Proceedings of the 6th International Symposium on Robotics Research (ISRR'93).
Hidden Valley, PA, Oct. 2–5, 1993.

Synthesis Methodology for Task Based Reconfiguration of Modular Manipulator Systems

Christiaan J.J. Paredis[*] Pradeep K. Khosla[†]

Department of Electrical and Computer Engineering and
The Robotics Institute,
Carnegie Mellon University,
Pittsburgh, PA 15213

Abstract: *In this paper, we deal with two important issues in relation to modular reconfigurable manipulators, namely, the determination of the modular assembly configuration optimally suited to perform a specific task and the synthesis of fault tolerant systems. We present a numerical approach yielding an assembly configuration that satisfies four kinematic task requirements: reachability, joint limits, obstacle avoidance and measure of isotropy. Further, because fault tolerance is a must in critical missions that may involve high costs if the mission were to fail due to a failure in the manipulator system, we address the property of fault tolerance in more detail. Initially, no joint limits are considered, in which case we prove the existence of fault tolerant manipulators and develop an analysis tool to determine the fault tolerant work space. We also derive design templates for spatial fault tolerant manipulators. When joint limits are introduced, analytic solutions become infeasible but instead a numerical solution procedure can be used, as is illustrated through an example.*

1 Introduction

Conventional (serial or parallel link) manipulators are often considered to be general-purpose and flexible systems. Unfortunately, these systems are not general purpose. In order to understand this, consider a computer which is a general purpose computing engine if it can compute a computable function. Following a similar logic, a manipulator will be general purpose if it could do a "doable" task. In defining a general purpose manipulator, one has, of course, to define a "doable" task first. For the time being, let us avoid this open issue and consider two tasks that two different manipulators can perform, but that cannot be performed by either manipulator separately. If this is the case, then one may conclude that none of the above two manipulators are general purpose. So if one has to define a general purpose manipulator, then one has first to define a criterion for "doable" tasks (like "doability"). Such a definition may lead to the development of models of "doability" (like computability) and maybe to Turing-like machine models of manipulators. While such a development would certainly do a lot for advancing the state-of-the-art in manipulators, it is not our intention to address this general problem.

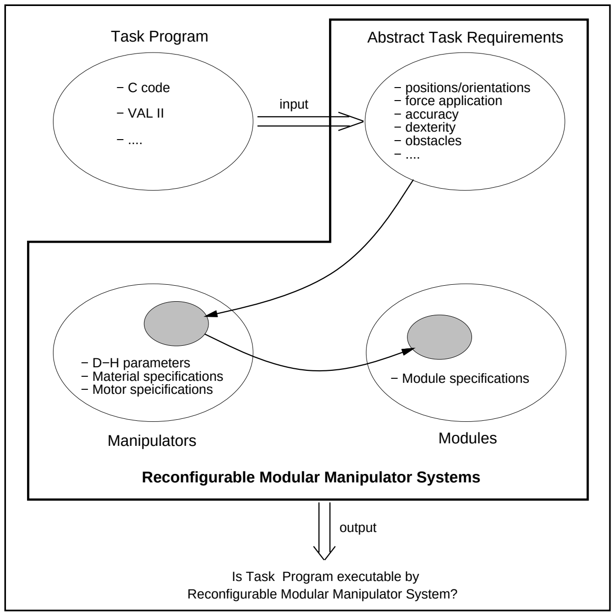

Figure 1: Definition of a general purpose manipulator.

In order to make the problem tractable, let us define a set of tasks that we would like to perform with a manipulator. Let us also define a set of basic modules (consisting of joints and links) that we may combine to create various manipulators. Finally, let us assume the existence of a methodology that will accept a task (in the form of a program or as a set of requirements) as input and find a manipulator that can be created from the given set of modules to perform the task. This scenario is described in Figure 1, and it allows us to put forth one possible definition of a general-purpose manipulator.

General-purpose Manipulator: If for every task in the set of tasks, it is possible to find a manipulator that can be created from the given set of modules to do the task, then we define the system of modules (or the system of all possible manipulators) as general purpose with respect to the set of tasks. We will call such a system a Reconfigurable Modular Manipulator System (RMMS).

Note that the above definition does not require us to define

[*]Department of Electrical and Computer Engineering and The Robotics Institute.
[†]Associate Professor, Department of Electrical and Computer Engineering and The Robotics Institute.

a set of all possible "doable" tasks nor does it require us to define the criteria for determining "doability" even though that is the ultimate goal of our research.

Our past work has addressed the development of the modules and the technology for the RMMS [29]. The RMMS has many potential applications in both hazardous and industrial environments. It puts forth the idea of designing a specific manipulator for a task and also the notion of the user writing device (or manipulator) independent code. The RMMS raises several theoretical issues and it is our aim to address one of these in this paper. Specifically, we describe a design methodology that accepts a task specification as its input, determines a kinematic configuration of the desired manipulator and selects the modules to create this manipulator.

In order to support the current practice of picking the best configuration amongst available robots, several *expert systems* have been built to aid the user or the applications development engineer [23]. A straightforward extension of this selection process has been the inclusion of the design of new manipulators, optimally suited for a specific application [1, 24]. A totally different approach to the robot design problem finds its roots in *simulation*. A variety of commercial robot simulation packages are currently available [6, 30], providing designers with convenient tools to quickly check the implications of different design decisions. In general, however, these simulation packages still require a human to make the design decisions. Finally, a third way of dealing with the problem of robot design, has grown out of the field of *mechanism design* [21, 27]. Unlike the rule based expert systems, these programs are algorithmic in nature. Commonly, the design process is subdivided in two stages: form synthesis and dimensional synthesis. The first stage is usually performed by searching over the set of feasible mechanism types, while the second stage consists of optimizing the set of dimensional parameters.

The approach we propose in this paper differs from the methods listed above, because we are specifically interested in *modular manipulators*. The interest in modular manipulators has grown steadily over the last decade [9, 32], and several related research issues have been addressed [2, 3, 12, 18, 22, 26]. However, the problem of determining the modular configuration optimally suited for one specific task, has never been addressed before to the best of our knowledge. In this paper, we investigate modular design from *kinematic* task requirements. These requirements affect only the kinematic structure of the manipulator, while *dynamic* requirements affect both its kinematic and dynamic structure. Examples of kinematic requirements are work space volume, maximum reach, and maximum positional error. Examples of dynamic requirements are maximum payload, maximum joint velocities, and maximum joint accelerations. Just as task requirements can be classified as kinematic or dynamic requirements, the design procedure can also be split into two parts: *kinematic* design and *dynamic* design [18]. Kinematic design determines the kinematic structure of the manipulator, while dynamic design determines the dynamic configuration. However, the dynamic design may require a change in kinematic structure, and thus a few iterations may be necessary to find a manipulator that satisfies both kinematic and dynamic requirements.

In the first part of this paper, we only consider reachability, joint limit, obstacle avoidance, and measure of isotropy requirements. A numerical procedure is proposed which determines a modular assembly configuration that meets all the requirements. In the second part, we focus our attention on one additional requirement, namely, fault tolerance. Recently, fault tolerant (or failure tolerant) robotics has been the subject of several publications [19, 31], in which different aspects of the problem are addressed. Visinsky et al. [31] propose a framework to include *failure detection* in fault tolerant robot systems. Lewis and Maciejewski [19], on the other hand, discuss the importance of the *controller* and the redundancy resolution. In this paper, the stress is on *design* of fault tolerant manipulators. We define fault tolerance as the ability to continue the performance of a task even after immobilization of a joint due to failure. Several properties of fault tolerant manipulators are discussed and in Section 5, it is shown how the task requirement of fault tolerance can be included in the numerical design procedure described in Section 2

2 Kinematic Design: Preliminary Results

2.1 Problem Statement

The problem solved in this section is the determination of a modular assembly configuration, that satisfies all the kinematic task requirements. These requirements are that the manipulator must be able to reach a specified set of positions/orientations, $\mathbf{p}_j$, (*reachability requirement*), without violating the motion constraints of the joint modules (*joint limit requirement*), and without colliding with any parallelepiped-shaped obstacles in the workspace (*obstacle avoidance requirement*). Moreover, at the positions/orientations, $\mathbf{p}_j$, the measure of isotropy, which is defined in Section 5.2, must be larger than a user specified minimum (*measure of isotropy requirement*).

