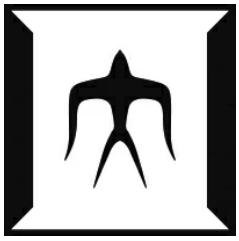
論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Stable Posture Tracking Framework for Modular Hyper-Redundant Serial Arm - A Singularity Avoidance Driven Approach
著者(和文)	DIONEAIMECHARLESALFRED
Author(English)	Aime Acadione
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第12680号, 授与年月日:2024年3月26日, 学位の種別:課程博士, 審査員:長谷川 晶一,小野 功,中本 高道,三宅 美博,山村 雅幸
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第12680号, Conferred date:2024/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Stable Posture Tracking Framework for Modular Hyper-Redundant Serial Arms

A Singularity Avoidance Driven Approach



Aime Charles Alfred Dione

Department of Computational Intelligence and Systems Science
Interdisciplinary Graduate School of Science and Engineering
Tokyo Institute of Technology

Thesis Submitted in Partial Fullfilment
of the Requirements for the Degree of
Doctor of Engineering

December 2023

I would like to dedicate this thesis to my parents Raymond Salla Dione and Anne Marie Dione. They went through tremendous sacrifices to always put me in the best conditions. Their unconditional love and support drive me forward. I owe them everything I am and will ever be. I devote my life to expressing to them the depth of my gratitude.

Acknowledgements

First and foremost, I would like to extend my deepest gratitude to my academic advisor Shoichi Hasegawa. He has always been there and provided countless invaluable guidance on my academic work. His unwavering patience, kindness, and support in my moments of weakness and doubts gave me the strength to carry on. I would never have hoped to complete this thesis without his help and I will always owe him a great deal of gratitude.

I am also grateful for the Japanese Government MEXT Scholarship program which gave me the opportunity to pursue higher education in Japan and provided me with the financial relief that allowed me to focus on my academic work over so many years. My thanks also go to the Honjo International Scholarship Foundation. It provided much-needed financial support and a stimulating community of brilliant minds.

Finally, I would be a miss if I did not acknowledge a handful of friends whom I consider family. Countless times I was down, they were always there to lift me up, their generosity is without limits, their high expectations of me boosted my motivation. They are my heroes and I will strive for the rest of my life to be there for them. My thanks go to Abdoulaye Niang, Souleymane Tamboura, Ibou Fall, Khadim Ndiaye, Sidy Yakhya Ba, Daouda Toumbou, Adama Gueye, Ibra Thiam, Daniel Dzissah Agbesy, and Djiby Marema Diallo.

Abstract

Robotic arms with redundant degrees of freedom exhibit characteristics that make them more desirable than their non-redundant counterparts in many real-world manipulation tasks. Indeed, the provision of redundant degrees of freedom gives robots increased flexibility which can be leveraged to achieve additional goals besides the primary task being performed at the end-effector. As the degree of redundancy increases with hyper-redundant robotic arms, the control of the posture or shape of the arm tends to supersede that of its end-effector pose. This is because now the arm can use its shape to grasp or manipulate objects in its workspace, or even produce propulsion forces which can result in a net displacement of the whole robot. Redundancy introduces several challenges in the design, analysis, and control of robotic arms. One such challenge is the difficulty of specifying tasks for hyper-redundant arms. This thesis proposes an intuitive way to specify tasks for hyper-redundant arms by encoding them as user-defined spline curves. The thesis then presents a control framework that is designed to track these arbitrary posture curves. Examples of a planar and spatial hyper-redundant arm were used to test the control framework in numerical simulations. The presented control schemes have shown capable of tracking arbitrary postures and they have stable operation in the vicinity of kinematic singularities.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Redundancy as defined in the dictionary commonly means the state of exceeding what is necessary. In the context of robotic arms, kinematic redundancy describes robots that have more independent degrees of freedom (DOFs) than are required to perform a given task. Therefore kinematic redundancy is with respect to the task at hand and indicates the capability for robots to achieve assigned tasks while still having self-motions left in them to pursue some other goals. This means that regardless of the degrees of freedom a robotic arm possesses, it can be kinematically redundant or not depending on the task assigned to it. Common tasks performed by robotic arms involve the positioning and orientation of the end-effector. For such tasks, 3 degrees of freedom are required for the planar case and 6 degrees of freedom for the spatial case thus robotic arms exceeding these numbers are considered redundant. Another kind of redundancy exists, namely actuator redundancy which refers to robots that have more actuators than strictly necessary to perform a task. The focus of this work however is on kinematic redundancy.

Robotic arms must adhere to a number of constraints while performing tasks. As mandatorily present constraints, mechanical joint limits, and singularity avoidance can be cited. Robotic arms equipped with minimum degrees of freedom are severely limited and they can only hope to honor these constraints (in addition to the end-effector task) at the expense of a reduced workspace. Most robotic tasks involve even more constraints such as obstacle avoidance, and minimal energy consumption. This means that in practice, robotic arms with some degrees of redundancy are more useful as they are capable of fulfilling these additional constraints [31, 27] alongside the main task performed at the end-effector.

Hyper-redundant robotic arms are redundant arms with large degrees of redundancy or which are continuously deformable (continuum robotic arms). These arms are characterized by increased dexterity and are capable of advanced manipulation tasks in highly congested environments. Besides their enhanced ability to maneuver around obstacles [6, 25], the

unprecedented flexibility of this category of arms allows robots to perform a new range of tasks that are out of the reach of non-redundant or mildly redundant arms. That is they can leverage their flexible shapes (postures) to perform whole arm manipulation and grasping tasks [7]. Furthermore, they become capable of modulating their shapes to produce propulsion forces to achieve various locomotion tasks [9, 28]. In other words, for hyper-redundant arms, the shape of the arm itself can be used as a hand to perform manipulation tasks and as legs to perform locomotion tasks (Figure 1.1). Therefore, the control of the posture become as important as the control of the end-effector.

(a) Unified Snake Robot [38] (b) ACM-R5 Snake Robot [42]

Fig. 1.1 Hyper-redundant robots using their redundancy for locomotion tasks

Redundancy is widely found in nature. Snakes, worms, and elephant trunks are a few examples that have been inspirations for numerous hyper-redundant robot designs (Figure 1.1). Another example is the human arm which has 7 degrees of freedom (3 at the shoulder, 1 at the elbow, and 3 at the wrist) and is capable of keeping the pose at the hand constant while having a degree of motion in the rest of the arm. Numerous human-arm-inspired robotic arms are also found in the literature.

Important applications for hyper-redundant robots range from search and rescue operations to inspection and maintenance of infrastructures like nuclear reactors (Figure 1.2) and highways. Other anticipated areas of application are in healthcare where they can be used to perform non-invasive surgical operations, in space and deep see exploration as they can reach behind obstacles and scrawl into tight spaces.

The additional complexity introduced by redundancy makes analysis and control of hyper-redundant robotic arms challenging. Numerous research works presented in Section 1.3 have been devoted to tackling these very challenges. One challenging topic is redundancy resolution which is the process of determining the appropriate arm configuration that achieves some desired task space goals. Thus, redundancy resolution is about solving the inverse kinematics problem for redundant robotic arms. This thesis presents new redun-

dancy resolution schemes in the context of posture tracking for modular hyper-redundant serial arms.

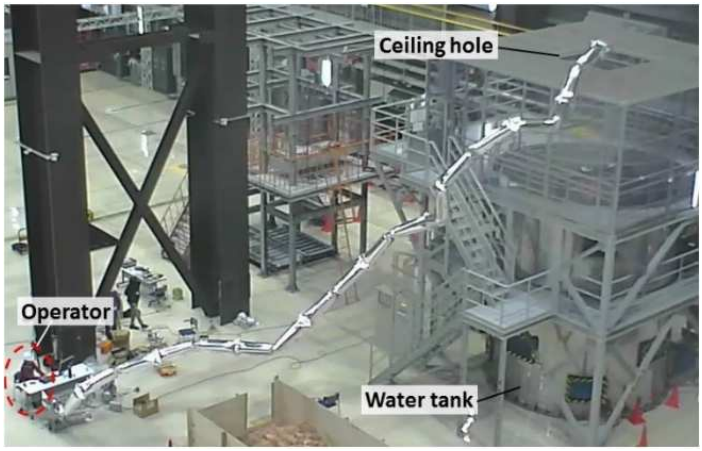


Fig. 1.2 Giacometti arm [36]
inspecting the interior of a water tank

1.1 Research Motivation

As seen in the previous section, redundancy is desirable in most manipulation tasks as a means to avoid singular regions in the workspace, increase the size of the workspace, and minimize the energy consumption of robotic arms. Redundancy is also very useful in obstacle avoidance for manipulation tasks in cluttered workspaces. In this case, popular methods are based on the potential fields that exert forces on the arm that can change its self-motion and move its posture away from obstacles [18, 37, 19]. Potential field methods are limited to obstacle avoidance. More versatile methods applicable to obstacle avoidance as well as whole-arm manipulation tasks are based on backbone curve methods [6, 7]. In these methods, hyper-redundant arms are modeled and controlled through the parametrization of continuous curves. Although efficient, this means that tasks need to be expressed through the same parametrization of the curve which can be difficult. For instance, in the modal backbone methods [6], it is necessary to have a different set of predefined mode functions for each kind of task the arm is to perform. This complexity associated with the specification of hyper-redundant tasks can also hinder their widespread adoption.

At the kinematic level, the specification of tasks for conventional non-redundant robotic arms performing conventional end-effector-focused tasks only needs to be concerned with the position, velocity, and acceleration of the end-effector, the rest of the arm is less of

a concern. As redundancy is introduced, the necessity to prevent entanglements and self-collisions arise. Performing manipulation tasks in cluttered environments also entails the specification of hyper-redundant tasks. For the Giacometti arm depicted in Figure 1.2, the operator guiding the arm through the hole at the top of the building must carefully monitor its posture as well. All this means that a control framework that features an intuitive specification of hyper-redundant tasks and that is applicable to most tasks various hyper-redundant tasks is needed.

A proposition for a simple hyper-redundant task specification scheme is depicted in Figure 1.3. The scene assumes a situation in which a broken prop inside a facility must be moved out of the way. Access through the door is not possible so the task must be performed from outside through a window. This is the kind of task tailored to hyper-redundant robots. An operator is instructed to use a hyper-redundant robot to perform the task. The task specification interface envisioned features a 3D graphical model of the arm that always displays the current configuration of the arm by using joint encoder data from the real arm. The interface also has a full 3D reconstruction of the workspace of the arm. This could be obtained by mapping the scene beforehand or by generating the map online by fitting the arm with some 3D sensing devices. The operator can then visually change the posture of the 3D model of the arm using spline control points along the 3D model of the arm. He specifies the task by providing a sequence of a few keyframe postures along the trajectory of the desired path of the posture. These keyframe postures are fed to a specialized posture tracking framework which produces joint velocities of intermediate postures needed to follow the path.

