

Predictable Motion of Hyper-redundant Manipulators Using Constrained Optimization Control*

Markus P.J. Fromherz, Warren B. Jackson

Xerox PARC, 3333 Coyote Hill Road, Palo Alto, CA 94304, U.S.A.
{fromherz,wjackson}@parc.xerox.com

June 29, 2000

Abstract

Hyper-redundant robotic manipulators are robots that have many more degrees of freedom than required for a typical task such as grasping an object in 3D space. The large number of joints, which may range from dozens to thousands, offers both opportunities and challenges for the control of such robots. An opportunity is to use the extra degrees of freedom to optimally control multiple objectives. A challenge is to deliver predictable behavior despite the large number of possible configurations. In this paper, we propose to use a control approach based on constrained optimization. We discuss a representative example for an under-constrained task and suggest and analyze possible solutions that ensure predictable behavior.

1. Introduction

Modular robots are robots consisting of many interchangeable robotic modules connected into a mechanical and functional assembly. Interest in modular robots has increased over recent years due to their potential for robustness, reduced costs, and wide range of applicability particularly in highly constrained environments [Chirikjian & Burdick 94, Yim 94]. Typically, there are only a small number of module types, and each module type only has limited motion capability, e.g., for the case considered in this paper only one rotational joint. The flexibility and low cost of modular robots is achieved through the large number of mass-fabricated modules, expected to range from dozens to thousands, and their many possible configurations. Because of the large number of degrees of freedom of the redundant modular limbs, such modular robots are also called hyper-redundant [Chirikjian & Burdick 94]. This paper concerns the control of a subclass of such robots, namely hyper-redundant manipulators that consist of a chain of modules with rotational joints (cf. Fig. 1).

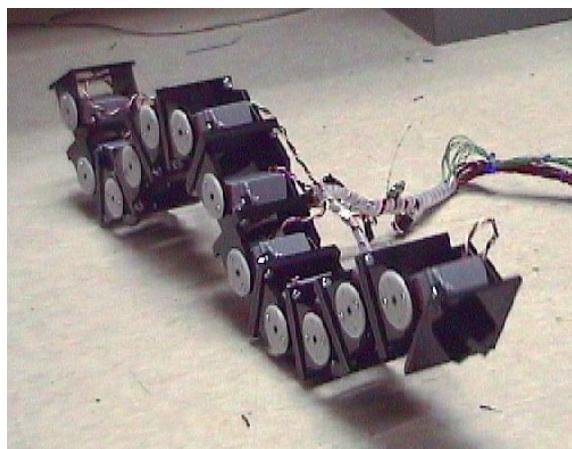


Fig. 1 Modular robot prototype PolyBot (12 joints)

In order to realize the potential of modular robots, robust, versatile and scaleable control algorithms must be developed while satisfying a number of changing joint and force constraints. For example, a manipulator may not only be asked to achieve typical goals such as following a trajectory with its end effector, but also to minimize velocities, distribute torques equally, and avoid obstacles. At the same time, each module has to obey its physical limits, such as limits on joint angles and motor torques. Finally, in order to simplify programming different configurations of modular robots, the software should also be generic with respect to constraints and goals.

In our work, we attempt to robustly handle a variety of constraints for the many degrees of freedom found in modular robots. We cast the control problem as a constrained optimization problem. The hard constraints represent the (mainly physical) limits of the robot, while the objective function expresses the various goals of interest (soft constraints). Furthermore, we assume that end-effector goals (e.g., position and orientation) are given by a higher-level controller such as a path planner, and we focus on the problem of *actuation*

* Presented at IC-AI'2000, Las Vegas, NV, June 2000.

allocation, the problem to finding optimal solutions for the joint angles at each time step. We have argued in favor of constraint-based angle allocation [Fromherz et al. 99] because (1) it scales to larger numbers of joints than traditional, analytic control algorithms, (2) a variety of constraints can be easily added either as terms in the objective function or hard constraints, and (3) constrained optimization readily handles a multitude of (sometimes conflicting) objectives.