In Section 2.2 and 2.3, we develop a numerical procedure to solve this design problem. To facilitate the implementation of our approach, we consider six types of modules, as shown in Figure 2. The choice of these specific modules guarantees a simple conversion from the module dimensions and orientations into the Denavit-Hartenberg (D-H) parameters of the resulting manipulator (A set of 3 D-H parameters per degree-of-freedom, determines unambiguously

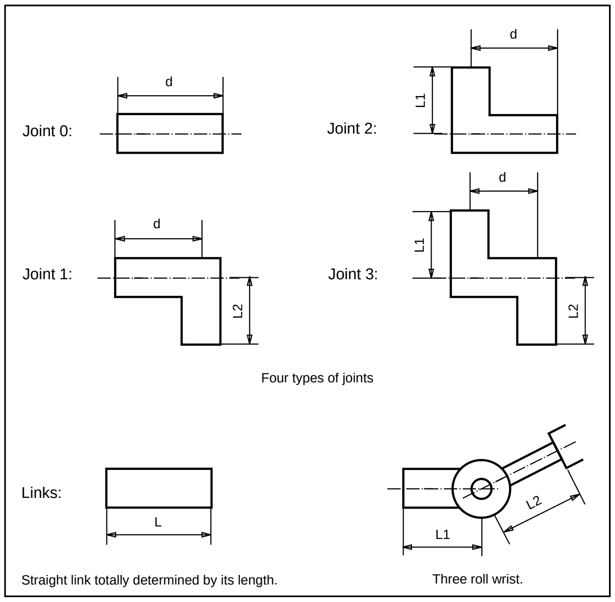

Figure 2: The manipulator module types considered in the design procedure.

the kinematic structure of any serial link manipulator). It has been shown by Kelmar and Khosla [12] that this conversion can be achieved for modules of arbitrary geometry. The actual number of different modules considered for the design can be far larger than six, due to variations in the parametrized dimensions and our design method is general enough to allow for this.

We also require that the robot base be fixed and known, that the first joint module be of type 0 or 1 (i.e. the first joint axis is vertical), and that the last module be a wrist with three axes intersecting at a point. These restrictions result from our implementation of the inverse kinematics and can be relaxed by using iterative solutions to the inverse kinematics problem, as proposed in [2]. Also, the requirement that the robot base be fixed and known, can be relaxed as was shown by Kim [15], who addressed the problem of kinematic synthesis and base position synthesis simultaneously.

Finally, we would like to point out that this design problem can possibly have more than one solution. Consider the design of a 2-DOF planar manipulator, with link lengths L_1 and L_2, satisfying the task requirement that the manipulator should be able to reach a point located behind an obstacle without violating the joint limits, as is illustrated in Figure 3. The region of the (L_1, L_2) -plane containing the solutions is bounded by the curves labeled c, d and e. All the manipulators inside this region satisfy all the design requirements and, therefore, are all *equally good* with respect to these requirements.

2.2 Solution Approach

In this section, we evaluate different approaches to the problem of determining the modular configuration, given some kinematic task specifications. The problem can be interpreted as a *mapping* from task specifications into constraints in the modular configuration space, as is shown in Figure 1. This mapping is nontrivial due to the highly nonlinear character of the kinematic relations and due to the complexity of the task specifications. Krishnan [18], therefore, suggested to solve the inverse problem first, namely, to analyze which task requirements are satisfied by a given modular configuration. This information is stored in lookup tables, which can then be used in a search procedure. One obvious disadvantage to this approach is the combinatorial explosion in the number of different configurations. Let the number of different modules available be N, and let R be the number of relative orientations in which one module can be mounted on the previous module. The total number of configurations that can be obtained from this set of mod-

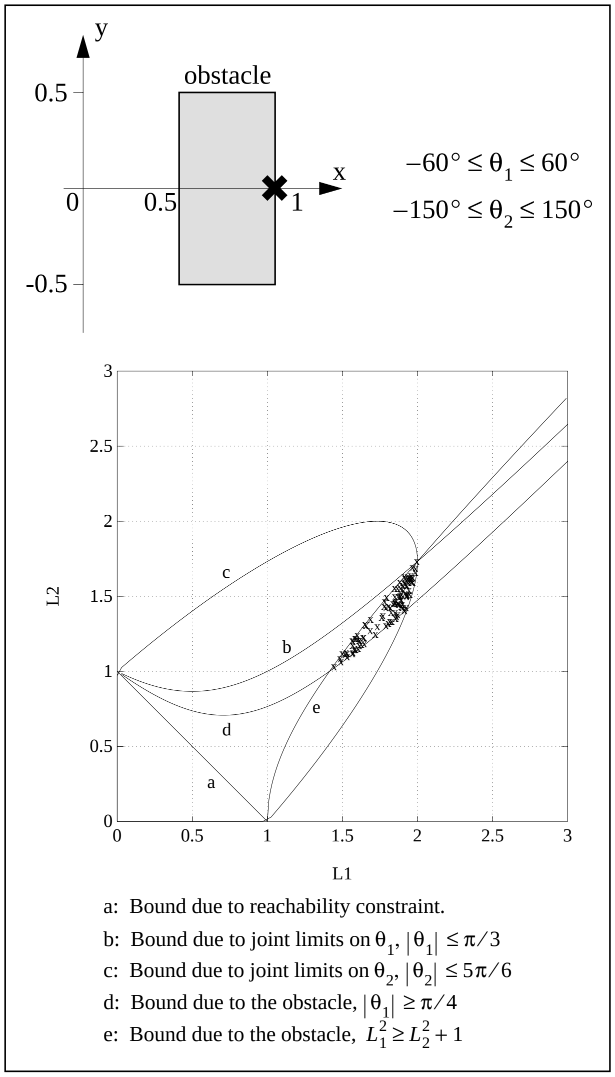

Figure 3: A two-DOF planar design example with solution.

ules is:

$$Num = \sum_{i=1}^{N} R^{(i-1)} \frac{N!}{(N-i)!} \tag{1}$$

This approach would therefore require a very large amount of memory storage for the lookup tables. Also, adding a new module requires that all the lookup tables be updated.

A different approach is to first design a manipulator defined by a set of continuously varying D-H parameters, a point of the D-H configuration space, as proposed in [25], and then transform this design into a modular configuration. The main problem here is the discretization of the continuous solution. As is known from integer programming, simply taking the discrete configuration nearest to the continuous solution might result in an infeasible solution. Therefore, we suggest working directly in the modular configuration space. Of course, an exhaustive search in this space suffers from combinatorial explosion in much the same way the look up table approach does. However, the efficiency of the search procedure can be improved drastically by 'guiding' the search to the most promising regions of the search space. Instead of answering the question whether a certain modular design meets all the task requirements with a simple 'yes' or 'no', we estimate the 'goodness' or 'badness' of the design, i.e., "How far are we away from a solution?" Guiding the search then means focusing the search effort on directions of decreasing 'badness'. This approach is usually referred to as a *heuristic* search technique [28], because in general, it is impossible to compute the 'badness', or the distance to the nearest solution, exactly. The heuristic function only estimates this distance so that it is possible that, locally, the heuristic decreases even though the actual distance to a solution increases. This corresponds to a local minimum in optimization terminology. To overcome this inadequacy, we have to employ a search method that allows for local hill climbing, such as *simulated annealing*.