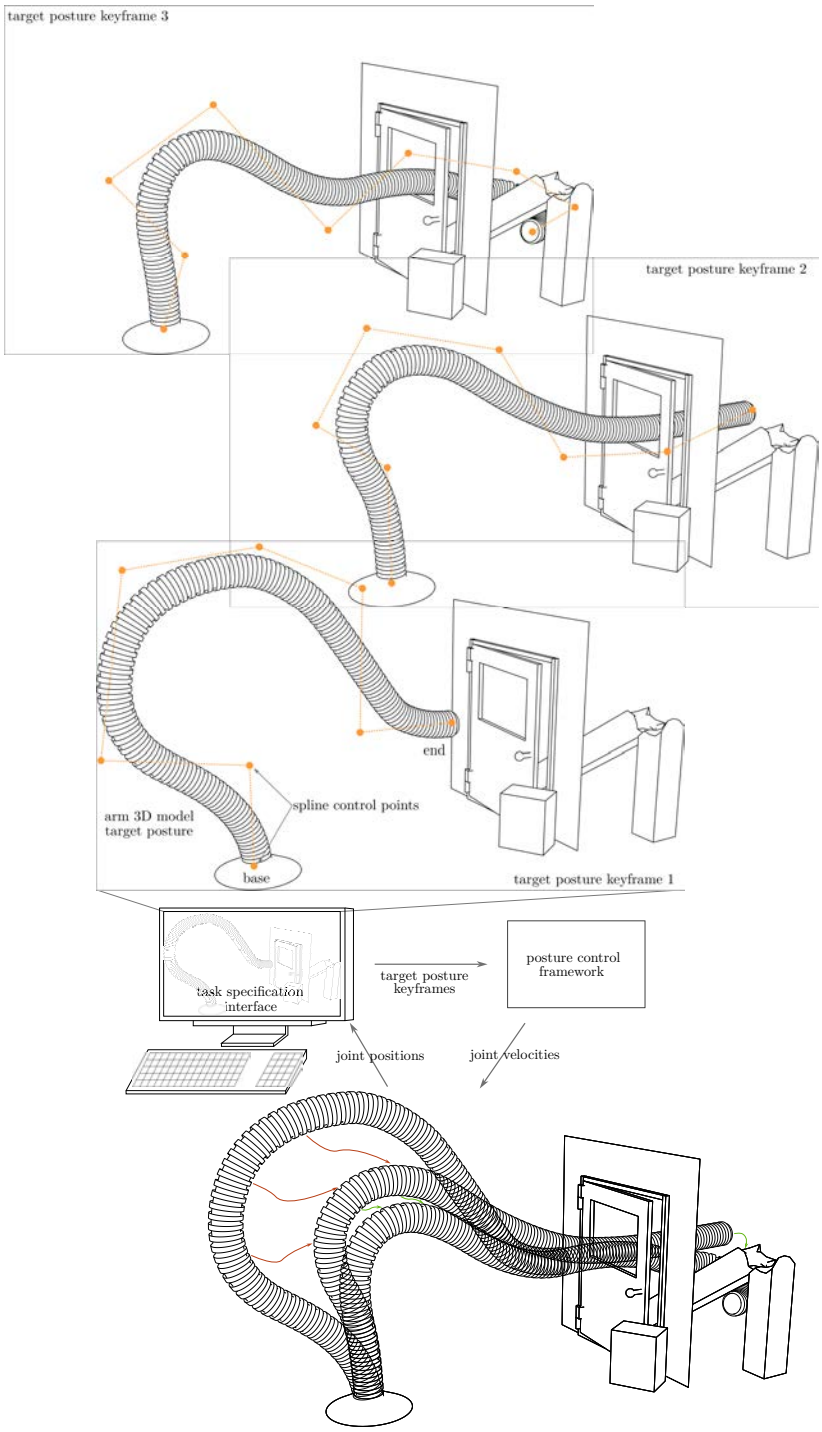


Fig. 1.3 Illustration of an intuitive hyper-redundant robot task specification system where the operator can visually construct tasks by creating virtual target posture keyframes

1.2 Research Proposal

The setup of Figure 1.3 discussed in the previous section could greatly ease the process of task specification for hyper-redundant arms. A control framework that is compatible with such kind of task specification is therefore necessary. The main requirement of the control framework is the ability to track any user-defined target postures. Since user-defined target postures could be near singularities, a robust control framework must be able to operate smoothly in and around singular postures.

The work in this thesis proposes a posture control framework for manipulation tasks of modular hyper-redundant serial arms that can (1) track arbitrarily set posture target curves and (2) operate smoothly in and around singular postures.

1.3 Related Works

Traditionally, the most important tasks robotic arms perform happen at the end-effector. This often involves appropriately moving and orienting the end-effector in the workspace. Therefore, the control of the pose (position and orientation) of the end-effector has always been the main focus of the formulations of robotics arms kinematics. A natural way of formulating it is by using the Jacobian matrix (Chapter 2) which is a linear mapping from joint space velocities to task space velocities [30]. A vast majority of Jacobian-based redundancy resolution methods are found in the literature. Initial works used the Jacobian pseudo-inverse [31] along with the projection of some arbitrary constraints on the null space [21] of the Jacobian matrix while others extended the Jacobian matrix by affixing to it a sufficient number of constraint equations to produce a square extended Jacobian matrix [3, 34, 5]. This use of the Jacobian matrix is computationally intensive for high degrees of redundancy so some works have tried to remedy this by segmenting the arm into two or three-link systems [12]. These methods are developed with the end-effector control in mind and are not well suited for the control of the posture or shape of the arm. Moreover, they perform poorly in the proximity of singular configurations and potentially introduce algorithmic singularities (Chapter 3).

With the recent advent of artificial neural networks, the more efficient recurrent neural network (RNN) based schemes are becoming mainstream. These methods reformulate the kinematics as a quadratic problem (QP) and use RNN derived methods like the projection neural network (PNN) [43, 16] or the zeroing neural network (ZNN) [17, 14] to solve the inverse kinematic problem. They offer lower computation cost, repetitive motion planning, and can handle inequality constraints and noise. RNN control schemes found in the literature are also concerned with the control of the pose of the end-effector.

As the degree of redundancy increases, the precise control of arm posture becomes more relevant than the control of the end-effector. This is because tasks that leverage redundancy do so by ultimately setting the adequate posture of the robot. For instance, a hyper-redundant arm operating in a cluttered environment would conform to collision-free postures [25, 6] while a hyper-redundant snake robot would need to fluctuate its posture in order to generate propulsion forces [39, 10]. Hyper-redundant arms could also modulate their posture so that the arm can grasp or manipulate objects with their whole bodies and not just their end-effectors [7]. Continuous curve-based control strategies emerged to enable more precise control of the arm posture. Continuous curve-based control strategies seek to align the robot to a desired target continuous curve. Notable examples are methods using backbone parametric curves [8]. The posture of the arm is represented by a spatial curve parameterized with a curvature, torsion, rotation, and extensibility which are each in turn expressed as a linear combination of predefined mode functions. Thus, redundancy resolution reduces to determining weights of the mode functions that satisfy given task constraints. The selection of an appropriate set of mode functions is not intuitive and attention is needed for modal singularities and degenerate modes.

Most of the control strategies above have joint space-based task specifications. The correlation between joint space quantities and the macroscopic posture of the robot is not trivial so specifying tasks can be difficult. Task space-based methods like [23, 22] exist and feature more intuitive task specifications in the task space.

A category of redundancy resolution works simplify the problem by reducing the arm to a set of non-redundant arms. The general strategy is to segment the arm into sections of lower complexity, the kinematics of these sections is analyzed individually and the results integrated to determine the kinematics of the entire arm. This has been termed redundancy decomposition in [13] where a subset of joints are kept constant to reduce the whole arm to a single non-redundant arm. Multiple combinations of joints can produce such an arm so each possible arm's non-redundant kinematics and dynamics are solved then the solution which optimizes a given criterion is adopted. Another work [1] in this category uses virtual layers which are an agglomeration of consecutive 3 or 4-link sub-arms. The inverse kinematics is solved in a cascading manner from the lowest to highest virtual layers. Other works like [29] use 3 sections to mimic the human hand and [15] subdivides a planar arm into 2-link sections which are geometrically fitted to a target curve.

Finally, works that explicitly control the position of points along the arm for the purpose of obstacle avoidance exits [26, 25]. These implementations keep a section of the arm away from obstacles and are not designed to deliberately control the overall posture of the arm.

Table 1.1 depicts a simple comparison between our proposed control scheme and those found in the literature.

Table 1.1 Comparison of this work and other hyper-redundancy resolution schemes. This work stands out as being a Jacobian based method that can truly control the posture of the arm.

	Jacobian Inverse Based	Continuous Curve Based	End-Effector Pose Control Focused	Arm Posture Control Focused	Task Space Task Specification	Redundancy Reduction Based	Control Through Control Points	Stable Near Singular Configurations	Repetitive Motion Planning	Inequality Constraints Incorporation
This Work	◎	○	◎	◎	◎	◎	◎	◎	○	×
[21, 3, 34, 5]	◎	×	◎	×	○	×	×	×	×	◎
[43, 16, 17, 14]	×	×	◎	×	○	×	×	×	◎	○
[25, 6, 8, 39, 10]	×	◎	◎	◎	×	×	○	×	○	×
[23, 22]	○	◎	◎	◎	◎	×	×	×	○	×
[29, 15, 13]	◎	×	◎	×	×	◎	×	×	×	×
[26, 25]	◎	×	◎	○	×	×	◎	×	×	

1.4 Thesis Organization

The remaining chapters of this thesis have two main groups: Chapters 2-3 review prerequisite topics relevant to the work presented in this thesis and Chapters 4-5 go through the details of the actual propositions of this thesis. Figure 1.4 shows a depiction of this organization.

Chapter 2 reviews Jacobian-based differential kinematics and their common use in redundancy resolution. Chapter 3 analyzes the nature of kinematic singularities and their consideration in the design and control of robotic arms. Chapter 4 details the proposition of the thesis which is a Jacobian-based posture control framework that avoids introducing singularities in its formulation and is capable of safely converging to singular target postures. Chapter 5 extends the concepts of Chapter 4 to improve convergence error and allow

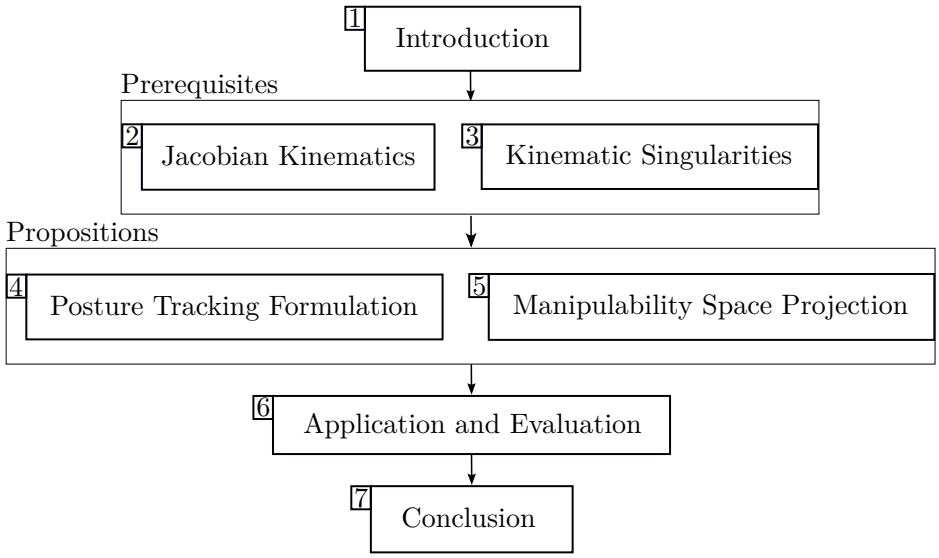


Fig. 1.4 Illustration of the organization of the thesis

posture tracking from singular initial postures. Chapter 6 applies the concepts to concrete examples of arm morphologies and conducts tracking simulations to assess the performance of the control schemes. Finally, Chapter 6 presents the conclusions of this thesis.