Because of the complexity of the objective function as well as the constraints, constrained optimization is sensitive to initial configurations by finding only nearby local minima. Also, because hyper-redundant manipulators are significantly under-constrained, widely different behaviors develop depending on minor differences in the starting conditions. For example, the foot of a spider-robot leg can be controlled to follow an arc trajectory, but the state of the modules in between foot and hip must be guided as well to avoid inefficient or even destructive configurations.

In the rest of the paper, we show that the flexibility in objective and constraint selection in our constrained optimization control framework provides the necessary tools to overcome these local minima related problems. After a review of related work and our robot and task models, we discuss and evaluate two aspects of constrained optimization control: (1) *conflicting objectives*; (2) *uncontrolled behaviors* sensitive to starting conditions.

2. Related Work

Previous work on the control of redundant manipulators has focused on their kinematics, involving the computation and inversion of the Jacobian mapping from joint space to the end-effector space [Khatib 87, Ghosal & Roth 88, Chiu 88]. These approaches typically include few or no constraints and have been applied to robots with a small number of degrees of freedom (less than 10). Attempts have been made to include more constraints by extending Jacobian methods with projected gradients of the constraints [Baillieul 86] or numerically computed gradients [Klein et al. 95]. These methods can work reasonably well for systems with a small number of joints and a few well-characterized constraints. However, modular robot control problems tend to have too many degrees of freedom and mixed task space, joint space, and dynamic inequality constraints to solve such systems in real time. Also, because these methods tend to require a detailed constraint

analysis, it is difficult to robustly adjust the priorities of different constraints over time as robot environment and pose change. Moreover, since the solutions are computed in velocity space, position errors can build up over time.

Control of hyper-redundant robots using continuous backbone curves has been well studied [Chirikjian & Burdick 94, Chirikjian & Burdick 95]. This approach scales very well to large numbers of modules and is of particular interest in hierarchical control approaches. Again, this approach has only been studied with kinematic constraints.

Taking into account constraints such as torque limits is particularly relevant for modular robots. Traditional robots are designed such that their joint torque limits are never exceeded (e.g., by making the base joints strong and the end joints lightweight). In a modular robot, all modules are similar and often relatively weak.

In summary, in contrast to traditional approaches to manipulator control, we attempt to separate constraints, goals, and control algorithms from each other, and to include a larger class of constraints. So far, we have focused on finding effective constraints and objective functions. We assume a generic nonlinear constraint solver and leave the development of a dedicated solver for real-time control to future work.

3. Robot and Task Models

We have previously presented a model for our manipulator and shown how constraints and task objectives can be expressed in terms of this model [Fromherz et al. 99]. We review these definitions in this section. The explanations are necessarily brief, and we refer the interested reader to the earlier publication for details.

3.1 Manipulator Model

For the purpose of this paper, we focus on a robotic manipulator that consists of a chain of n links. This manipulator is attached to a base at one end (link 1), and the primary task is to reach a goal position, or follow a goal trajectory, with the other end (link n). Such a manipulator may, for example, be an arm (which is attached to a table or robot body and whose “hand” end grasps or pushes other objects) or a leg (which is attached to a robot body and whose “foot” end is moving over the ground during locomotion). Note that for now we consider only statically stable configurations, and, due to the relatively slow movement of the joints, we are not concerned with dynamics.

Each manipulator link i has its own local coordinate system, or *frame*. Following robotics conventions [Craig 89], the kinematics of each link is defined by its Denavit-Hartenberg parameters $\langle \alpha_i, a_i, \theta_i, d_i \rangle$, the extensions and rotations from one frame to the next. These parameters define a *homogeneous transform matrix* ${}^{i-1}T_i$, a rotation plus a translation, that maps a point ${}^i p$ defined in frame i to point ${}^{i-1} p$ defined in frame $i-1$ by the multiplication ${}^{i-1} p = {}^{i-1}T_i {}^i p$ [Craig 89]. We will also use the functional denotation ${}^{i-1}T_i = T(\alpha_i, a_{i-1}, \theta_i, d_i)$ to describe the matrix. Transform matrices can be composed by multiplying them. In particular, the transformation from base frame 0 to frame i is given by