Simulated annealing was first proposed by Kirkpatrick [16] as a combinatorial optimization algorithm. The method is a random iterative improvement algorithm with the modification that, under certain conditions, an increase in the heuristic function is accepted (In order to be compatible with the standard terminology in discussions of simulated annealing, we use the term objective function instead of heuristic function, henceforth). A new trial configuration is generated randomly in the neighborhood of the current configuration. The condition for acceptance of this trial configuration is:

$$\begin{cases} \Delta F_{obj} \le 0 & \Rightarrow \text{accept} \\ \exp(-\Delta F_{obj}/T) > \text{random}\,[0, 1\,) & \Rightarrow \text{accept} \end{cases} \tag{2}$$

which depends on a control variable, T, the temperature. The algorithm is started at a high temperature for which most new configurations are accepted. After each iteration the temperature is decreased until no new acceptable configuration can be found. The search is then *frozen*. We adapted this basic algorithm slightly to include the special properties of our objective function. In particular, the algorithm is stopped when a new trial configuration has an objective function value equal to zero, even if the search is not yet frozen. We know that a configuration with a 'badness' of zero satisfies all the design requirements.

2.3 Computation of the Objective Function

The goal in this section is to find an objective function which is zero when all the design specifications are satisfied and which is otherwise proportional to the amount of violation of these specifications. Minimizing the objective function by simulated annealing corresponds then to a search, guided towards the most promising regions of the search space. For a given modular configuration, the corresponding objective function can also be interpreted as a *penalty* for violating certain task specifications. The goal of the search is then to find a configuration with zero penalty.

We now propose a methodology for constructing a penalty function. Let us first define some terminology. A *configuration* is the set of D-H parameters which determines unambiguously the kinematic structure of a modular manipulator configuration. A *posture* is the position of a manipulator corresponding to a specific set of joint angles. A *task point* is a specified position/orientation of the end effector that the manipulator must be able to reach without violating the other task requirements.

By taking a closer look at the task requirements, one notices that all the requirements are defined for a specific configuration in a specific posture reaching for a specific task point. The penalty for such a posture should be defined such that, if any single requirement is not satisfied, the penalty for the posture is positive. This can be achieved by defining a non-negative penalty for each task requirement, as described in [25], and summing these penalties for a posture:

$$P_{post} = \sum_{requirements} P_{req} \tag{3}$$

The task penalty is now defined as the minimum over all the posture penalties, so that it is zero when all the task requirements are satisfied for at least one posture:

$$P_{task} = \min_{post} P_{post} = \min_{post} \Big(\sum_{req} P_{req}\Big) \tag{4}$$

Finally, the total penalty of a manipulator configuration is given by the sum of all the task penalties:

$$P_{conf} = \sum_{tasks} P_{task} = \sum_{tasks} \Big(\min_{post} \Big(\sum_{req} P_{req}\Big)\Big) \quad . \tag{5}$$

2.4 Example

The example solved in this section is the design of a seven degree-of-freedom manipulator that is able to reach eight different task points, located in an environment which includes five obstacles. The joint modules have a limited motion range and the task points must be reached in a posture with a measure of isotropy of at least 0.6. It is also required that the manipulator consists of a subset of the twenty different modules (4 joints, 1 wrist, 16 links: we also consider a link of length zero). It is assumed, at this point, that an unlimited number of each type of module is available, so that a design which includes the same module type several times is acceptable. All the task requirements are summarized in the input file in Figure 4.

```
7                           # number of degrees of freedom
3                           # number of dimensions

8                           # number of relative orientations
16                          # number of link modules
#number       |length
#---------------------
0             0.0
1             0.1
2             0.2
3             0.3
4             0.4
5             0.5
6             0.6
7             0.7
8             0.8
9             0.9
10            1.0
11            1.1
12            1.2
13            1.3
14            1.4
15            1.5

4                           # number of joint modules
#number|type  |l1    |d     |l2    |th_min|th_max
#-------------------------------------------------
0      0      0.0    0.1    0.0    -150.0 150.0
1      1      0.1    0.1    0.0    -150.0 150.0
2      2      0.1    0.1    0.1    -150.0 150.0
3      3      0.1    0.1    0.1    -150.0 150.0

1                           # number of wrist modules
#number|l1    |l2    |min1  |max1  |min2  |max2  |min3  |max3
#-------------------------------------------------------------
-
0      0.1    0.05   -100.0 100.0  -150.0 150.0  -266.0 266.0


0.6                         # mi_min: min measure of isotropy.
8                           # num_points: number of points
#xpos         |ypos         |zpos  |Xrot  |Yrot  |Zrot
#-------------------------------------------------------
0.5           0.5           1.0    0.     0.     90.
0.5           0.0           0.5    0.     90.    0.
0.5           0.0           1.5    0.     -90.   0.
0.5           -0.5          1.0    0.     0.     -90.
1.5           0.5           1.0    0.     0.     90.
1.5           0.0           0.5    0.     90.    0.
1.5           0.0           1.5    0.     -90.   0.
1.5           -0.5          1.0    0.     0.     -90.

5                           # num_obst: number of obstacles.
#xpos  |ypos  |zpos  |Xrot  |Yrot  |Zrot  |xdim  |ydim  |zdim
#-------------------------------------------------------------
-
2.     -2.    0.425  0.     0.     0.     3.7    3.7    0.85
2.     2.     0.425  0.     0.     0.     3.7    3.7    0.85
2.     -2.    1.425  0.     0.     0.     3.7    3.7    0.85
```

Figure 4: The input file of the 7-DOF example.

	link#	angle#	joint#
1	10	4	1
2	14	4	3
3	3	3	0
4	12	6	3
5	14	0	—

Table 1: Module numbers of 7-DOF design.

dof	d_i	a_i	α_i
1	1.1	1.6	180˚
2	0.1	0.0	90˚
3	1.8	0.0	–90˚
4	0.1	0.0	90˚
5	1.6	0.0	90˚
6	0.0	0.0	-90˚
7	0.05	0.0	—

Table 2: D-H parameters of 7-DOF design.

A quick calculation gives us an idea of the extent of the search space. A 7-DOF manipulator consists of five links, four joints and one wrist, and is further determined by five angles, specifying the relative orientations of the joint modules. Taking into account the restrictions, that the first joint module must be of type zero or one and the last module must be a wrist, the number of configurations in the search space equals:

$$\begin{aligned}&(\#\text{links})^5(\#\text{joints})^3(\#\text{joints of type 0 or 1})\\ &\quad(\#\text{wrists})(\#\text{rel. orient.})^5 \qquad (6)\\ &= 16^5\cdot 4^3\cdot 2\cdot 1\cdot 8^5 \approx 4.4\times 10^{12}\end{aligned}$$

Starting from a random initial guess, the simulated annealing algorithm evaluated on the average only about 2700 configurations before finding a solution. One of these solutions is tabulated in Table 1 and Table 2. It is a SCARA-like manipulator with a nearly spherical joint at the end of the second link. The offset along the first axis is 1 meter and the first twist angle is 180°, so that the first and second link move in a horizontal plane exactly between the four obstacles. Because of the spherical joint, link 3 can move either in a horizontal or a vertical plane, so that all the task points can be reached without hitting any obstacles, as shown in Figure 5.

3 Fault Tolerance

In the rest of this paper, we focus our attention on one additional task requirement, namely, fault tolerance. To set the stage for our development, we define the following properties of fault tolerant manipulators [2]:

- **Fault Tolerant (FT) Manipulator**: An n-DOF manipulator that will still be able to meet the task specifications, even if any one or more of its joints fail and are frozen at any arbitrary joint angles.
- **Reduced Order Derivative (ROD)**: When some joints of a manipulator fail, the effective number of joints is smaller than the initial number of joints. The resulting faulty manipulator is called a reduced order derivative.
- **Order of Fault Tolerance**: An n-DOF manipulator is FT of the i-th order, if and only if all $(n-i)$ DOF reduced order derivatives can still perform the specified task.
- **Fault Tolerant Work Space (FTWS)**: The fault tolerant work space of an i-th order FT n-DOF manipulator is the set of points reachable by all possible $(n-i)$ DOF reduced order derivatives.