Chapter 2

Jacobian Kinematics

Robots are required to perform tasks in the real world, therefore it is natural to express their desired tasks in terms of the task space quantities such as end-effector position, velocity, and acceleration. Robot control, however, needs these tasks expressed as joint space-level quantities such as joint positions, velocities, and accelerations. The inverse kinematics deals with finding joint space quantities that realize some desired task space quantities.

Redundancy resolution refers to the process of solving the inverse kinematics problem for redundant robots. Unlike non-redundant robots where the inverse kinematics yields a finite number of joint space solutions, for redundant robots, an infinite number of solutions is possible. Redundancy resolution aims at selecting a unique solution which has some desirable characteristics. The inverse kinematics problem has traditionally been resolved through the use of the Jacobian differential kinematics. This chapter reviews the fundamentals concepts of this. Section 2.1 overviews the nature of the Jacobian matrix. Section 2.2 defines the Jacobian null-space and its uses in redundancy resolution. A discussion of versions of the Jacobian which incorporate additional tasks follows in Section 2.3 and Section 2.4.

2.1 Jacobian Matrix

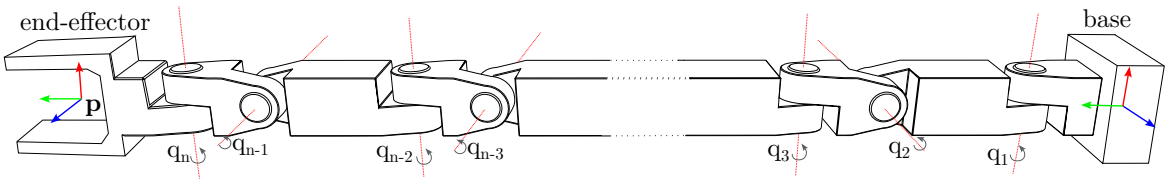


Fig. 2.1 An example hyper-redundant arm with *n* joints

Figure. 2.1 depicts an example of a hyper-redundant arm. $\{q_1, \ldots, q_n\}$ are joint variables and together form the joint space vector $\mathbf{q} = [q_1 \ldots q_n]^T \in \mathbb{R}^n$ which describe the configuration of the arm. The arm in the figure only uses revolute joints but a different arm morphology could use a mixture of revolute and prismatic joints in which case q_k describes an angular displacement for revolute joints and a linear displacement for prismatic ones. The Denavit-Hartenberg framework provides a unified framework for defining joint variables, link lengths, and various other offsets needed to formulate the kinematics of an arbitrary arm.

A frame appropriately affixed to the end-effector moves along with it in the workspace while an inertial frame affixed to the base is stationary. The end-effector frame expression relative to the base frame is described by the direct or forward kinematics of the arm and is only dependent on the joint space vector $\mathbf{q}_n$. The end-effector position and orientation are given by the end-effector frame origin and orientation respectively and it can be concisely expressed by the m dimensional task space vector $\mathbf{p}$. For a planar arm $\mathbf{p} = [x \;\; y \;\; \alpha]^T \in \mathbb{R}^3$ and for a spatial arm $\mathbf{p} = [x \;\; y \;\; z \;\; \alpha \;\; \beta \;\; \gamma]^T \in \mathbb{R}^6$ where $[x \;\; y \;\; z]^T$ is the position of the end-effector and $\{\alpha, \;\; \beta, \;\; \gamma\}$ is a minimal representation of the end-effector rotation using methods such as Euler angles or roll-pitch-yaw. For kinematically hyper-redundant arms, the dimension of the joint space is greater than that of the task space $n > m$. The pose vector relates to the joint vector through the following well-known non-linear equation (2.1).

$$\mathbf{p} = \mathbf{f}(\mathbf{q}) \quad \text{where} \quad \mathbf{q} \in \mathbb{R}^n, \;\; \mathbf{p} \in \mathbb{R}^m \tag{2.1}$$

The inverse kinematics is concerned with the reverse mapping from task space vectors to their corresponding joint space vectors. Although the forward kinematics is straightforward at the joint level, the inverse kinematics at the joint level is usually impossible. Therefore, for the purpose of solving the inverse kinematics, the first order differential of equation (2.1) in time t is used instead (2.2). The resulting $(m \times n)$ matrix is termed the task Jacobian matrix.

$$\dot{\mathbf{p}} = \frac{d\mathbf{p}}{dt} = \frac{d\mathbf{f}(\mathbf{q})}{d\mathbf{q}^T}\frac{d\mathbf{q}}{dt} = J(\mathbf{q})\dot{\mathbf{q}} \quad \text{where} \quad J(\mathbf{q}) \in \mathbb{R}^{m \times n} \tag{2.2}$$

The first 3 components of $\dot{\mathbf{p}}$ are the actual components of the linear velocity vector $\mathbf{v}$ of the end-effector. However, the last 3 components are not the components of the angular velocity vector ω of the arm but are some physically non-meaningful quantities representing the rate of change of the parameters characterizing the minimal representation of the end-effector orientation. Let $\dot{\mathbf{r}} = [\mathbf{v}^T \;\; \omega^T]^T$, then it can be shown that a matrix $T(t)$ exists based on the minimal representation of end-effector orientation such that equation (2.3) holds. $J_g(\mathbf{q})$ is termed the geometric Jacobian matrix which is independent from any mini-

mal representation of the end-effector orientation. Thus, the task Jacobian matrix $J(\mathbf{q})$ can be expressed as in equation (2.4).

$$\dot{\mathbf{p}} = T(t)\dot{\mathbf{r}} = T(t)J_g(\mathbf{q})\dot{\mathbf{q}} \tag{2.3}$$

$$J(\mathbf{q}) = T(t)J_g(\mathbf{q}) \tag{2.4}$$

2.2 Jacobian Null Space

The inverse kinematics problem can be solved by inverting equation (2.2). Since J is a low-rectangular matrix for hyper-redundant arms, its inverse is not defined so its generalized inverse is used. If J has full row rank then its pseudo-inverse matrix J^+ is a suitable generalized inverse and is given by equation (2.5) according to the Moore-Penrose conditions [2]. Hence, the general solution is written as in equation (2.6).

$$J^+ = J^T(JJ^T)^{-1} \tag{2.5}$$

$$\dot{\mathbf{q}} = J^+\dot{\mathbf{p}} + (I - J^+J)\boldsymbol{\kappa} \tag{2.6}$$

I is an $(n \times n)$ identity matrix and $(I - J^+J)$ is the orthogonal projection matrix in the null space of the Jacobian matrix. The null space of the Jacobian matrix is defined as the set of joint space velocities that do not produce task space velocities. The null space is the space spanned by $(n-m)$ linearly independent n dimensional column vectors of $(I - J^+J)$. $\boldsymbol{\kappa}$ is an arbitrary n dimensional joint space vector that is projected to the null space to produce null space joint velocities $(I - J^+J)\boldsymbol{\kappa}$ which do not generate end-effector velocities. In practice, $\boldsymbol{\kappa}$ is used to achieve a secondary task once the primary task of the end-effector placement is achieved.

2.2.1 Optimization of an Objective Function

A common use for $\boldsymbol{\kappa}$ is for locally optimizing a given objective function $h(\mathbf{q})$. This is done by projecting the gradient of the objective function in the null space [24]. $\boldsymbol{\kappa}$ is therefore expressed according to equation (2.7) and the objective function is minimized when $k < 0$ and maximized when $k > 0$.

$$\boldsymbol{\kappa} = k\frac{dh(\mathbf{q})}{d\mathbf{q}^T} \quad \text{where} \quad k \in \mathbb{R} \tag{2.7}$$

A common choice for $h(\mathbf{q})$ is one of the measures for manipulability discussed in Section 3.3 in order to avoid local singular configurations. Another choice for $h(\mathbf{q})$ is the norm

of the joint velocity vector as an attempt to move the arm away from configurations with large joint velocities which typically occur in proximity of singular postures. $h(\mathbf{q})$ can also be formulated to keep joint values within their mechanical limits by maximizing the expression in equation (2.8) where $[\underline{q}_i,\ \overline{q}_i]$ is the range of joint i. A use of $h(\mathbf{q})$ as an artificial potential function for the purpose of obstacle avoidance is also found in [18].

$$h(\mathbf{q}) = \frac{1}{2}\sum_{i=1}^{n}\left((q_i - \underline{q}_i)^{-2} + (q_i - \overline{q}_i)^{-2}\right) \tag{2.8}$$

2.2.2 Realization of Task Constraints

When additional task constraints can be formulated in the form of equation (2.9) then κ can take the form of equation (2.11) [40]. An Example of such usage is [26] which uses the kinematics at a fixed point along the arm as constraints and succeeds in keeping that section of the arm away from obstacles in the workspace.

$$\mathbf{p}_c = \mathbf{f}_c(\mathbf{q}) \in \mathbb{R}^{\alpha} \quad \text{where} \quad \alpha \leq n - m \tag{2.9}$$

$$J_c = \frac{d\mathbf{f}_c(\mathbf{q})}{d\mathbf{q}^T} \in \mathbb{R}^{\alpha \times n} \tag{2.10}$$

$$\kappa = \left(J_c(I - J^+J)\right)^+ (\dot{\mathbf{p}}_c - J_c J^+ \dot{\mathbf{p}}) \tag{2.11}$$

2.3 Extended Jacobian Matrix

The extended Jacobian introduced in [3] constructs a constraint equation from the projection of the gradient of some objective function to be optimized on the null space of the Jacobian matrix. It exploits the fact that at extremum values of the objective function, the projection of the gradient of the objective function does not produce joint velocities (2.12). $(I - J^+J)_{base}$ has $(n-m)$ columns of n dimensional vectors spanning the null space of a non singular Jacobian matrix.

$$\mathbf{f}_c(\mathbf{q}) = \left(\frac{dh(\mathbf{q})}{d\mathbf{q}^T}(I - J^+J)_{base}\right)^T = \mathbf{0} \in \mathbb{R}^{n-m} \tag{2.12}$$

A squared extended Jacobian matrix is then obtained by stacking the task Jacobian matrix and the constraints Jacobian matrix J_c (2.10) as shown in equation (2.13). The inverse

kinematics is solved by taking the inverse of the extended Jacobian matrix (2.14).

$$J_e\dot{\mathbf{q}} = \begin{bmatrix} J \\ J_c \end{bmatrix} \dot{\mathbf{q}} = \begin{bmatrix} \dot{\mathbf{p}} \\ \mathbf{0} \end{bmatrix} \tag{2.13}$$

$$\dot{\mathbf{q}} = J_e^{-1} \begin{bmatrix} \dot{\mathbf{p}} \\ \mathbf{0} \end{bmatrix} \tag{2.14}$$

2.4 Augmented Jacobian Matrix

The methods shown above attempt to realize additional tasks after the primary task of end-effector positioning is achieved. In contrast, augmented Jacobian methods [33, 32, 11, 35] attempt to realize both tasks simultaneously. Let us consider the task constraints defined in equation (2.9) and (2.10). The augmented Jacobian matrix J_a is formed by simply stacking the task Jacobian matrix and the constraints Jacobian matrix as shown in equation (2.15). Note that the augmented Jacobian can be considered as a general form of the extended Jacobian matrix. A pseudo-inverse formulation similar to equation (2.5) is used to express the inverse kinematics (2.16).

$$J_a\dot{\mathbf{q}} = \begin{bmatrix} J \\ J_c \end{bmatrix} \dot{\mathbf{q}} = \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{p}}_c \end{bmatrix} \tag{2.15}$$

$$\dot{\mathbf{q}} = J_a^{+} \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{p}}_c \end{bmatrix} + (I - J_a^{+}J_a)\boldsymbol{\kappa} \tag{2.16}$$

2.5 Discussion

This chapter reviewed Jacobian matrix based differential kinematics for redundancy resolution of robotic arms. Redundancy is consumed by formulating a secondary task consisting of realizing of a set of constraints equations or optimizing an objective function. Null-space and extended Jacobian based formulations attempt to achieve the secondary task after the primary task of end-effector placement is fully achieved. On the other hand, augmented Jacobian methods attempt to achieve both primary and secondary tasks at once and control formulations presented in Chapter 4 and Chapter 5 are an extension of this. Next, Chapter 3 uses these formulations to expose singularities originating from the geometry of the arm as well as singularities introduced by the control formulations themselves.

