta_k) \ \text{ where} \qquad (12) $$

$$ f({}^{ij}\theta_k) = \begin{cases} 0 & \text{if } \bar{\theta}_k - \mathrm{JLI}_k/2 < {}^{ij}\theta_k < \bar{\theta}_k + \mathrm{JLI}_k/2 \\ |\,{}^{ij}\theta_k - \bar{\theta}_k| - \mathrm{JLI}_k/2 & \text{otherwise.} \end{cases} $$

9.7 Joint Angle Change Constraint (JAC)

JAC is to prevent a large change of joint angles between two adjacent task points. One example of a large joint angle change is from elbow up to down. In general, JAC is used to make all joint angles be of the same kind among multiple solutions of inverse kinematics.

The JAC for an individual of the i-th task point is based on the average joint angle of the (i-1)-th and (i+1)-th task points. The first order continuity condition among three adjacent task points for the k-th joint angle can be expressed as

$$\frac{d\theta_k}{ds} = 0. \tag{13}$$

This can be interpreted as saying that the joint angle for a task point must be linearly interpolated between two adjacent task points. The corresponding penalty from JAC for a non-cyclic trajectory can be expressed as

$$^{ij}\mathrm{JAC} = -\,{}^{i}w_{JAC} \sum_{k=1}^{n} \left| {}^{ij}\theta_k - {}^{i}\hat{\theta}_k \right| \quad \text{for } i=1,..,t \tag{14}$$

where

$$^{1}\hat{\theta}_k = \frac{(s_1+s_2)\,{}^{2}\bar{\theta}_k - s_1\,{}^{3}\bar{\theta}_k}{s_2} \quad \text{if } i=1, \tag{15}$$

$$^{i}\hat{\theta}_k = \frac{s_i\,{}^{i-1}\bar{\theta}_k + s_{i-1}\,{}^{i+1}\bar{\theta}_k}{s_{i-1}+s_i} \quad \text{if } i=2,...,(t\text{-}1) \text{ and} \tag{16}$$

$$^{t}\hat{\theta}_k = \frac{(s_{t-2}+s_{t-1})\,{}^{t-1}\bar{\theta}_k - s_{t-1}\,{}^{t-2}\bar{\theta}_k}{s_{t-2}} \quad \text{if } i=t \tag{17}$$

Again, as with JIC, the periodicity problem of joint angles has to be solved when we compare ${}^{ij}\theta_k$ and ${}^{i}\hat{\theta}_k$ in (14).

9.8 Task Constraint (TC)

Any task that can be expressed using dimension, pose and base position as variables can be included as a TC. For the tile servicing task, three task constraints are used; (a) $|\det(J_{OO})| > 0.5$ to avoid the singularity of the wrist structure, (b) 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 at the end-effector, and (c) obstacle avoidance that requires a manipulator to move below the shuttle base.

When ${}^{ij}J_{OO}$ is the i-th population, j-th individual Jacobian matrix of the orientational structure, the corresponding penalty can be given as

$$^{ij}\mathrm{TC}_1 = \begin{cases} 0 & \text{if } \left|\det({}^{ij}J_{OO})\right| > 0.5 \\ {}^{i}w_1 \left(\left|\det({}^{ij}J_{OO})\right| - 0.5\right) & \text{otherwise} \end{cases} \tag{18}$$

When ${}^{ij}\beta$ is the force transmission ratio, the corresponding penalty can be expressed as

$$^{ij}\mathrm{TC}_2 = \begin{cases} 0 & \text{if } {}^{ij}\beta > 1.0 \\ {}^{i}w_2 \left({}^{ij}\beta - 1.0\right) & \text{otherwise} \end{cases} \tag{19}$$