These definitions differ from the concept of fault tolerance as proposed Maciejewski [20]. Instead of attributing the property of fault tolerance to a *manipulator*, he quantifies a measure of fault tolerance for a manipulator *posture* and describes a technique to determine the optimal FT posture, based on the singular value decomposition of the Jacobian matrix. If a joint fails in this optimal posture, the resulting reduced order derivative will have maximum possible dexterity. However, a failure at a different angle may make the execution of the task impossible.

If no specific task is mentioned, it is assumed that the task consists of reaching a nonzero volume of points in the task space, i.e., an m-dimensional manifold in the m-dimensional task space. A manipulator that can only reach a manifold of dimension lower than m in a FT way, is considered not to be fault tolerant.

4 Properties of Fault Tolerant Manipulators

4.1 Existence

Suppose that we have an n-DOF manipulator, $\mathtt{M}_n$, that satisfies all the kinematic requirements of a given task. It has been shown that such a manipulator can be found by using

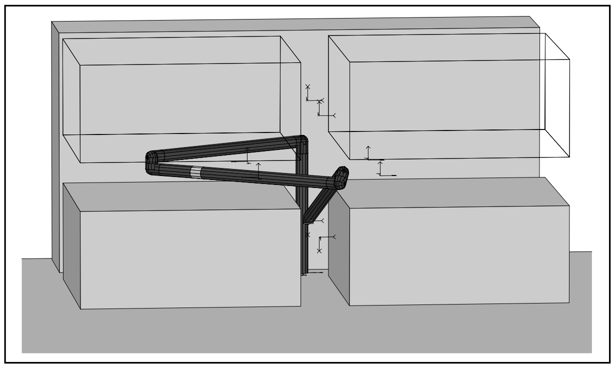

Figure 5: The manipulator reaching point 2 avoiding all the obstacles.

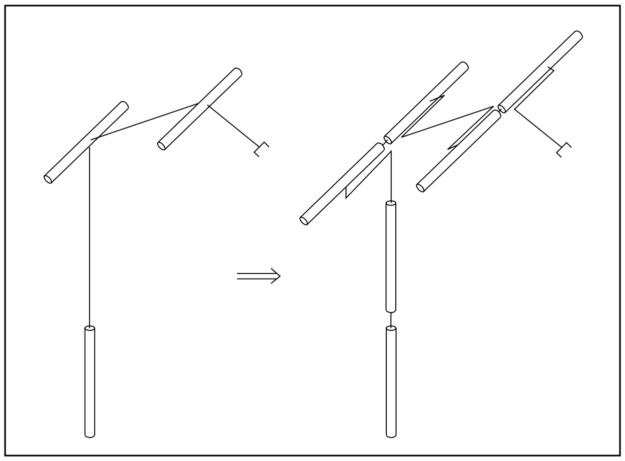

Figure 6: An example of forming a six DOF first order FT manipulator from a three DOF manipulator.

the task based design approach described in Section 2. An obvious way to make this manipulator FT is to design every joint with a redundant actuator [33], as is shown in Figure 6. If one of the actuators of the resulting $2n$-DOF FT manipulator were to fail, the redundant actuator could take over and the manipulator would still be functional. Similarly, a k-th order FT manipulator can be constructed by duplicating every DOF k times, resulting in a $(k+1)\,n$-DOF manipulator.

4.2 Boundary of the Fault Tolerant Work Space

In this section, we show that a boundary point of the FTWS is a critical value (A critical value is an end-effector position that can be reached in a singular configuration, i.e., that is the image of a critical point [4]).

Consider a kth order FT planar manipulator, $\mathtt{M}$. A boundary point, $\boldsymbol{p}_b$, of the FTWS has to be an element of the boundary of the work space of at least one ROD, $\mathtt{M}^*$, obtained by freezing k joints of $\mathtt{M}$. Indeed, if $\boldsymbol{p}_b$ were an interior point of the work spaces of all RODs, then it would by definition be an interior point of the FTWS and not a boundary point. The Jacobian of $\mathtt{M}^*$, J_{M*}, can be obtained from the Jacobian of $\mathtt{M}$, J_M, by deleting the columns corresponding to the frozen DOFs. Because $\boldsymbol{p}_b$ is a boundary point of the work space of $\mathtt{M}^*$, the Jacobian of $\mathtt{M}^*$ at $\boldsymbol{p}_b$ is singular. We prove now that J_M is singular too. Suppose that J_M were non-singular, then at least one of the columns corresponding to a frozen DOF would be outside the column space of the singular matrix, J_{M*}. Physically this means that a small change in the angle of that frozen DOF would cause the end effector of $\mathtt{M}$ to move in a direction with a component perpendicular to the boundary of the work space of the ROD, $\mathtt{M}^*$, as illustrated in Figure 7. The ROD with this new frozen angle would be unable to reach

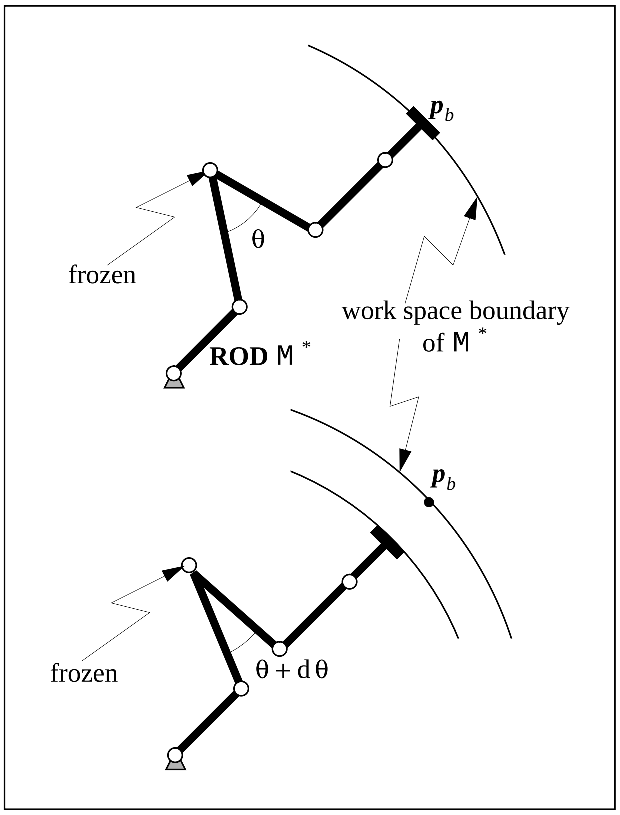

Figure 7: A ROD unable to reach a point outside the FTWS.

the point, $\boldsymbol{p}_b$, which would therefore be outside the FTWS, and this contradicts the fact that $\boldsymbol{p}_b$ is a boundary point of the FTWS. Thus, J_M is singular and $\boldsymbol{p}_b$ is a critical value.

Consequently, the FTWS is bounded by critical value manifolds. For planar positional manipulators, the critical value manifolds are concentric circles, and the FTWS is an annulus with inner radius $R^{\text{FTWS}}_{\text{min}}$ and outer radius $R^{\text{FTWS}}_{\text{max}}$.

4.3 Required Degree of Redundancy

In Section 4.1, it is shown that, in general, kn redundant DOFs—i.e. $(k+1)n$ DOFs in total—are sufficient to achieve kth order fault tolerance. For planar positional manipulators, however, we prove that $2k$ DOFs are also necessary for kth order fault tolerance.