Chapter 3

Kinematic Singularities

Kinematic singularity as relevant to redundancy resolution, occurs when the final Jacobian matrix used to compute the inverse kinematics of the robot is rank deficient. This is undesirable because the inverse kinematics equations break down at singularities. Moreover, the arm motion usually degenerates at singularities and is no longer able to produce motion in one or several directions in the task space. The arm also tends to become unstable in proximity of a singularity as unacceptably large joint space velocities are commanded as the arm approaches a singularity. Therefore, taking in account singularities is a key aspect in the design, control, and motion planning of robotic arms.

Several factors can cause this loss of rank of the Jacobian matrix. Section 3.1 discusses the fundamental causes of kinematic singularities. Section 3.3 looks into ways of quantifying how close a given arm configuration is to a singular configuration.

3.1 Types of Kinematics Singularities

3.1.1 Geometric Singularities

Geometric singularities arise from the geometric Jacobian J_g (2.3) being rank deficient. This type of singularity is specific to the chosen morphology of the robot. They highlight the mechanical limitations of the robot mechanisms. Some obvious singular configurations that fall in this category are those at the boundary of the workspace. These are termed unavoidable singularities [4] as opposed to avoidable singularities that occur within the workspace and for which redundancy can be leveraged to avoid those configurations.

3.1.2 Representation Singularities

This type of singularity occurs because of the choice of $T(t)$ when constructing the task Jacobian matrix (2.4). $T(t)$ encodes the minimal representation of the orientation of the end-effector. Since the task Jacobian matrix is the product of $T(t)$ with the geometric Jacobian matrix, J will lose rank whenever $T(t)$ and/or J_g do. Depending on the minimal representation adopted, the number and configurations where these singularities appear differ.

3.1.3 Algorithmic Singularities

Algorithmic singularities are introduced by the particular formulation used to solve the arm kinematics. For instance, the augmented Jacobian matrix J_a (2.15) is known to introduce algorithmic singularities because J_a may be rank deficient while the task Jacobian matrix J is full rank. This happens when the constraints Jacobian matrix J_c is rank deficient and/or the stacked matrix is rank deficient. This is often a sign that the end-effector positioning task and constraints task conflict with each other and can not be achieved simultaneously [3].

3.2 Anatomy of a Singular Matrix

The singular value decomposition (SVD) can provide further insights into the properties of the Jacobian matrix. The SVD is given by equation (3.1) where $\{\mathbf{u}_1,\ldots,\mathbf{u}_m\}$ are m-dimensional left-singular vectors, $\{\mathbf{v}_1,\ldots,\mathbf{v}_n\}$ are n-dimensional right-singular vectors, and $\{\sigma_1,\ldots,\sigma_m\}$ are scalar singular values. $diag_{m,n}(\mathbf{a})$ is the $(m\times n)$ diagonal rectangular matrix with the components of $\mathbf{a}$ in its diagonal. Singular values are sorted in descending order $(\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m)$.

$$J = U\Sigma V^T = \sum_{i=1}^{m} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{3.1}$$

$$\text{where} \quad U = [\mathbf{u}_1 \ldots \mathbf{u}_m] \in \mathbb{R}^{m\times m}, \;\; V = [\mathbf{v}_1 \ldots \mathbf{v}_n] \in \mathbb{R}^{n\times n}$$

$$\text{and} \quad \Sigma = diag_{m,n}([\sigma_1 \ldots \sigma_m]) \in \mathbb{R}^{m\times n}$$

Let the rank of the Jacobian matrix be denoted as r, then $r \leq m$. The SVD demonstrates that any task space velocity $\dot{\mathbf{p}}$ mapped from a joint space velocity $\dot{\mathbf{q}}$ is a linear combination of the first r left-singular vectors (3.2). This means that when the Jacobian matrix is full rank ($r = m$), any task space velocity in $\mathbb{R}^m$ can be achieved, whereas only task velocities in

a subspace $\mathbb{R}^r$ of $\mathbb{R}^m$ are achievable when the Jacobian is rank deficient ($r < m$).

$$\dot{\mathbf{p}} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \dot{\mathbf{q}} = \sum_{i=1}^{r} k_i \mathbf{u}_i \quad \text{where} \quad k_i = \sigma_i \mathbf{v}_i^T \dot{\mathbf{q}} \in \mathbb{R}$$

The SVD also shows that a joint space velocity ($\dot{\mathbf{q}} = \lambda \mathbf{v}_j$) in the direction of $\mathbf{v}_j$ leads to a task space velocity in the direction of $\mathbf{u}_j$ (3.2). This means that task space velocities in the direction of $\mathbf{u}_m$ become harder to achieve as the Jacobian matrix approaches singularities and σ_m gets smaller. At singularities where the rank falls and σ_m is null, task space velocities along $\mathbf{u}_m$ are no longer achievable, and the dimension of the range space of the Jacobian matrix shrinks while that of its null space grows by the same amount.

$$\dot{\mathbf{p}} = \lambda \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \mathbf{v}_j = \lambda \sigma_k \mathbf{u}_k \tag{3.2}$$

3.3 Quantifying Proximity to a Singularity

As singularities play an essential role in robots kinematics, means of quantifying closeness of a configuration to a singularity are necessary in order to devise ways to construct and operate robots with singularity in mind. The commonly used measures of proximity to singularities can all be derived from the manipulability ellipsoid [41] which is formed by the subset of all realizable task space velocities originating from joint velocities $\dot{\mathbf{q}}$ such that $|\dot{\mathbf{q}}|^2 \leq 1$. These measures can be formulated using the SVD of the Jacobian matrix seen in Section 3.2.

3.3.1 Manipulability Measure

Since singularities are characterized by a loss of rank of the Jacobian matrix, a good measure for proximity to a singularity is the determinant which can be generalized to rectangular matrices as the manipulability measure w [41] in equation (3.3). The manipulability measure can also be expressed as the product of singular values of the Jacobian matrix which is just a scalar multiple of the volume of the manipulability ellipsoid.

$$w = \sqrt{|JJ^T|} = \prod_{i=1}^{m} \sigma_i \tag{3.3}$$

3.3.2 Condition Number

Another measure is the condition number k [20] in equation (3.4). It better quantifies the ability to uniformly produce motion in all directions of the task space. It measures the closeness of the manipulability ellipsoid to a unit sphere.

$$k = \frac{\sigma_1}{\sigma_m} \tag{3.4}$$

3.3.3 Minimum Singular Value

A more useful measure for proximity to a singularity is the minimum singular value σ_m [20] of the Jacobian matrix. It is often considered as the most effective measure since it is more sensitive to some configuration changes that would not affect the values of the manipulability measure or the condition number.

3.4 Discussion

This chapter presented the concept of kinematic singularity. A discussion of the different types of singularities and characteristics of the Jacobian matrix at singularities was done. This enabled the quantification of the distance to singularities. The understanding of how and where singularities occur is crucial to the control and motion planning of robotic arms and control formulations developed in Chapter 4 and Chapter 5 are designed to not introduced any algorithmic singularities and to operate stably in proximity of geometric singularities.

Chapter 4

Posture Tracking Formulation

The approach to controlling the posture of the arm is to identify strategic points along the arm and manage to control their velocities so to move the posture toward the desired target posture (Figure 4.1). An analogy of this approach is thinking of the hyper-redundant arm as a flexible puppet with strings attached all along its body, one would pull a different set of strings a different amount at different times to conform the arm to a desired shape.

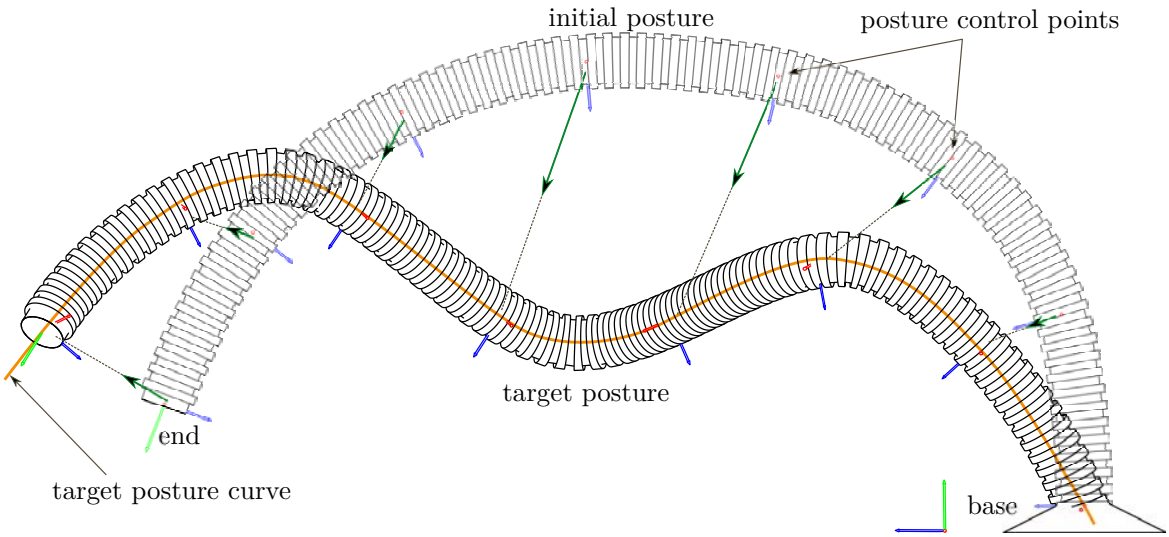


Fig. 4.1 Illustration of the overall strategy to posture control based on changing the posture by moving predefined points along the arm

This is achieved through several steps. First, in Section 4.1, the arm is segmented into modules of identical morphology, and the posture control points are taken at the boundary of those modules. In Section 4.2, the target position of each posture target point on the user-supplied target posture curve is found. Then, the sets of control points eligible for

control are determined in Section 4.3 and a single set is adopted for control in Section 4.4 and finally, a posture control matrix is constructed from it in Section 4.5.

4.1 Arm Segmentation into Modules

Hyper-redundant arms considered in this work have a modular structure and they can be segmented into smaller sections of identical morphology. A section ideally does not have redundancy and its kinematics is well understood. These sections are referred to as modules and controlling the posture of the hyper-redundant arm is reduced to controlling the end-effectors of these modules which are termed posture control points.

A segmentation is adequate if the base of the module it produces when placed at any point on the expected posture target curve in the tangent direction of the curve, can have its end-effector positioned on the target curve i.e. the reachable manifold of the module must intersect any arbitrary target curve passing through the base of the module. This is a basic requirement for the arm to be physically able to conform to the desired posture. The link lengths and joint limits of the module along with the maximum curvature and torsion of the target curve are used to do this analysis. Furthermore, a suitable module must supply the following three quantities that are local to the module.