$${}^0T_i = {}^0T_1 {}^1T_2 \dots {}^{i-1}T_i = {}^0T_i {}^{i-1}T_i$$

In addition to the kinematics parameters, each link has the following parameters: link mass m_i , center of mass position ${}^i c_i$ (in the link's frame), joint angle limits $\theta_{\min,i}$ and $\theta_{\max,i}$, maximum rotational velocity $\omega_{\max,i}$ achievable by the joint, and maximum motor torque $\tau_{\max,i}$. Angles are expressed in radians, times in seconds, distances in meters, masses in kilograms, and torques in Newton-meters.

3.2 Manipulator Constraints

The manipulator's only actuator variables are the joint angles $\theta_1, \dots, \theta_n$, and thus all constraints ultimately have to be expressed as constraints on these angles and/or their derivatives. Given the links' parameters, we take into account at least three types of constraint: joint angle limits, torque limits, and velocity limits.

Angle limits. A joint's movement is restricted by the mechanics of the link in either direction of the rotation and is defined trivially as

$$\theta_{\min,i} \leq \theta_i \leq \theta_{\max,i} \text{ for all } i = 1, \dots, n$$

Torque limits. For each joint i , the links from i through n together exert a torque onto joint i , which is limited by the strength of the joint. As noted before, this problem has traditionally not been addressed in manipulator control, and related work has been restricted to 2D manipulators [Agrawal & Garimella 94, Gokce & Agrawal 99].

A torque is determined primarily by the gravitational force (applied at the center of gravity of links i through n) and the moment arm (the vector from the joint axis to where the force is applied). As a vector, this torque is both perpendicular to the moment arm and tangential to the rotation. The torque τ_i on joint i can be computed as

$$\tau_i = \hat{Z}_i \bullet (\mathbf{R}_i \times \mathbf{F}_i)$$

where \hat{Z}_i is the axis of rotation, \mathbf{R}_i is the vector from joint i to the center of mass C_i of links i through n , and \mathbf{F}_i is the gravitational force on links i through n . The nonlinear torque constraint is defined as

$$\tau_i(\theta) \leq \tau_{\max,i} \text{ for all } i = 1, \dots, n$$

Velocity limits. For the movement from a previous position, we define as a constraint the maximum velocity achievable by each joint, i.e., $|\omega_i| \leq \omega_{\max,i}$ for all i . We use a linear approximation for joint velocity, i.e., $\omega_i = (\theta'_i - \theta_i)/dt$, where dt is the time step for control and θ'_i is the previous angle value for joint i . Thus, this constraint is defined as

$$\theta'_i - \omega_{\max,i} dt \leq \theta_i \leq \theta'_i + \omega_{\max,i} dt$$

for all $i = 1, \dots, n$

3.3 Task Model

For the purpose of this paper, the main goal of the robot manipulator is to reach or follow a target position with its end link. There may be secondary goals, however, such as also achieving a target orientation with the end link, minimizing the difference from a previous position, and distributing actuation equally. At the same time, the manipulator has to satisfy its constraints, and it may not always be able to achieve its goals exactly. Thus, we define the actuation allocation task as the constrained optimization problem

$$\begin{aligned} & \text{minimize} && h(\theta) \\ & \text{subject to} && c(\theta) \end{aligned}$$

where $\theta = \{\theta_1, \dots, \theta_n\}$ are the free variables. The hard constraints $c(\theta)$ are those presented above. The objective function is reviewed in the balance of this section.

Goals we have implemented include a position goal, an orientation goal, minimal actuation energy, and actuation distribution. The total objective function is a weighted, normalized sum of the individual goal functions $h_i(\theta)$:

$$h(\theta) = \frac{\sum_i w_i \frac{h_i(\theta)}{q_i}}{\sum_i w_i}$$

Thus, both the individual goal functions and the entire sum are normalized to be in the range [0,1]. The weights $w_i \in [0,1]$ determine the relative priority of the goals, and the scaling factors $q_i = \max_{\theta} h_i(\theta)$ normalize the goal functions (not discussed in this paper).