The proof shows that $(2k+1)$ DOFs (or $2k-1$ redundant DOFs) are insufficient, by finding a lower bound for $R^{\text{FTWS}}_{\text{min}}$ and an upper bound for $R^{\text{FTWS}}_{\text{max}}$. First consider the ROD obtained by freezing the first k joints at 0 radians, as illustrated in Figure 8. The maximum reach in the opposite direction is an upper bound for $R^{\text{FTWS}}_{\text{max}}$:

$$R^{\text{FTWS}}_{\text{max}} \le -\sum_{i=1}^{k} l_i + l_{k+1} + \sum_{i=k+2}^{2k+1} l_i \qquad (7)$$

where l_i is the length of the ith link. In order for $R^{\text{FTWS}}_{\text{max}}$ to be positive, we must have that:

$$\sum_{i=1}^{k} l_i \le l_{k+1} + \sum_{i=k+2}^{2k+1} l_i. \qquad (8)$$

Making this assumption, we find that $R^{\text{FTWS}}_{\text{min}}$ is bounded below by the inner radius of the work space of the ROD obtained by freezing the k last joints at 0 radians, as illustrated in Figure 8:

$$R^{\text{FTWS}}_{\text{min}} \ge \sum_{i=k+2}^{2k+1} l_i + l_{k+1} - \sum_{i=1}^{k} l_i. \qquad (9)$$

From Equation (7) and Equation (9), it follows that at best

$$R^{\text{FTWS}}_{\text{max}} = R^{\text{FTWS}}_{\text{min}}, \qquad (10)$$

resulting in a one-dimensional FTWS. Therefore, a $(2k+1)$-DOF manipulator cannot be FT ■

4.4 Including Orientation

Thus far, we have only considered planar positional manipulators. The results for positional manipulators can be easily extended to the case in which orientation is considered also, by converting the orientational problem into an equivalent positional problem:

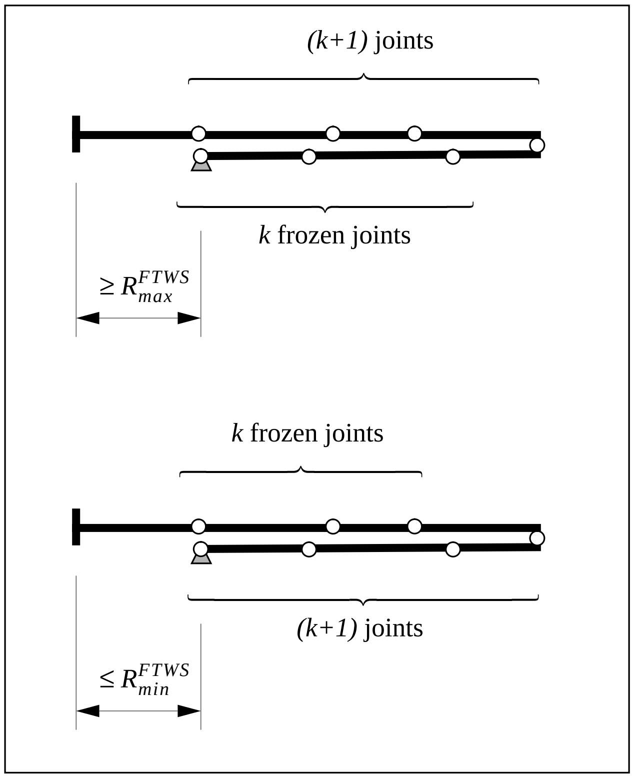

Figure 8: Upper bound for $R^{\text{FTWS}}_{\text{max}}$ and a lower bound for $R^{\text{FTWS}}_{\text{min}}$.

An n-DOF manipulator, $\mathtt{M}$, is kth order FT with respect to a set of points, $W = \{ (x_i, y_i, \varphi_i) \}$, if and only if:

1. the positional manipulator, $\mathtt{M}'$, obtained from $\mathtt{M}$ by deleting its last link, l_n, is kth order fault tolerant with respect to the set of points $W' = \{ (x_i - l_n \cos\varphi_i, y_i - l_n \sin\varphi_i) \}$
2. $\mathtt{M}'$ can reach the points in W' in any direction in a $(k-1)$ th fault tolerant way.

The positional manipulator, $\mathtt{M}'$, needs at least $(2k+2)$ DOFs to be kth order FT with respect to W'; therefore, the manipulator $\mathtt{M}$ needs at least $(2k+3)$ DOFs. Now, consider a $(2k+3)$-DOF manipulator with the first $(2k+2)$ links having length, l, and the last link having length zero. It is easy to verify that this manipulator can reach $W = \{ (x, y, \varphi) | \sqrt{x^2+y^2} \le 2l \text{ and } \varphi \in [0, 2\pi) \}$ in a kth order FT way. Thus, $(2k+3)$ DOFs are necessary and sufficient for kth order fault tolerance of planar manipulators when orientation is included

This result and the result obtained in Section 4.3 can be summarized in the following theorem:

Theorem:

For planar manipulators, $2k$ redundant DOFs are necessary and sufficient for kth order fault tolerance.

4.5 Spatial Fault Tolerant Manipulators

For planar FT manipulators, we were able to prove that $2k$ is the required degree of redundancy. The proof was based on geometric work space analysis. However, the geometric analysis becomes too complex for spatial manipulators, especially since we are dealing with redundant manipulators. Therefore, we will demonstrate some properties of spatial FT manipulators using two examples.

As a first example, consider a 5-DOF spatial positional manipulator. Its D-H parameters are listed in Table 3. This manipulator is first order FT, and because of its simple kinematic structure, an analytic expression for the boundary of the FTWS can be derived. The FTWS is symmetric with respect to the first axis. A cross section (the X-Z plane), as shown in Figure 9, can be described by two segments of a circle with radius 2 and center at $(x = 1, z = 0)$, and a straight line from $(x = 2, z = \sqrt{3})$ to $(x = 2, z = -\sqrt{3})$. An important property of this FTWS is that it does not have any holes or a central void, so that the FTWS of the same manipulator scaled by any factor, $\lambda > 1$, contains the original FTWS. As a result, this FT manipulator can be used as a *design template*. Any specified set of points can be reached in a first order FT way by a scaled version of the template.

DOF i	d_i	a_i	α_i
1	0	1	90˚
2	a	1	0˚
3	-a	1	90˚
4	b	1	0˚
5	-b	1	—

Table 3: D-H parameters of a first order FT spatial manipulator, without orientation.

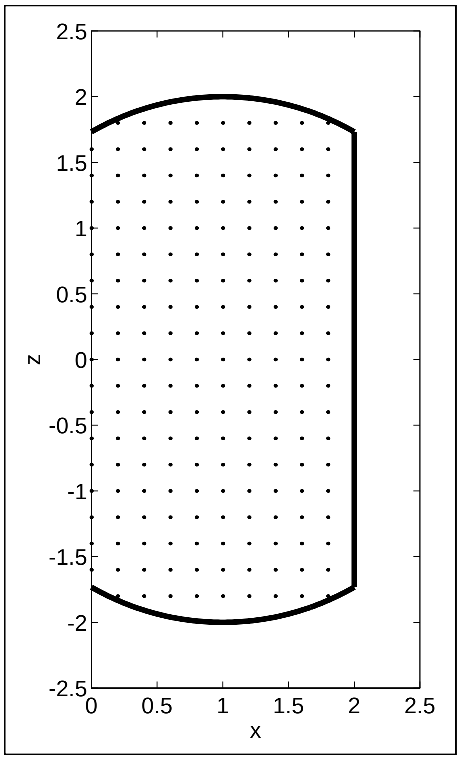

Figure 9: A cross section of the FTWS of a 5-DOF spatial manipulator example.

In Section 4.2, it is shown that the boundary of the FTWS of a planar manipulator coincides with its critical value manifolds. Figure 10 demonstrates that this property also holds for the 5-DOF spatial manipulator considered in this example. The critical value manifolds are computed using the algorithm described in [4] and are depicted in a solid line. The bold part of the critical value manifolds is the boundary of the FTWS.