1. A forward kinematics solver $FK(\mathbf{q})$ that maps the joint positions of the module to its local end-effector position.

2. An inverse kinematics solver $IK(\mathbf{p})$ which is the inverse function of $FK(\mathbf{q})$.

3. A target position resolver $TP(\mathbf{s})$ which determines the local target position of the module end-effector on the posture target curve. This is achieved by determining the intersecting point between the target posture curve and the workspace envelope of the module.

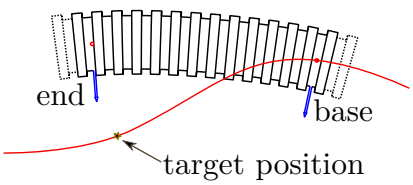


Fig. 4.2 An isolated module with the target curve passing through its base

Let $\{\mathbf{p}_b, E_b\}$ and $\{\mathbf{p}_e, E_e\}$ the coordinate system associated with the base and end-effector of a module respectively where $\mathbf{p} \in \mathbb{R}^3$ is the coordinate of the origin and $E \in \mathbb{R}^{3\times3}$ is the

orientation matrix. Modules are chained together in such way that $\{\mathbf{p}_b, E_b\}$ of a given module is $\{\mathbf{p}_e, E_e\}$ of the module immediately preceding it. Let m be the number modules and $k \in \{1,\ldots,m\}$ the index of the k^{th} module. The inertial coordinate system $\{\mathbf{0}, I\}$ is fixed at the base of module$_1$. The end-effector of a module$_k$ has a position $\mathbf{p}_k$ and an orientation E_k relative to that inertial coordinate system. The forward kinematics of module$_k$ is formulated in equation (4.1) using $\{\mathbf{p}_k^{(k-1)}, E_k^{(k-1)}\}$ which are the end-effector position $\mathbf{p}_k$ relative to $\{\mathbf{p}_{k-1}, E_{k-1}\}$ and the orientation of E_k relative to E_{k-1}. $\{\mathbf{p}_k^{(k-1)}, E_k^{(k-1)}\}$ are provided by the forward kinematics solver $FK(\mathbf{q})$ of module$_k$.

$$\begin{cases} \mathbf{p}_k = E_{k-1}\mathbf{p}_k^{(k-1)} + \mathbf{p}_{k-1} \\ E_k = E_{k-1}E_k^{(k-1)} \end{cases} \tag{4.1}$$

4.2 Posture Control Point Targets Acquisition

Target Postures are initially available in the form of continuous spline curves. A spline curve is defined as a piecewise polynomial of degree p which is $(p-1)$ times continuously derivable at the junction points of adjacent polynomial pieces. $(p = 3)$ Cubic B-splines are used in this work. The target posture is derived from the target curve and is just the set of target positions of each posture control point on the target curve. An explicit analytic form of the target curve is not required as a set of points appropriately sampled along the curve is a sufficient approximation used in this work.

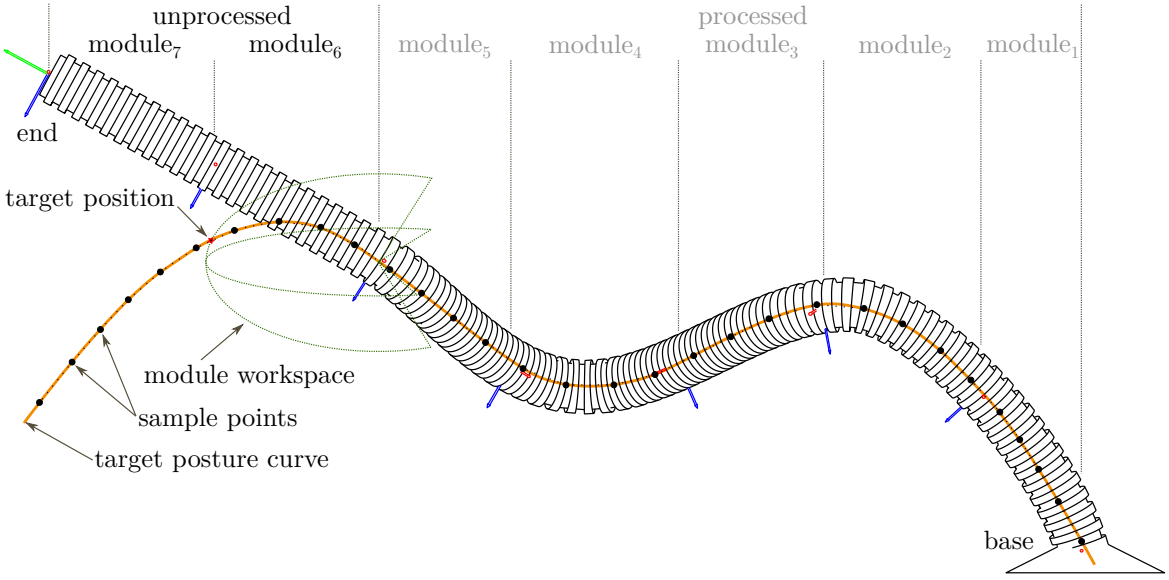


Fig. 4.3 Acquisition of the posture control point target positions by fitting the arm to the posture target curve using module kinematics

A posture control point target is therefore the target position of the end-effector of a module on the target curve relative to the inertial frame. The forward kinematics solver $FK(\mathbf{q})$, the inverse kinematics solver $IK(\mathbf{p})$, and the target position resolver $TP(\mathbf{s})$ of Section 4.1 along with the $(n+1)$ target curve sample points are combined in Algorithm 1 to determine each of the posture control point target positions for an arm with m modules. $\mathbf{s}_i \in \mathbb{R}^3$ is the position of the i^{th} curve sample point relative to the inertial coordinate system $(\mathbf{s}_i : i \in \{0,\ldots,n\})$.

Algorithm 1 Posture control point targets acquisition algorithm

Require: $m > 1$ **and** $n > 2$ **and** $\{\mathbf{s}_i\} : i \in \{0,\ldots,n\}$
1: $i \leftarrow 0,\ k \leftarrow 0$
2: $\mathbf{p}_0 \leftarrow \mathbf{s}_0,\ E_0 \leftarrow E_t$ {target position and orientation}
3: **for** $k = 0$ **to** m **do**
4: $\quad t_k \leftarrow 0$
5: $\quad$ **while** $i < n$ **do**
6: $\quad\quad \mathbf{s}_{i+1}^{(k)} \leftarrow E_k^{-1}(\mathbf{s}_{i+1} - \mathbf{p}_k)$ {$\mathbf{s}_{i+1}$ relative to $\{\mathbf{p}_k, E_k\}$}
7: $\quad\quad \mathbf{p}(t) \leftarrow \mathbf{s}_i^{(k)} + (\mathbf{s}_{i+1}^{(k)} - \mathbf{s}_i^{(k)})t$ {sample line segment$_i$}
8: $\quad\quad t_k \leftarrow TP(\mathbf{p}(t))$ {t of intersection between module$_k$ workspace envelope and the line segment$_i$}
9: $\quad\quad$ **if** $t_k \in]0,\ 1]$ **then**
10: $\quad\quad\quad$ **break**
11: $\quad\quad$ **end if**
12: $\quad\quad i \leftarrow i + 1$
13: $\quad$ **end while**
14: $\quad \mathbf{p}_{k+1}^{(k)} \leftarrow \mathbf{s}_i^{(k)} + (\mathbf{s}_{i+1}^{(k)} - \mathbf{s}_i^{(k)})t_k$ {local target position}
15: $\quad \mathbf{q}_k \leftarrow IK(\mathbf{p}_{k+1}^{(k)})$ {target position to joint positions}
16: $\quad E_{k+1}^{(k)} \leftarrow FK(\mathbf{q}_k)$ {joint angles to orientation matrix}
17: $\quad E_{k+1} \leftarrow E_k E_{k+1}^{(k)}$ {update rotation matrix}
18: $\quad \mathbf{p}_{k+1} \leftarrow E_k \mathbf{p}_{k+1}^{(k)} + \mathbf{p}_k$ {update end-effector position}
19: **end for**

The algorithm works by sequentially placing the modules on the target curve starting from the module at the end-effector of the arm. At each iteration, the target resolver of the module being processed finds the intersection of the workspace envelope of that module with a target curve segment expressed relative to the base of the module. The inverse kinematics solver of the module is then used to extract the joint positions of the module which are subsequently used by the forward kinematics solver of the module to compute the absolute position and orientation of the end-effector of the module needed in the following iteration.

This process is repeated from module_1 to module_m and all posture target points are found. An illustration of this process is depicted in Figure 4.3.

4.3 Candidate Control Sets Enumeration

Once the posture control point target positions are known, an adequate control scheme needs to guide the posture control points to their corresponding target positions. A task augmentation scheme similar to the augmented Jacobian in Section 2.4 could be used. Basically, the velocity constraint of each posture control point would be considered as an independent task, and resulting Jacobian matrices are stacked into a posture control Jacobian matrix. Such a naive approach would be plagued with algorithmic singularities (Section 3.1.3) as some velocity constraints would conflict with each other and could not be realized simultaneously. A workaround is to use at each iteration time a subset of posture control points whose velocity constraints are guaranteed to not conflict. Such a set of posture control points is one out of many other viable sets termed candidate control sets. The list of candidate control sets differs at each configuration of the arm.

A candidate control set must have one fundamental property: its derived posture control matrix must be far from singularities. Therefore, to find what combinations of posture control points qualify as candidate control sets, one could construct their corresponding posture control matrices and compute one of the measures of the distance to singularities in Section 3.3. However, that would be inefficient so a module-based analysis can be used to infer the manipulability of the posture control matrix derived from an arbitrary set of posture control points.

Observing the posture control Jacobian matrix constructed from stacking the Jacobian matrix of an arbitrary set of posture control points, it is clear that the posture control Jacobian matrix is rank deficient if any of the posture control point Jacobian matrices are rank deficient. This means that the velocity constraints of each posture control point in a candidate control set by itself must be achievable. Furthermore, when any two posture control point Jacobian matrices are full rank but their stacked matrix is not, then the posture control Jacobian matrix will be rank deficient. This means that the velocity constraints of two control points in a candidate control set must be achievable simultaneously without conflicts. The criteria for a posture control set to be considered a candidate control set are captured by the following two conditions. The posture control matrix derived from a candidate control set is guaranteed to be full rank. Figure 4.4 shows an example of candidate control set and the sections of the arm subject to each condition.

- *Absolute Manipulability Condition*
  This condition ensures that the Jacobian matrix of every posture control point involved in the set has a high manipulability index. It is tested by computing a manipulability index of the Jacobian matrix of every control point in the set. A property of this condition is that if a posture control point of a given index clears the condition then any other posture control point of a higher index also automatically clears the condition. The proof of this condition is trivial.

- *Relative Manipulability Condition*
  This condition ensures that the stacked matrix of the Jacobian matrix of any two posture control points in the set has a high manipulability index. The absolute manipulability condition implies that testing this condition on adjacent posture control points in the set is sufficient to clear the condition. The proof of this condition for the planar case is provided in Appendix A.