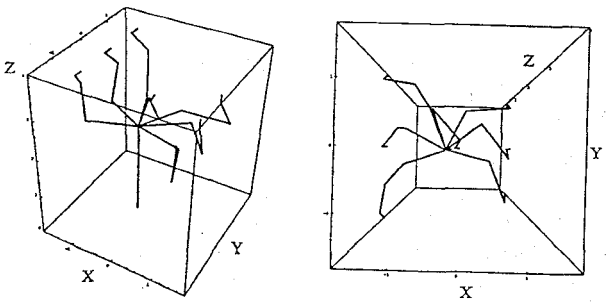
The third task constraint needs a little complex computation. This checks if any part of a manipulator corresponding to the i-th population, j-th individual collides with the surface of the space shuttle. If the manipulator penetrates into the surface, the third penalty function ${}^{ij}\mathrm{TC}_3$ proportional to the normal distance of penetration will be added. Therefore, the total penalty from three task constraints is a summation;

$$^{ij}\mathrm{TC} = {}^{ij}\mathrm{TC}_1 + {}^{ij}\mathrm{TC}_2 + {}^{ij}\mathrm{TC}_3 \tag{20}$$

For other task constraints, refer to [7].

10. Design Result

Kinematic Design was performed for 29 candidate types with 7 DOF. Type **2-4-3-2-4-2** was found optimal after Kinematic Design. This optimal manipulator at the seven task points is shown in Figure 7. The dimension of the optimal manipulator is shown in Figure 8. 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 modules. At the opposite side of the joint center, there is a forbidden region of joint limit. This optimal manipulator was further tested at sub-task points in the step of Planning and again on the whole trajectory in the step of Kinematic Control, and was found to perform the given task successfully. Due to the limited space, these results are not presented. Refer to [7] for more detail.



(a) The optimal manipulator (b) Top view

Figure 7: The optimal 7 DOF manipulator

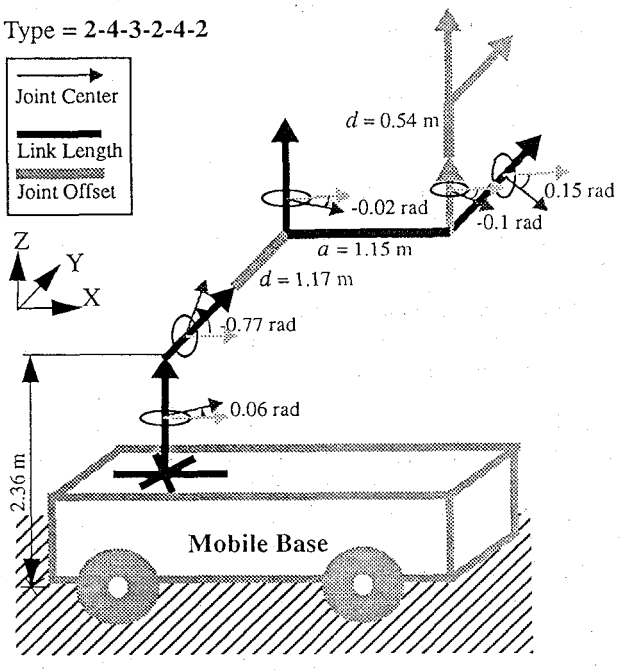


Figure 8: The optimal manipulator designed for tile servicing

11. Summary

In this paper, we have introduced a formulation together with a new framework for Task Based Design (TBD) in which we design an optimal manipulator that is optimal 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 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 and are discussed in detail in this paper. The correct formulation of these constraints from task and manipulator plays the crucial role for the success of Task Based Design. We believe that our formulation is one of many correct approaches and hope to encourage further research in the field of Task Based Design.

Our approach for Task Based Design has been tested with an example of the space shuttle tile servicing task. The result demonstrates the efficiency and effectiveness of our design framework and the our formulation of all design inputs. Extensions to more complex problem domains can be made with a small modification because the framework and optimization algorithm are developed considering its extension. Our approach is not limited to the RMMS, but for design of general manipulators.

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.

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] Kim, J.-O. and Khosla, P.K., "Design of Space Shuttle Tile Servicing Robot: An Application of Task Based Kinematic Design", 1993 IEEE int. conf. on Robotics and Automation, Atlanta, GA, May, 1993.

[12] 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.

[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.