Position goal. Given a goal position $p_g = [x_g \ y_g \ z_g \ 1]^T$ for the end effector (the end of link n), the goal function is

$$h_1(\theta) = \sqrt{(x_g - x_e)^2 + (y_g - y_e)^2 + (z_g - z_e)^2}$$

where p_e is the position of the end effector.

Our normalization factor for this goal is the sum of the distance from base origin to goal position and the maximum reach of the manipulator:

$$q_1 = \sqrt{x_g^2 + y_g^2 + z_g^2} + \sum_{i=1}^n a_i$$

Orientation goal. We currently allow one to define a goal twist α_g and rotation θ_g for the end effector. These angles define a frame ${}^0_g T = T(\alpha_g, 0, \theta_g, 0)$. Using Euler's theorem on rotation [Craig 89], we relate the rotation from this frame to the frame ${}^0_e T$ of the end-effector with a single axis and a single rotation angle ${}^e_g \Theta$. This angle can be computed from the rotation matrix ${}^e_g R$ between the two frames as an angle in the interval $[0, \pi]$.

We define the goal function for an orientation goal by this rotation angle:

$$h_2(\theta) = {}^e_g \Theta$$

Since the maximum rotation angle is by definition π , the normalization factor for this goal is

$$q_2 = \pi$$

We have often used additional objectives, such as minimal actuation energy and equal actuation distribution. See the earlier paper for details [Fromherz et al. 99].

4. Achieving Predictable Behavior

The problem is to use the objective functions and constraints of the previous sections to obtain desirable trajectory-following behavior for a modular robot chain. Using constrained optimization, our approach to controlling a manipulator consists of three levels: a *system-level (feedback) controller* that determines via points on the trajectory together with other objectives and their weights, the *manipulator-level controller* where a solver determines suitable configurations for the given objectives, and *module-level feedback controllers* that attempt to achieve the sequence of joint angles given by these configurations.

Here, we focus on the interaction between the upper two levels. Given a sequence of objectives over time, the solver finds configurations that optimize these objectives and satisfy the constraints.

4.1 Simulation

In order to explore the possibility of using constrained optimization in a control approach for modular robots, a testbed was implemented in Matlab in which a manipulator can be represented according to the model described in the previous section. In particular, modules are described by the following parameters: link twist α_i , link length a_i , rotation θ_i , extension d_i , link mass m_i , center of mass position ${}^i c_i$, joint angle limits $\theta_{\min,i}$ and $\theta_{\max,i}$, maximum rotational velocity $\omega_{\max,i}$ achievable by the joint, and maximum motor torque $\tau_{\max,i}$.

In a typical manipulator configuration, joints are attached to each other at offsets of 90 degrees to one another so that their rotations are in orthogonal planes, i.e., $\alpha_i = (-1)^i \pi/2$. The center of mass for each joint, ${}^i c_i$, is set at the origin of frame i , where the motor would be located for a physical manipulator. Link length a_i , link mass m_i , and maximum motor torque $\tau_{\max,i}$ are set such that a motor can lift approximately 5 other modules without exceeding the maximum motor torque. Joint angle limits $\theta_{\min,i}$ and $\theta_{\max,i}$ are set to -80 degrees and 80 degrees respectively. The results discussed in this paper were achieved using a manipulator composed of 10 joints (cf. Fig. 2).

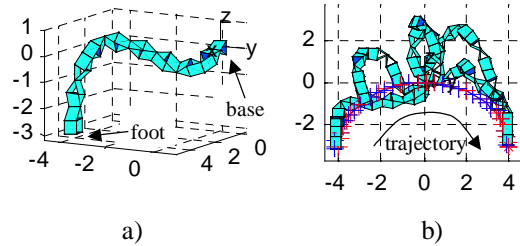


Fig. 2 Rendering of a) a typical robot leg configuration, b) snapshots from a leg motion

At present we are using an active set nonlinear constrained optimization routine, *fmincon*, provided as part of the Matlab optimization toolbox. This nonlinear optimizer allows us to solve the constraint optimization problem directly in joint space with a set of current joint angles as starting point.