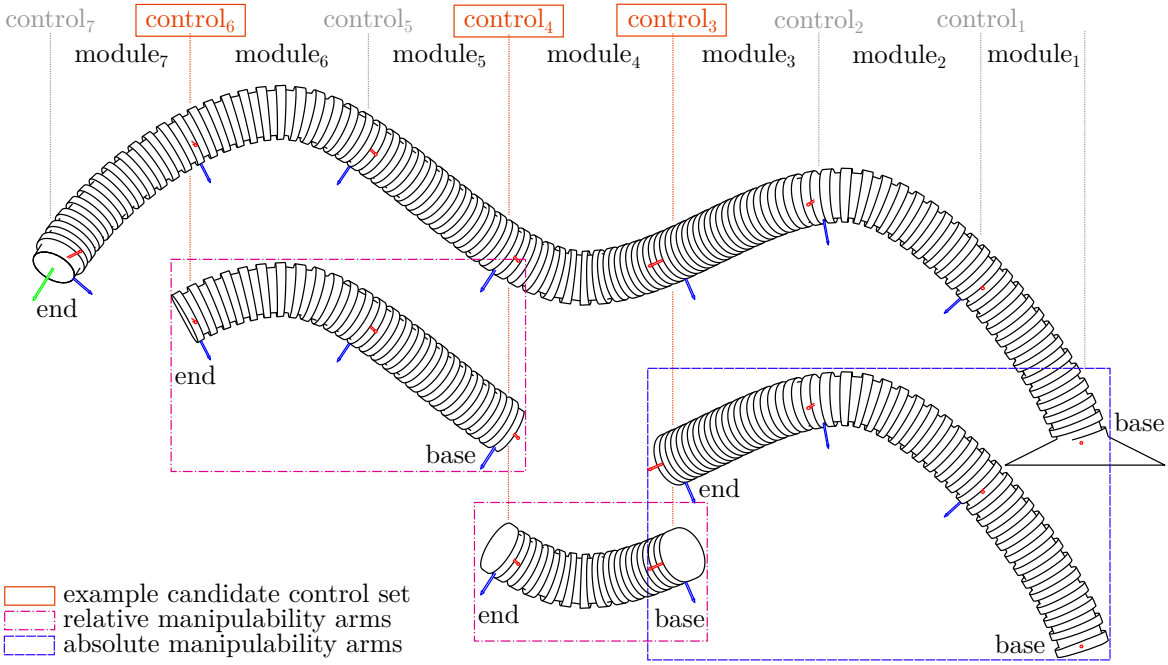


Fig. 4.4 Example of a candidate control set and the sections of the arm subject to the absolute and relative manipulability conditions

To more efficiently test the above conditions, the local Jacobian matrix of modules is used to recursively compute the Jacobian matrices involved as shown in equation (4.2). This equation reduces the amount of computations that otherwise would have been required to

compute the Jacobian matrix between any arbitrary two arbitrary control points.

$$\frac{d\mathbf{p}_{k+1}}{dq_u} = \frac{d\mathbf{p}_k}{dq_u} + E_0 \prod_{j=1}^{k} E_j^{(j-1)} \mathbf{p}_{k+1}^{(k)} \tag{4.2}$$

The manipulability index used in this work is a variation of the manipulability ellipsoid volume presented in Section 3.3. Indeed, the manipulability ellipsoid of the Jacobian matrix J_k of the k^{th} posture control point has a volume $\bar{w}_k$ which is proportional to the product of the singular values $\{\sigma_{ki}\}$ of J_k (4.3). The manipulability index w_k is then defined as the normalized value of $\bar{w}_k$ shown in equation (4.4). w_k is generalized as w_k^s which is the manipulability index of the k^{th} posture control point relative to the s^{th} posture control point. That is the manipulability index of the end-effector of module$_k$ relative to the base of module$_{s+1}$.

$$\bar{w}_k \approx \sqrt{\left|J_k J_k^t\right|} = \prod_{i=1}^{3} \sigma_{ki} \tag{4.3}$$

$$w_k = (\max_{i=\{1,2,3\}} \sigma_{ki})^{-3} \bar{w}_k \tag{4.4}$$

The enumeration of all candidate sets of posture control points that have cleared both the absolute and relative manipulability conditions stated above is performed. This is done according to the following iterative algorithm that minimizes duplicate operations. Assuming we have already found all candidate sets from the posture control point of index k up to the end of the arm and we need to find next all candidate sets from the posture control point of index $k-1$. A relative manipulability check is performed on the manipulability index of sections of the arm delimited by each of the following pairs of posture control point indices $\{\{k-1,k\},\{k-1,k+1\},\ldots,\{k-1,m\}\}$. This corresponds to manipulability indices $\{w_k^{k-1}, w_{k+1}^{k-1}, \ldots, w_m^{k-1}\}$. If a manipulability index w_i^{k-1} from this list is acceptable then the already known candidate sets starting from the posture control point index i are appended to the list of candidate sets starting at index $k-1$ after prepending control point $k-1$ to each of these later sets. Figure 4.5 shows an illustration of this process for an arm with 3 modules ($m = 3$).

4.4 Active Control Set Selection

The criteria for selecting which among the available candidate control sets to use ultimately dictates the characteristics of the resulting motion of the arm. For instance, if the utmost stability of the posture control matrix is prioritized then the candidate set that constructs the posture control matrix with the highest manipulability index should be used. Since all

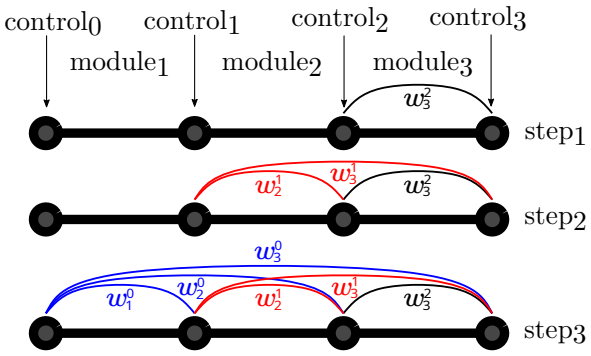


Fig. 4.5 Enumeration process of the candidate control sets of an arm of 3 modules

the candidate control sets already construct a sufficiently stable posture control matrix, a criterion that prioritizes the candidate control set with the highest posture error is instead used in this work. This criterion takes the form of a score assigned to each candidate control set. The candidate control set that has the highest score is the one used to derive the posture control matrix for that iteration step and is termed the active control set. For the k^{th} candidate control set containing N posture control points of indices $\{s_1, s_2, \ldots, s_N\}$, the score assigned is defined as the cumulative error χ_k between its end-effector positions and their corresponding target positions $\mathbf{c}_{s_i}$ (4.5). For simplification of notation, $\mathbf{p}_{s_i}$ without an explicit reference frame is equivalent to $\mathbf{p}_{s_i}^{(0)}$.

$$\chi_k = \sum_{i=1}^{N} \left\| \mathbf{p}_{s_i} - \mathbf{c}_{s_i} \right\| \tag{4.5}$$

4.5 Inverse Kinematics Formulation

The active control set is used to construct the posture control matrix which is the matrix obtained by simply stacking the Jacobian matrix of each control point in the active set. Each of these Jacobian matrices is adequately padded with zero column vectors in order to match the size as the widest Jacobian matrix within the set. For a given active set $\{s_1, s_2, \ldots, s_N\}$, the following $(3N \times 2s_N)$ dimension posture control matrix J is obtained along with a corresponding $2s_N$ dimension velocity vector $d\mathbf{r}$ (4.6).

$$J = \begin{bmatrix} J_{s_1} & 0 & \cdots\cdots & 0 \\ & J_{s_2} & 0 & \cdots & 0 \\ & & \ddots & & \\ & & & J_{s_N} & \end{bmatrix} \quad d\mathbf{r} = \begin{bmatrix} \mathbf{p}_{s_1} - \mathbf{c}_{s_1} \\ \mathbf{p}_{s_2} - \mathbf{c}_{s_2} \\ \vdots \\ \mathbf{p}_{s_N} - \mathbf{c}_{s_N} \end{bmatrix} \tag{4.6}$$

The posture control matrix obtained through the above process is guaranteed full-row rank. Therefore, the inverse kinematics problem can be resolved with the conventional pseudo-inverse formulation (Section 2.2) in equation (4.7). The first term on the right guarantees that active posture control points (those in the active control set) are moved such that their respective task space errors are reduced. However, other control points that are not part of the active control set could move in a way that increases the posture error of the entire arm. Thus, the second term on the right projects the gradient of the posture error onto the null space of the Jacobian ensuring that the posture error decreases with each iteration. K is a constant diagonal matrix containing the gains for each posture control point and α is a negative scalar governing how fast the posture error is minimized.

$$d\mathbf{q} = J^t(JJ^t)^{-1}Kd\mathbf{r} + \alpha\left(I - J^t(JJ^t)^{-1}J\right)\sum_{i=s_1}^{s_N} -J_i^t(\mathbf{p}_i - \mathbf{c}_i) \tag{4.7}$$

4.6 Special Case of a Planar Arm

In the case of a planar hyper-redundant arm, if the choice of the module morphology of the arm is done according to the prescriptions of Section 4.1, then a 1-DOF module should be enough. Indeed, a 1-DOF revolute module is able to position its end-effector on any arbitrary planar curve passing through its base. A 1-DOF module however is unable to track 2-DOF task velocities and would inevitably produce singular posture control matrices whenever 2 adjacent posture control points are used (Appendix A). Therefore, a better alternative is using 2-DOF modules like in the case of the spatial arm. However, unlike in the case of spatial arms where 2-DOF modules were used uniformly throughout the arm, in the case of planar arms a combination of 2-DOF and 3-DOF modules are used instead. The rationale for this choice is to be able to use as posture control points joints that would normally be unavailable if a uniform 2-DOF module segmentation was used. This strategy leads to a closer fit to the target curve and could also be applied to spatial arms in the future.

4.6.1 Planar Arm Candidate Control Sets Enumeration

Due to the presence of 2-DOF and 3-DOF modules, the enumeration of the candidate control sets for the planar case is done differently than for the spatial case.

1. Once the region of the arm that will be subject to posture control is determined (from control point i to control point $j = i+9$ in Figure 4.6) then all possible 2-DOF and 3-DOF modules are extracted and their manipulability indices computed.

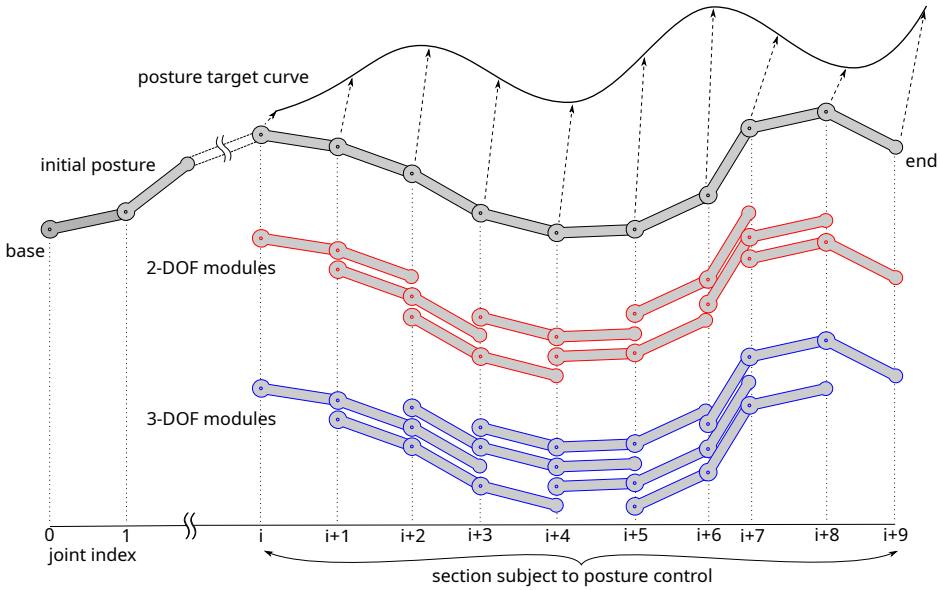


Fig. 4.6 Planar arm segmentation in 2-DOF and 3-DOF modules

2. The modules are combined together to form a tree in order to explore all possible sets of posture control points. An example of such a tree is illustrated in Figure 4.7 where the nodes represent posture control points and edges represent 2-DOF and 3-DOF modules. All nodes along a given path from the root of the tree to any leaf (referred to as branch) form a unique set of posture control points.