As a second example, consider an 8-DOF manipulator, with D-H parameters listed in Table 4. It is the same manipulator as in example one, with a zero-length 3-roll-wrist added at the end. This manipulator can reach in a first order FT way all the points in the FTWS of example one *in any direction*. This property can be demonstrated with the following arguments. When one of the first five DOFs fails, the manipulator can still reach any position in the FTWS (because the 5-DOF positional manipulator is FT) and can take any orientation at this position using the intact 3-roll-

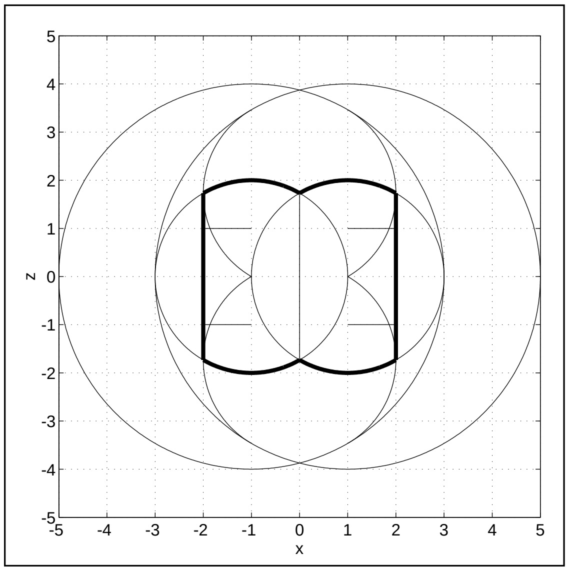

Figure 10: A cross section of the boundary of the FTWS of a 5-DOF spatial manipulator (bold) as part of its critical value manifolds.

wrist. When one of the DOFs in the wrist fails, we are left with a 7-DOF manipulator which has enough orientational capabilities to reach any point in the FTWS in any orientation. Consequently, one could call this the *dextrous* FTWS. Since there are again no holes or voids in the FTWS, this manipulator can also be used as a design template.

Finally, one should notice that both examples have only two redundant DOFs, which seems to indicate that the theorem in Section 4.4 is extendable to spatial manipulators.

4.6 Joint Limits

In Section 3, the definition of a FT manipulator included the specification that joints could fail "at any arbitrary angle." Thus far, we did not consider joint limits and therefore "at any arbitrary joint angle" meant at any angle between 0 and 2π. From now on, however, an arbitrary joint angle is restricted to be within the joint limits. Using two examples, we discuss now how the introduction of joint limits changes the properties of FT manipulators.

First, consider a 4-DOF positional planar manipulator with links of length one. Figure 11 illustrates how the work space changes when the joint limits (the same limits for all joints) vary from $\pm 180°$ through $\pm 30°$. As one would expect the FTWS shrinks, but it also changes position. A point that is *outside* the FTWS of a manipulator without

DOF i	d_i	a_i	α_i
1	0	1	90°
2	a	1	0°
3	-a	1	90°
4	b	1	0°
5	-b	0	90°
6	1	0	90°
7	0	0	90°
8	0	0	—

Table 4: D-H parameters of an 8-DOF first order FT spatial manipulator.

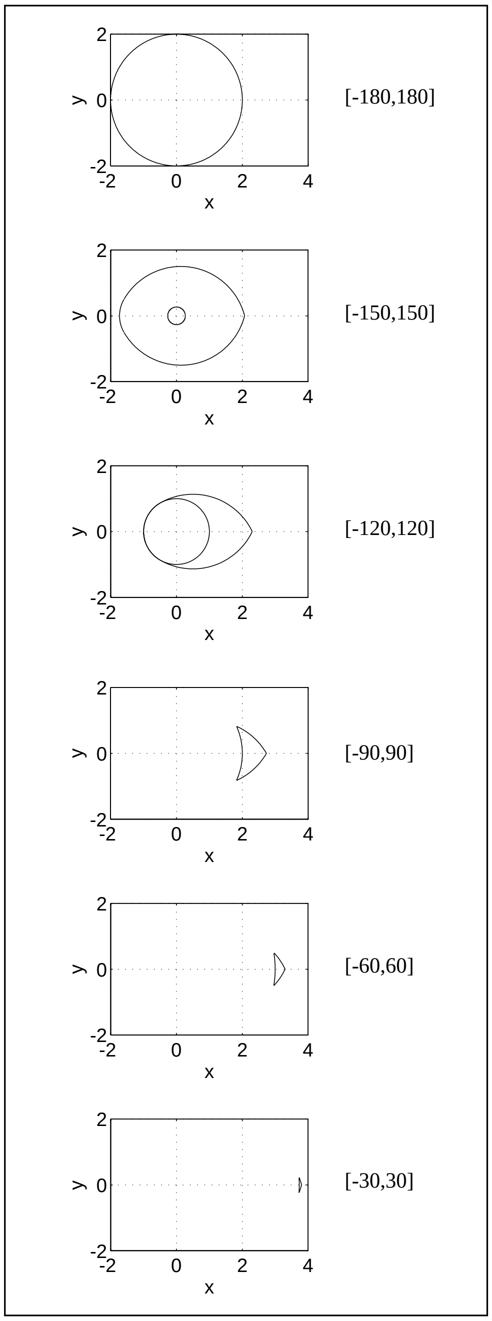

Figure 11: The FTWS of a 4-DOF planar manipulator with joint limits.

joint limits can become *inside* the FTWS when joint limits are introduced. This can be understood by thinking of the FTWS as the intersection of the work spaces of all RODs. When joint limits are introduced, the work space of every ROD reduces in size, but the intersection of fewer RODs has to be taken to obtain the FTWS.

As a second example, we revisit the 5-DOF spatial manipulator of the previous section. As depicted in Figure 12, the FTWS of this 5-DOF manipulator shrinks rapidly when the joint angle intervals are reduced. Moreover, a central hole appears and the FTWS splits into two disconnected parts. When the joint angle interval becomes smaller than $[150°, -150°]$, the FTWS vanishes.

From the above examples, it is clear that the analysis, and therefore also the design of FT manipulators is very complex when joint limits are introduced. Whereas we were able to generate design templates for the case of no joint limits (or joint limits of ±180°), finding a *general* design template is infeasible due to the infinite number of possible joint limits. Therefore, in the next section, we extend the numerical procedure developed in Section 2 to include the determination of the kinematic structure of FT manipulators when joint limits are considered.

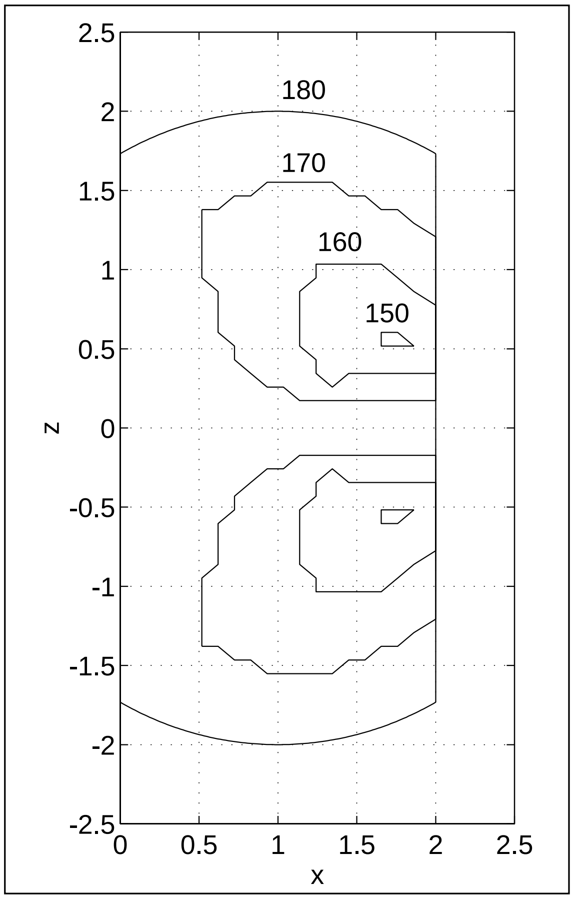

Figure 12: A cross section of the FTWS of a 5-DOF spatial manipulator with joint limits.