In the following evaluations, we will continue to use the example of a robot leg whose foot is asked to move on an arc trajectory (cf. Fig. 2b). Besides

different objectives, we also compare different starting configurations; Figs. 2a and 3 show some of the variability in possible start configurations for the same base and end positions.

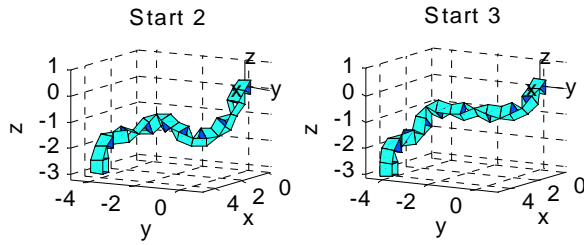


Fig. 3 Two test start configurations

4.2 Conflicting Objectives

One cause for undesirable local minima arises from multiple conflicting objectives. In the leg example, we typically have multiple goals, such as putting down and lifting the foot at appropriate times, putting the foot down at the right angle, leaving enough clearance under the leg to avoid common obstacles, etc. A solution for this type of problem is to establish a ranking of the various objectives and turn off the less important ones at first while optimizing only over the most important ones. Later, the less crucial objectives are gradually turned on, thereby finding a local solution that preferentially optimizes the important objectives over the less important ones. In evaluating the leg example, the orientation angle goal is eliminated until the leg successfully tracks the position goal. Then, the angle (orientation) goal weight is adjusted to be inversely proportional to the distance from the ground to the foot, i.e., the closer the foot comes to the ground, the more important the angle goal becomes. By the time the foot arrives at the goal, it is at the correct orientation as well as the correct position. We compare this dynamic adjustment of the angle portion of the objective function with an angle goal that is fixed from start to end. The results are shown in Fig. 4, comparing the effects of fixed and variable angle goals on trajectory error for five different start configurations. Varying the angle goal performs significantly better, especially on the “hard” start configuration.

In general, constrained optimization control with conflicting objectives can lead to jerky motion, while adapting objective weights as needed makes the motion smoother and more predictable.

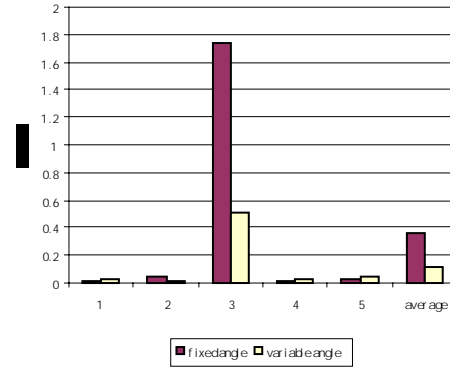


Fig. 4 Trajectory error comparison of fixed vs. variable end-effector angle goals for five start configurations

4.3 Uncontrolled Behavior

A second problem arises when the objective function does not strongly differentiate between a large number of alternative local minima. Different start configurations can lead to qualitatively very different behaviors. The snapshots in Figs. 2b and 5 from leg motions for different objectives indicate that even if the foot follows its trajectory in the same way, the configuration overall usually reflects the start configuration (e.g., middle modules stay back, or low).

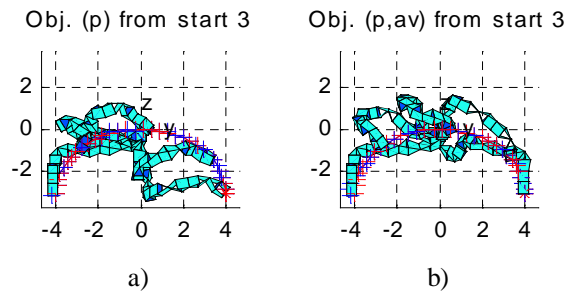


Fig. 5 Two leg motions from start configuration 3: a) with just an end-position goal, b) with end-position and variable angle goals

In such conditions, exactly reproducible behavior is not required, but qualitatively similar behavior for each run is preferred, e.g., an arc from base to foot (to avoid common objects on the ground) or a straight line from base to hand (to ensure maximum movement flexibility in any direction). In this situation it is necessary to add additional objectives, which is in contrast to the previous section where we eliminated objectives.