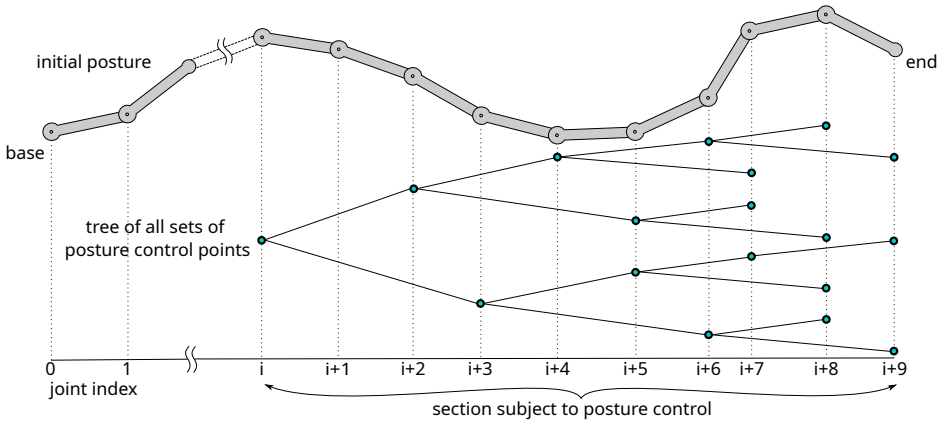


Fig. 4.7 Enumeration of all posture control sets by building a tree of 2-DOF and 3-DOF modules where nodes in the same branch produce a single set

3. An unstable edge is one for which the corresponding module is close to a singularity. Those edges always lead to an unstable posture matrix as demonstrated in Appendix A so they need to be excluded from all branches. This is done by performing a depth-first

search on the tree during which branches are inspected for unstable edges according to Algorithm 2.

Algorithm 2 Unstable edges removal algorithm

Require: $threshold > 0$

```
for all branch ∈ tree do
    create branch newbranch
    push branch's first node to newbranch
    for all edge ∈ branch do
        if edge's manipulability > threshold then
            push edge's last node to newbranch
        end if
    end for
end for
```

4. Finally, the candidate control sets are derived from the resulting branches and a score prioritizing a high manipulability index is assigned to each of them. For instance, the score of the i^{th} branch is expressed as ω_i in equation 4.8 where η_i is the number of edges in the branch and ω_j^i is the manipulability index of the j^{th} edge in the branch.

$$\omega_i = (\eta_i + 1) \prod_{j=1}^{\eta_i} \omega_j^i \tag{4.8}$$

4.6.2 Integration of the Selection and Projection Methods

In the preceding sections, two methods for tracking a posture have been developed. The Selection method is meant to be used as the primary method to control the posture while the Projection method is meant to get the control unstuck at singular postures and improve posture convergence error. One way of integrating both methods under a single posture control framework is shown in Figure 4.8. In this algorithm, the Selection method is used as long as it can get posture control points and the posture error is reduced, otherwise the control switches briefly to the Projection method. The Selection method takes over control as soon as favorable conditions for it are restored.

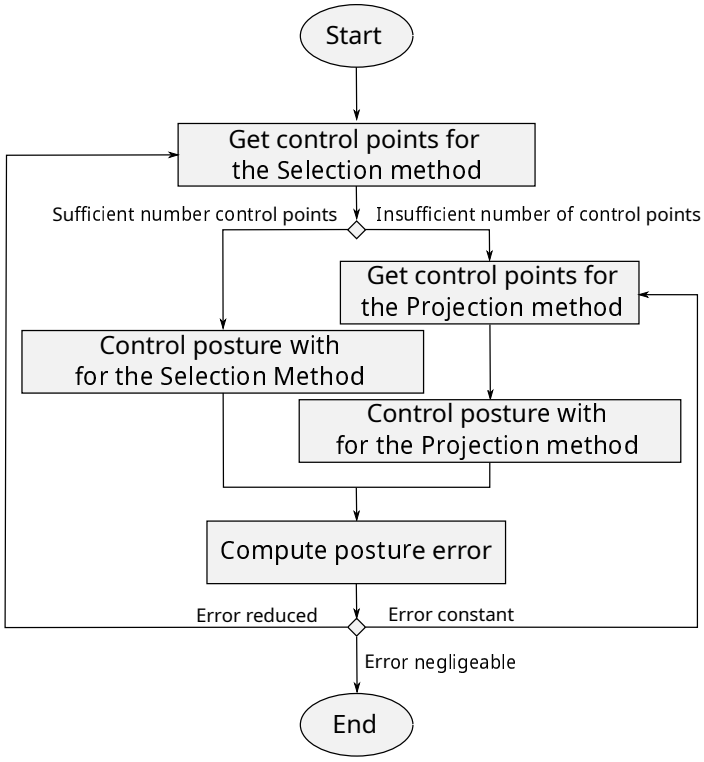


Fig. 4.8 Integration example of the Selection method with the Projection method that use the Projection method only as a failover

4.7 Discussion

This chapter presented the main control formulation of this work. Posture control points are identified along the arm and a sub-task that maps a posture control point task space velocity to the joint velocities of the arm is formed using the Jacobian matrix at the posture control point. A method to build a posture control matrix that can realize as many of these sub-tasks at once as possible without introducing algorithmic singularities is presented. This allows the posture to be safely (algorithmic singularity free manner) steered towards the target posture at each iteration step. A target singular posture can be safely approached with this method although with some potentially larger residual error. Therefore, Chapter 5 presents a complementary formulation that features a better convergence error.

Chapter 5

Manipulability Space Projection

In the previous chapter, the approach for controlling the posture of the arm relied on devising ways to construct a posture control matrix that is fully manipulable along all the dimensions of the task space. In other words, we attempted to construct a posture matrix that can map any arbitrary objective task space velocities onto some joint space velocities (Section 3.2). This is only possible if the posture matrix has full row rank. Therefore, great care was taken to select only those specific posture control points that would lead to that end. That approach is termed the Selection method.

One can expect the Selection method to run out of posture control points as the arm approaches a singular posture. This could cause higher convergence errors. Furthermore, despite being able to track singular postures, the Selection method can not initiate tracking in the vicinity of singular postures. Therefore, a complementary method is proposed. This method is based on the fundamental idea of constraining the desired task velocities to a subspace of the task space in which the arm still has motion. Section 5.1 gives an overview of this space termed the manipulability space. Thus, only feasible task space velocities are considered in its formulation explained in Section 5.2 and as a result, the posture control matrix has always full row rank. Such an approach is named the Projection method. The Projection method forces posture control point velocities toward directions of achievable motion rather than in the directions that immediately minimize posture errors. Figure 5.1 is an illustration of this.

5.1 Manipulability Space

The manipulability space is defined as the subspace of task space spanned by a subset of the principal axes of the manipulability ellipsoid which are left-singular vectors of the Jacobian matrix. The dimension of the manipulability space at any configuration of the arm is deter-

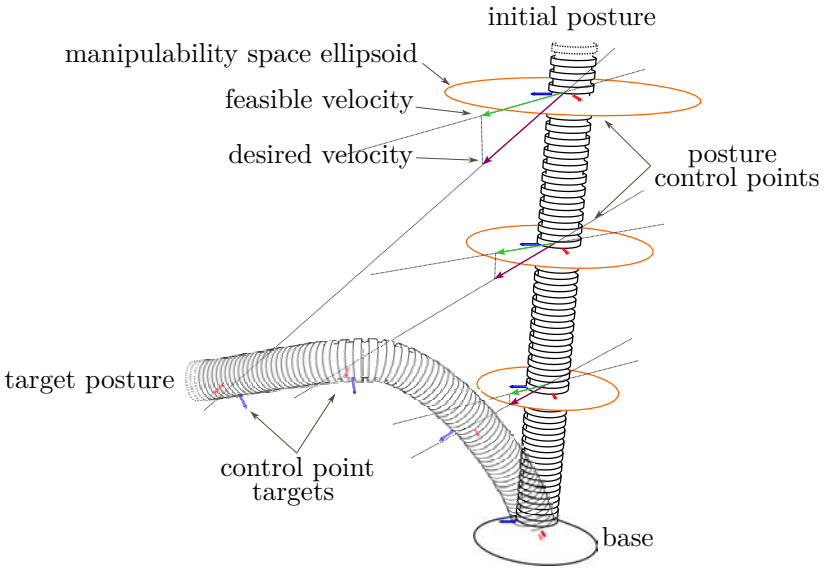


Fig. 5.1 Desired vs feasible tasks velocities at a singular posture

mined by the radius of the ellipsoid along each of these axes. Task space velocity vectors within the manipulability space are guaranteed to have corresponding achievable joint space velocities as discussed in Section 3.2.

5.2 Control Formulation

Let us consider a section of the arm ending at module k, its joints velocity vector $d\mathbf{q}_k$, Jacobian matrix J_k and end-effector position $d\mathbf{p}_k$ are relative to the frame at the base of that section. As seen in Section 3.2, the singular value decomposition of the Jacobian matrix yields the relation (5.1).

$$d\mathbf{p}_k = U_k \Sigma_k V_k^t d\mathbf{q}_k \tag{5.1}$$

$$d\mathbf{p}_k = [\mathbf{u}_1, \ldots, \mathbf{u}_m] diag_{m,n}(\sigma_1, \ldots, \sigma_{\min(m,n)}) [\mathbf{v}_1^T, \ldots, \mathbf{v}_n^T]^T d\mathbf{q}_k \tag{5.2}$$

Singular values are inspected and dimensions for which the motion of the end-effector is difficult or not feasible are excluded from the formulation. This is achieved by finding the index r of the smallest singular value that is greater than a given threshold κ and excluding the left-singular vectors of higher indices (5.4). The manipulability space is spanned by the

first r left-singular vectors $\{\mathbf{u}_1,\ldots,\mathbf{u}_r\}$.

$$d\mathbf{p}_k = \bar{U}_k\bar{\Sigma}_k V_k^t d\mathbf{q}_k \tag{5.3}$$

$$d\mathbf{p}_k = [\mathbf{u}_1,\ldots,\mathbf{u}_r]diag_{r,r}(\sigma_1,\ldots,\sigma_r)[\mathbf{v}_1^T,\ldots,\mathbf{v}_n^T]^T d\mathbf{q}_k \tag{5.4}$$

$$\text{where } r = \{i \in \{1,\ldots,\min(m,n)\} : \sigma_i \leq \kappa > \sigma_{i+1} \tag{5.5}$$

Thus, the motion of the posture control point can be explicitly confined within the r-dimensional manipulability space by projecting $d\mathbf{p}_k$ on the manipulability space as follows.

$$d\bar{\mathbf{p}}_k = \begin{bmatrix} \mathbf{u}_1^t \\ \vdots \\ \mathbf{u}_r^t \end{bmatrix} d\mathbf{p}_k = \bar{U}_k^t\bar{U}_k\bar{\Sigma}_k V_k^t d\mathbf{q}_k \tag{5.6}$$