5 Task Based Design of Fault Tolerant Manipulators

5.1 Mathematical Formulation

In this section, we formulate a method to determine the kinematic structure of a manipulator which is FT with respect to the task of reaching a set of positions/orientations, $W = \{\mathbf{p}_j = (x_j, y_j, z_j, \vartheta_j, \varphi_j, \psi_j)\}$, and for which the joint limits for each of the DOFs are given. The formulation is based on the approach of Section 2 which can be summarized with the following equation:

$$\min_{\text{DH}} \sum_{\text{task points}} \min_{\text{postures}} P_{post}. \tag{11}$$

A manipulator is first order FT with respect to a task if all possible first order RODs can fulfill the task. Equation (11) can then be adapted to include first order fault tolerance as follows:

$$\min_{\text{DH}} \sum_{\text{task points}} \left(\sum_{j=1}^{n} \max_{\theta_j} \left[\min_{\substack{\theta_i \\ (i \neq j)}} P_{post} \right] \right) \tag{12}$$

The maximization yields zero when all the RODs, resulting from a failure of joint j, are able to reach the task point. As a result, the minimization over the D-H parameters equals zero if and only if all possible RODs are able to reach every specified task point.

5.2 Implementation

Equation (12) contains three nested optimizations. For each ROD, the innermost minimization finds the posture, for which the penalty function is minimal. The task that we consider is to reach a certain point without violating the joint limit constraints. The penalty function therefore consists of two parts. A first part penalizes the failure to reach the point (without considering joint limits). This penalty is equal to the norm squared of the residue, $\|r\|^2$, namely, the distance between the actual position/orientation of the end effector and its desired position/orientation. A second part, penalizes the joint limit violations. This penalty is equal to

$$\sum_{i=1}^{n} \{\max(0, \theta_i - \theta_i^{max})^2 + \max(0, \theta_i^{min} - \theta_i)^2\} . \tag{13}$$

The resulting minimization is thus a nonlinear least squares (NLLS) problem, and is commonly solved in robotics applications using a Newton-Raphson scheme [13] (cfr. numerical inverse kinematics literature). This scheme converges very quickly when the penalty at the optimum is zero, i.e., for a zero residual NLLS problem. However, it is possible that a task point is outside the reach of the manipulator, in which case the minimization becomes a large residual problem. For this class of problems, the Newton-Raphson scheme (or Gauss-Newton algorithm) converges slowly and is outperformed by the BFGS algorithm (Broyden-Fletcher-Goldfarb-Shanno [8]). As shown by Fletcher and Xu [7],

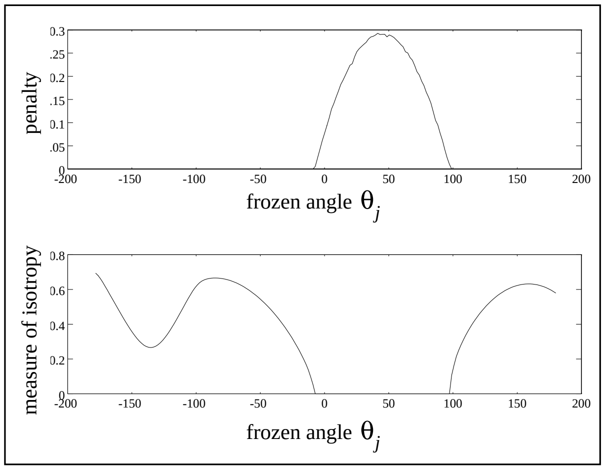

Figure 13: The penalty function and the measure of isotropy.

the best overall performance is obtained by a hybrid algorithm, which combines the advantages of both the Gauss-Newton and BFGS algorithms.

The second optimization in Equation (12) is the maximization over the joint angle, θ_j. A typical example of the objective function is shown in Figure 13. Notice that the function and its gradient are zero possibly over a large part of the domain, namely, where the task point is reachable. To solve the maximization, we need to incorporate additional information. The key observation is that when the desired point is outside the reach of the manipulator, the innermost minimization algorithm always gets stuck at the boundary of the work space of the manipulator, where the Jacobian matrix of the manipulator loses rank [17]. The measure of isotropy, Δ, is defined as [14]:

$$\Delta = \frac{\sqrt[m]{\det(JJ^T)}}{\operatorname{trace}(JJ^T)/m}, \tag{14}$$

with m the number of Cartesian coordinates. For a singular Jacobian, Δ is zero. Consequently, a minimization over Δ leads to regions where the penalty is possibly positive. Thus, the derivative of Δ with respect to θ_j, can be used to compensate for the lack of gradient information in the norm of the residue. As one can see in Figure 13, however, the measure of isotropy is a multimodal function of θ_j. Because there is only one optimization variable, an efficient global optimization algorithm, such as the P^* algorithm developed by Zilinskas [34], can be used. P^* is a combination of a one-stage Bayesian algorithm and Brent's local minimization resulting in smart global coverage and accurate local refinement.

The third and outermost optimization minimizes the total penalty of a kinematic configuration, defined by the $(3n-1)$ D-H parameters, and is solved again using the simulated annealing algorithm.

DOF i	5 DOF solution		
	d_i	a_i	α_i
1	0.2	0.7	90°
2	0.1	1.0	0°
3	0.9	0.1	135°
4	0.6	0.5	0°
5	0.1	1.3	—

Table 5: D-H parameters of a first order FT spatial solution.

6 Numerical Results

To illustrate the approach developed in the previous section, we give an example of a first order FT spatial manipulator design: a five DOF spatial manipulator that is required to be first order FT while accomplishing the task of reaching three points:

$$\begin{aligned} \mathbf{p}_1 &= (0.5, 0, 0.5) \\ \mathbf{p}_2 &= (0, 0.5, 0.25) \\ \mathbf{p}_3 &= (0.75, 0, 0.5) \end{aligned} \tag{15}$$

Notice that we do not consider end effector orientation at this time. From the example in Section 4.5, we know that five degrees of freedom are sufficient for first order fault tolerance. Our numerical results confirm this. The D-H parameters of a possible design are listed in Table 5. Figure 14 shows a cross section along the XZ-plane of the

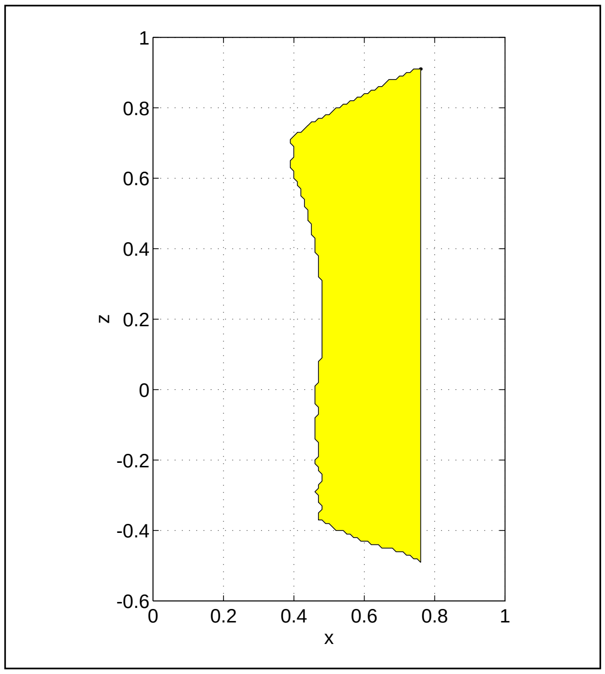

Figure 14: A cross section of the FTWS of the 5-DOF solution

boundary of FTWS.

7 Summary

In this paper, we developed an approach for determining a configuration for a reconfigurable modular manipulator able to fulfill a specific task. We considered tasks that included four kinematic requirements: reachability, joint limits, obstacle avoidance and measure of isotropy. Attributing a penalty to each manipulator configuration, enabled us to reduce the search effort drastically, by guiding the search to the most promising regions of the configuration space. Local minima in the penalty were avoided by using simulated annealing as a search algorithm. We also defined a property of a small class of redundant manipulators, called *fault tolerance*. Based on the definition, we were able to constructively prove the existence of FT manipulators by duplicating the joints. When no joint limits are considered, we proved analytically that, $2k$ redundant DOFs are necessary and sufficient for fault tolerance of planar manipulators. We also proved that the boundary of the FTWS consists of critical values. For spatial manipulators, design templates were introduced, with which a manipulator that is able to reach a specified set of points, can be designed by scaling the template appropriately. For manipulators with joint limits, analytical solutions become infeasible. Therefore, we introduced a numerical design method based on an earlier developed "task based design" approach. All the different steps of our development were illustrated with design examples.