For the leg example, one such solution is to specify goals for one or more middle module positions.

Instead of specifying goal positions, we define control points towards which specific modules are pulled with what we think of as virtual, elastic strings. For example, a suitable control point for the leg trajectory is a point vertically above the mid-point between base and via point, such that the middle section of the leg is pulled upwards.

To measure the desired behavior, we use a benchmark model that defines, for each via point on the foot trajectory, an elliptical arc from the base to the via point. Achieving the desired behavior described above is then defined as conforming to these arcs as closely as possible, and we measure as “arc error” the sum of the distances between leg modules and corresponding points on the benchmark arc.

Fig. 6 shows results for runs with different combinations of objectives from five different initial configurations. The objectives are foot position (p), fixed or variable foot angle (af or av), and strings (s). We measure the total trajectory (“pos”) error, arc error, and time. The results show the positive impact of strings without negative impact on trajectory error.

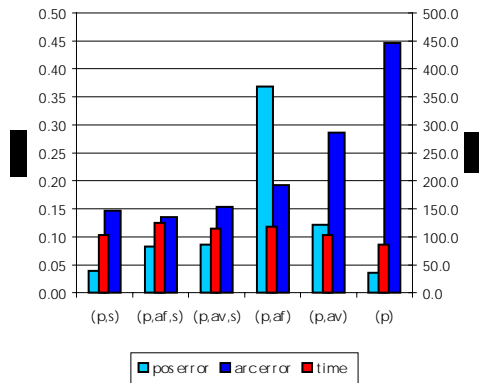
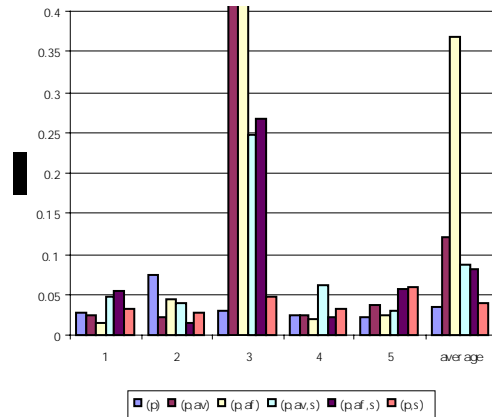


Fig. 6 Measurements of trajectory (pos) error, behavior benchmark (arc) error, and time for different combinations of objectives when controlling leg motion

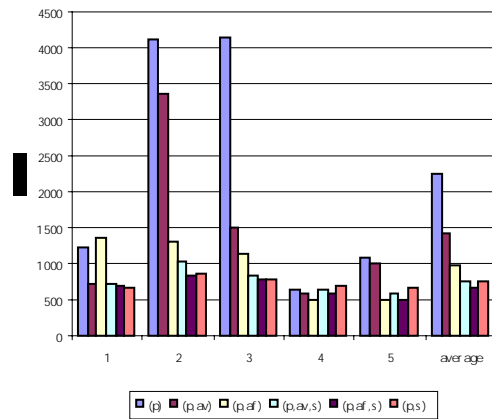
In Fig. 7, the measurements are further broken down by start configurations. This reveals again how some combinations of objectives depend on the start configuration (and thus are less predictable), while others are more independent of the initial conditions.

We have found that the addition of objectives for under-constrained parts of a hyper-redundant manipulator usually makes its controlled behavior significantly more robust. The use of “string constraints” in particular, which guide the robot

akin to control points for a spline, have proved to be a simple and versatile tool in different kinds of trajectory following tasks. For example, specifying strings for one or two points is usually sufficient, and the objective can be defined independent of the number of modules or their particular parameters. Furthermore, as the results above suggest, such string constraints not only help enforce desirable robot configurations, but also help reduce the trajectory following error. For example, we have found situations where the middle part of the manipulator actually gets in the way of the end-effector unless it is pulled out of the way in time.



a)



b)

Fig. 7 Details of the measurements of Fig. 6, broken down by start configurations: a) trajectory error, b) behavior benchmark (arc) error

5. Conclusions

In this paper, we have presented constrained optimization based control for hyper-redundant, reconfigurable manipulators that can be extended readily with additional constraints on joints and sets of joints without detailed analysis. In fact, constraints and objectives can be added dynamically during operation of the robotic control system. This approach was applied to a problem of reaching various objectives such as goal position and actuation distribution for a modular robot. In this application we found that, while the constrained optimization based control was scalable and flexible, occasionally it would fall into undesirable local minima. When the problem was over-constrained with conflicting objectives leading to a large objective error, sequential adaptation of prioritized goals proved to result in greatly improved behavior. In the second type of problem, the local minima and start point dependence arose from an under-constrained specification. In this case predictable behavior was obtained by adding terms to the objective function. Specifying the position of internal modules in addition to the end modules results in better behaved solutions from the constrained optimization. In both cases, the adjustment of the objectives is performed dynamically as the trajectory is followed. This dynamic feedback between optimizer performance and adaptation of the objective functions from the supervisory level has the potential to significantly improve the robustness of the constrained optimization approach for hyper-redundant modular robot control.

6. References

- [Agrawal & Garimella 94]
S. K. Agrawal and R. Garimella, "Workspace boundaries of free-floating open and closed chain planar manipulators." In: *Journal of Mechanical Design*, vol. 116, March 1994, pp. 105-110.
- [Baillieul 86]
J. Baillieul, "Avoiding obstacles and resolving kinematic redundancy." In: *Proc. 1986 IEEE Int. Conf. on Robotics and Automation*, 1986, pp. 1698-1704.
- [Chirikjian & Burdick 94]
G. S. Chirikjian and J. W. Burdick, "A modal approach to hyper-redundant manipulator kinematics." In: *IEEE Trans. On Robotics and Automation*, 10, 1994, pp. 343-354.
- [Chirikjian & Burdick 95]
G. S. Chirikjian and J. W. Burdick, "Kinematically optimal hyper-redundant manipulator configurations."

- In: *IEEE Trans. On Robotics and Automation*, vol. 11, 1995, pp. 794-806.
- [Chiu 88]
S. L. Chiu, "Task compatibility of manipulator postures." In: *Int. Journal of Robotics Research*, vol. 7, no. 5, 1988, pp. 13-21.
- [Craig 89]
J. J. Craig, *Introduction to Robotics Mechanics and Control*. Addison Wesley, 1989.
- [Fromherz et al. 99]
M. P. J. Fromherz, M. Hoeberechts, and W. B. Jackson, "Towards Constraint-based Actuation Allocation for Hyper-redundant Manipulators." In: *CP'99 Workshop on Constraints in Control (CC'99)*, Alexandria, VA, Oct. 1999. <http://www.parc.xerox.com/fromherz/publications/sm-cc99-abstract.html>
- [Ghosal & Roth 88]
A. Ghosal and B. Roth, "A new approach for kinematic resolution of redundancy." In: *Int. Journal of Robotics Research*, vol. 7, no. 2, March/April 1988, pp. 22-35.
- [Gokce & Agrawal 99]
A. Gokce and S. K. Agrawal, "Mass center of planar mechanism using auxiliary parallelograms." In: *Trans. of the ASME*, vol. 121, March 1999, pp. 166-168.
- [Khatib 87]
O. Khatib, "A unified approach to motion and force control in robotic manipulators: The operational space formulation." In: *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, 1987, pp. 43-53.
- [Klein et al. 95]
C. A. Klein, C. Chu-Jenq, and S. Ahmed, "A new formulation of the extended Jacobian method and its use in mapping algorithmic singularities for kinematically redundant manipulators. In: *IEEE Trans. On Robotics and Automation*, vol. 11, 1995, pp. 50-55.
- [Yim 94]
M. Yim, *Locomotion with a Unit-Modular Reconfigurable Robot*. Ph.D. Thesis, Dept. of Mechanical Engineering, Stanford University, 1994.