Acknowledgment

This research was funded in part by DOE under grant DE-F902-89ER14042, by Sandia under contract AC-3752-A, by the Department of Electrical and Computer Engineering, and by The Robotics Institute, Carnegie Mellon University.

References

[1] V.P. Agrawal, V. Kohli, S. Gupta, "Computer Aided Robot Selection: the 'Multiple Attribute Decision Making' Approach," *International Journal of Production Research*, Vol. 29, No. 8, pp. 1629–1644, 1991.

[2] W.K.F. Au, C.J.J. Paredis, P.K. Khosla, "Kinematic Design of Fault Tolerant Manipulators," in *Proceedings of the Allerton Conference*, October 2, 1992, Urbana-Champagne, Illinois.

[3] B. Benhabib, G. Zak, M.G. Lipton, "A Generalized Kinematic Modeling Method for Modular Robots," *Journal of Robotic Systems*, Vol. 6, No. 5, pp. 545–571, 1989.

[4] J.W. Burdick, "Kinematic Analysis and Design of Redundant Robot Manipulators," Stanford Computer Science Report no. STAN-CS-88-1207, 1988.

[5] J. Denavit, R.S. Hartenberg, "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," *Journal of Applied Mechanics*, Vol. 22, No. 2, pp. 215–221, June 1955.

[6] P. Fanghella, C. Gellatti, E. Giannotti, "Computer-Aided Modeling and Simulation of Mechanisms and Manipulators," *Computer Aided Design*, Vol. 21, No. 9, pp. 577–583, Nov. 1989.

[7] R. Fletcher, C. Xu, "Hybrid Methods for Nonlinear Least Squares," *IMA Journal of Numerical Analysis*, Vol. 7, No. 3, pp. 371–389, July 1987.

[8] R. Fletcher, *Practical Methods of Optimization, Second Edition*, John Wiley & Sons, New York, 1987.

[9] .T. Fukuda *et al.*, "A Study on Dynamically Reconfigurable Robotic Systems. Assembling, Disassembling and Reconfiguration of Cellular Manipulator by Cooperation of Two Robot Manipulators," in *Proceedings of the IEEE/RSJ International Workshop on Intelligent Robots and Systems, IROS '91*, pp. 1184–1189, Osaka, Japan, Nov. 3–5, 1991.

[10] B.A. Gardone, R.K. Ragade, "IREX: An Expert System for the Selection of Industrial Robots and its Implementation in Two Environments," *Proceedings of the Third International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE 90)*, Vol. 2, pp. 1086–1095, Charleston, SC, USA, July 15–18, 1990. ACM.

[11] A. Groch, L.M. Vidigal, S.W. Director, "A New Global Optimization Method for Electronic Circuit Design," *IEEE Transactions on Circuits and Systems*, Vol. 32, No. 2, pp. 160–169, February 1985.

[12] L. Kelmar, Pradeep K. Khosla, "Automatic Generation of Forward and Inverse Kinematics for a Reconfigurable Modular Manipulator System," *Journal of Robotic Systems*, Vol. 7, No. 4, pp. 599–619, 1990.

[13] P.K. Khosla, C.P. Neuman, F.B. Prinz, "An Algorithm for Seam Tracking Applications," *The International Journal of Robotics Research*, Vol. 4, No. 1, pp. 27–41, Spring 1985.

[14] J.-O. Kim, P.K. Khosla, "Dexterity Measures for Design and Control of Manipulators," in *Proceedings of IROS'91: International Workshop on Intelligent Robots and Systems*, pp. 758–763, Osaka, Japan, Nov. 3–5, 1991.

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

[16] S. Kirkpatrick, C.D. Gelatt Jr., M.P. Vecchi, "Optimization by Simulated Annealing," *Science*, Vol. 220, No. 4598, pp. 671–680, May 1983.

[17] D. Kohli, M.-S. Hsu, “The Jacobian Analysis of Workspaces of Mechanical Manipulators,” *Mechanism and Machine Theory,* Vol. 22, No. 3, pp. 265–275, 1987.

[18] A. Krishnan, P.K. Khosla, “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, also in *Proceedings of the 1990 ISMCR*, Houston, Texas, June 1990.

[19] C.L. Lewis, A.A. Maciejewski, “Dexterity Optimization of Kinematically Redundant Manipulators in the Presence of Joint Failures,” to appear in *Computers and Electrical Engineering*, 1993.

[20] A.A. Maciejewski, “‘Fault Tolerant Properties of Kinematically Redundant Manipulators,” in *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, Cincinnati, Ohio, pp. 638–642, May 1990.

[21] S. Manoochehri, A.A. Seireg, “A Computer-Based Methodology for the Form Synthesis and Optimal Design of Robot Manipulators,” *Journal of Mechanical Design*, Vol. 112, pp. 501–508, Dec. 1990.

[22] S. Murthy, P.K. Khosla, S. Talukdar, “Designing Manipulators From Task Requirements: An Asynchronous Team Approach,” in *Proceedings of the 1st WWW Workshop on Multiple Distributed Robotic Systems*, July, 1993, Nagoya, Japan.

[23] O.F. Offodile, B.K. Lambert, R.A. Dudek, “Development of a Computer Aided Robot Selection Procedure (CARSP),” *International Journal of Production Research*, Vol. 25, pp. 1109–1121, 1987.

[24] O.F. Offodile, W.M. Marcy, S.L. Johnson, “Knowledge Base Design for Flexible Assembly Robots,” *International Journal of Production Research*, Vol. 29, No. 2, pp. 317–328, 1991.

[25] C.J.J. Paredis, *An Approach for Mapping Kinematic Task Specifications into a Manipulator Design*, M.S. Thesis, Electrical and Computer Engineering Department, Carnegie Mellon University, September 1990.

[26] C.J.J. Paredis, P.K. Khosla, “Kinematic Design of Serial Link Manipulators from Task Specifications,” *The International Journal of Robotics Research*, Vol. 12, No. 3, pp. 274–287, June 1993.

[27] V. Potkonjak, M Vukobratovic, “Computer-Aided Design of Manipulation Robots via Multi-Parameter Optimization,” *Mechanism and Machine Theory,* Vol. 18, No. 6, pp. 431–438, 1983.

[28] E. Rich, K. Knight, *Artificial Intelligence, Second Edition*, McGraw-Hill series in artificial intelligence, McGraw-Hill Inc., New York, 1989.

[29] D.E. Schmitz, P.K. Khosla, T. Kanade, “The CMU Reconfigurable Modular Manipulator System,” in *Proceedings of the 19-th International Symposium and Exposition on Robots (ISIR)*, Australia, 1988. ISIR.

[30] A.A. Tseng, “Software for Robotic Simulation,” *Advances in Engineering Software*, Vol 11, No. 1, pp. 26–36, Jan. 1989.

[31] M.L. Visinsky, I.D. Walker, and J.R. Cavallaro, “Layered Dynamic Fault Detection and Tolerance for Robots,” in *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Atlanta, Georgia, pp. 180-187, May 1993.

[32] R.H. Weston, R. Harrison, A.H. Booth, P.R. Moore, “Universal Machine Control System Primitives for Modular Distributed Manipulator Systems,” *International Journal of Production Research*, Vol. 27, No. 3, pp. 395–410, 1989.

[33] E. Wu, M. Diftler, J. Hwang, J. Chladek, “A Fault Tolerant Joint Drive System for the Space Shuttle Remote Manipulator System,” in *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, Sacramento, California, pp. 2504–2509, April 1991.

[34] A. Zilinskas, “Optimization of One-Dimensional Multimodal Functions, Algorithm AS 133,” *Applied Statistics*, Vol. 23, pp. 367–375, 1978.