For an arbitrary task velocity $d\bar{\mathbf{r}}_k$ relative to the inertial frame, its projection on the manipulability space is performed and the resulting velocity $d\mathbf{r}_k$ combined with equation (5.6) leads to the final expression in equation (5.8).

$$d\mathbf{r}_k = \begin{bmatrix} \mathbf{u}_1^t \\ \vdots \\ \mathbf{u}_r^t \end{bmatrix} E_{k-1}^t d\bar{\mathbf{r}}_k = \bar{U}_k^t E_{k-1}^t d\bar{\mathbf{r}}_k \tag{5.7}$$

$$\bar{U}_k^t E_{k-1}^t d\bar{\mathbf{r}}_k = \bar{U}_k^t\bar{U}_k\bar{\Sigma}_k V_k^t d\mathbf{q}_k \tag{5.8}$$

Therefore, for a given set of control points $\{s_1, s_2, \ldots, s_N\}$, the posture control matrix is formed according to equation 5.9 in a similar manner as in Section 4.5 and $d\mathbf{r}$ is obtained according to equation (5.10).

$$J = \begin{bmatrix} J_{s_1} & 0 & \cdots & 0 \\ 0 & J_{s_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & J_{s_N} \end{bmatrix} \tag{5.9}$$

$$d\mathbf{r} = \begin{bmatrix} \bar{U}_{s_1}^t E_{s_1-1}^t(\mathbf{p}_{s_1} - \mathbf{c}_{s_1}) \\ \bar{U}_{s_2}^t E_{s_2-1}^t(\mathbf{p}_{s_2} - \mathbf{c}_{s_2}) \\ \vdots \\ \bar{U}_{s_N}^t E_{s_N-1}^t(\mathbf{p}_{s_N} - \mathbf{c}_{s_N}) \end{bmatrix} \tag{5.10}$$

5.3 Discussion

This chapter introduced an alternate formulation to the control presented in Chapter 4. In Chapter 4, the task space target velocities of each posture control point were chosen in the directions that immediately minimize the position errors of individual posture control points. These velocities were not always achievable at once so the necessity of selecting a subset of posture control points with achievable velocities arose. In the other hand, this chapter sets the task space target velocities of each posture control point in the directions of achievable motion for each control point for a specific configuration of the arm. The achievable velocities are found using the concept of a manipulability space which is the subspace of R^3 in which target velocities of the posture control point are achievable. The dimension of the manipulability space grows and shrinks depending on the proximity to singularities. This results in more posture control points controlled at each iteration leading potentially to smaller convergence errors.

Chapter 6

Application and Evaluation

This chapter is dedicated to applying the main propositions of the thesis presented in Chapter 4 and Chapter 5 to an example of a concrete arm morphology and performing some posture tracking simulations. Section 6.1 and Section 6.2 treat the case of a spatial and planar arm respectively.

6.1 Application to a Spatial Arm

In this section, the proposed methods will be tested on a spatial arm. The aim is to first demonstrate how to prepare an actual example of arm morphology for use in the control methods in Section 6.1.1. This is then followed by a number of sections showcasing tracking simulations designed to test the performance of each control methods individually.

6.1.1 Example Hyper-Redundant Spatial Arm

Assuming that a module is traversed by a target spatial curve from its base then the module needs to have at least 2-DOFs in order for its end-effector to be able to track a position on the curve. Therefore, simple 2-DOF modules formed from a sequence of revolute joints around X-axis (R_x), Y-axis (R_y), or Z-axis (R_z) are considered. The module morphology consisting of a R_y joint followed by a R_z joint (R_yR_z-module) is chained to build the example hyper-redundant arm used in this section and shown in Figure 6.1.

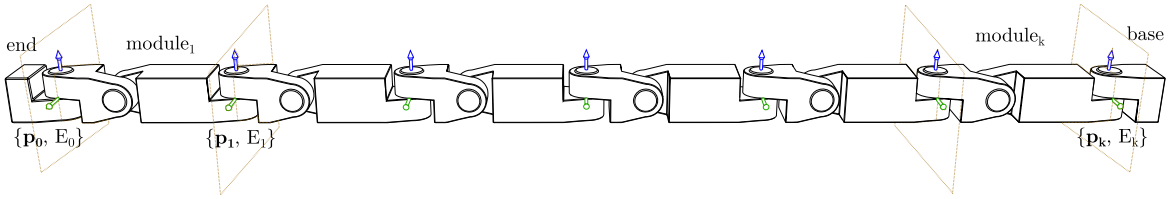


Fig. 6.1 An example hyper-redundant arm with 2-DOF R_yR_z-modules

The module morphology (R_yR_z-module) is depicted in Figure 6.2. All notations used are those defined in Section 4.1.

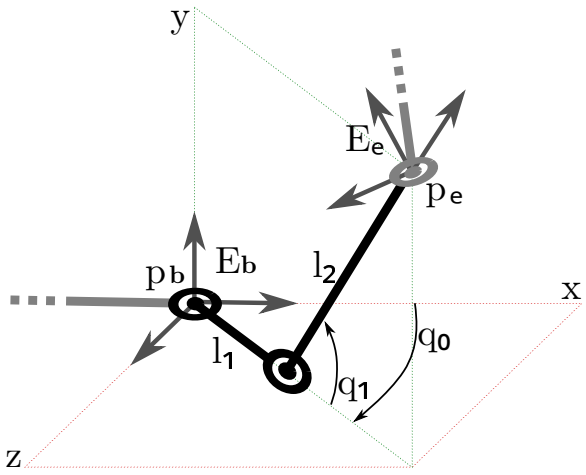


Fig. 6.2 A 2-DOF R_yR_z-module morphology parametrization

Module Kinematics Solvers

$FK(\mathbf{q})$ for a R_yR_z-module is expressed by the position-level forward kinematics equations in (6.1) while $IK(\mathbf{p})$ is describe by the inverse kinematics equation in (6.2).

$$\mathbf{p}_e = E_b R_y\{q_0\}\,(l_1 I + l_2 R_z\{q_1\})\,\mathbf{e}_x + \mathbf{p}_b \tag{6.1a}$$

$$E_e = E_b R_y\{-q_0\} R_z\{q_1\} \tag{6.1b}$$

$$[q_0\ q_1]^T = \left[\arctan(\frac{-z}{x})\ \arcsin(\frac{y}{l_2})\right]^t \tag{6.2}$$

Given an inertial frame at $\{\mathbf{p}_0, E_0\}$ as in shown Figure 6.1, the velocity level-forward kinematics of the k^{th} posture control point relative to the inertial frame can be expended as

follows and yields the equation (6.7).

$$\mathbf{p}_k = \mathbf{p}_0 + \sum_{i=0}^{k-1} E_i \mathbf{p}_{i+1}^{(i)} \tag{6.3}$$

$$\mathbf{p}_k = \mathbf{p}_0 + E_0 \sum_{i=0}^{k-1} \prod_{j=0}^{i} E_j^{(j-1)} \mathbf{p}_{i+1}^{(i)} \tag{6.4}$$

$$E_{j+1}^{(j)} = R_y\{q_{2j}\} R_z\{q_{2j+1}\} \tag{6.5}$$

$$\mathbf{p}_{i+1}^{(i)} = R_y\{q_{2i}\} \left(l_1 I + l_2 R_z\{q_{2i+1}\}\right) \mathbf{e}_x \tag{6.6}$$

$$\frac{d\mathbf{p}_k}{dq_u} = E_0 \sum_{i=u/2}^{k-1} \prod_{j=0}^{i} \tilde{E}_j^{(j-1)} \tilde{\mathbf{p}}_{i+1}^{(i)} \tag{6.7}$$

$$\tilde{E}_{j+1}^{(j)} = \tilde{R}_y\{q_{2j}\} \tilde{R}_z\{q_{2j+1}\} \tag{6.8}$$

$$\tilde{\mathbf{p}}_{i+1}^{(i)} = \tilde{R}_y\{q_{2i}\} \left(l_1 \tilde{I}_{2i+1} + l_2 \tilde{R}_z\{q_{2i+1}\}\right) \mathbf{e}_x \tag{6.9}$$

$$\text{where} \qquad \tilde{R}_y\{q_k\} = \begin{cases} I_y R_y\{q_k + \frac{\pi}{2}\} & (k = u) \\ R_y\{q_k\} & (otherwise) \end{cases} \tag{6.10}$$

$$\tilde{I}_k = \begin{cases} 0 & (k = u) \\ I & (otherwise) \end{cases} \tag{6.11}$$

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} I_y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} I_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.12}$$

Module Target Resolver

The workspace of a R_yR_z-module is constrained on the surface of a torus of center $[0\ 0\ 0]^T$, major radius l_1, and minor radius l_2. The target position resolver $TP(\mathbf{s})$ for this module must find the intersection of this surface with a given sample line segment connecting a sample point **a** to sample point **b**. This can be expressed as the solution of the following quartic

equation (6.15).

$$\mathbf{a} = [a_x\ a_y\ a_z]^t\ ,\ \mathbf{b} = [b_x\ b_y\ b_z]^t$$
$$(\mathbf{p}^t\mathbf{p} - (l_1 + l_2))^2 - 4l_1^2(l_2^2 - \mathbf{p}^t E\mathbf{p}) = 0 \tag{6.13}$$
$$\mathbf{p} = \mathbf{a} + (\mathbf{b} - \mathbf{a})t \tag{6.14}$$

$$\begin{aligned}
&\mathbf{b}^t\mathbf{b}t^4 + 4\mathbf{a}^t\mathbf{b}\mathbf{b}^t\mathbf{b}t^3 \\
&+2((\mathbf{a}^t\mathbf{a} - (l_1^2 + l_2^2))\mathbf{b}^t\mathbf{b} + 2\mathbf{a}^t\mathbf{b} + 2l_1^2\mathbf{b}^t H\mathbf{b})t^2 \\
&+2(\mathbf{a}^t\mathbf{b}(\mathbf{a}^t\mathbf{a} - (l_1^2 + l_2^2)) + 4l_1^2\mathbf{a}^t H\mathbf{b})t \\
&+(\mathbf{a}^t\mathbf{a} - (l_1^2 + l_2^2))^2 + 4l_1^2(\mathbf{a}H\mathbf{a} - l_2^2) = 0
\end{aligned} \tag{6.15}$$

$$\text{Where} \quad H = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

A valid target position occurs when among all 4 possible solutions only one is positive and is within the range $(0, 1]$.

6.1.2 Arbitrary Non-Singular Posture Tracking

An arm is formed with 20 of the simple R_yR_z-modules from the previous section. The length of each inner link of the module is 2.5 cm. The performance of the control methods is tested against three categories of tacking scenarios: 1) non-singular initial posture to arbitrary non-singular target posture (Section 6.1.2), 2) non-singular initial posture to singular target posture (Section 6.1.3), and 3) singular initial posture to arbitrary target posture (Section 6.1.4). Postures sampled along the straight line of equation (6.16) and the posture curve of equation (6.17) are used as typical examples of singular and non-singular posture curves respectively.

$$x = 110t,\ \ y = 0,\ \ z = 0 \tag{6.16}$$
$$x = 60t,\ \ y = 6\sin(8\pi t),\ \ z = 6\cos(8\pi t) \tag{6.17}$$

Figure 6.3 illustrates an example of tracking an arbitrary non-singular posture from a non-singular initial posture. An arbitrary non-singular posture is generated and used as the initial posture of the arm in blue. The target non-singular posture is along the continuous posture curve in black. Postures depicted in red and green are the final postures of the arm when controlled using the Selection method and Projection method respectively.

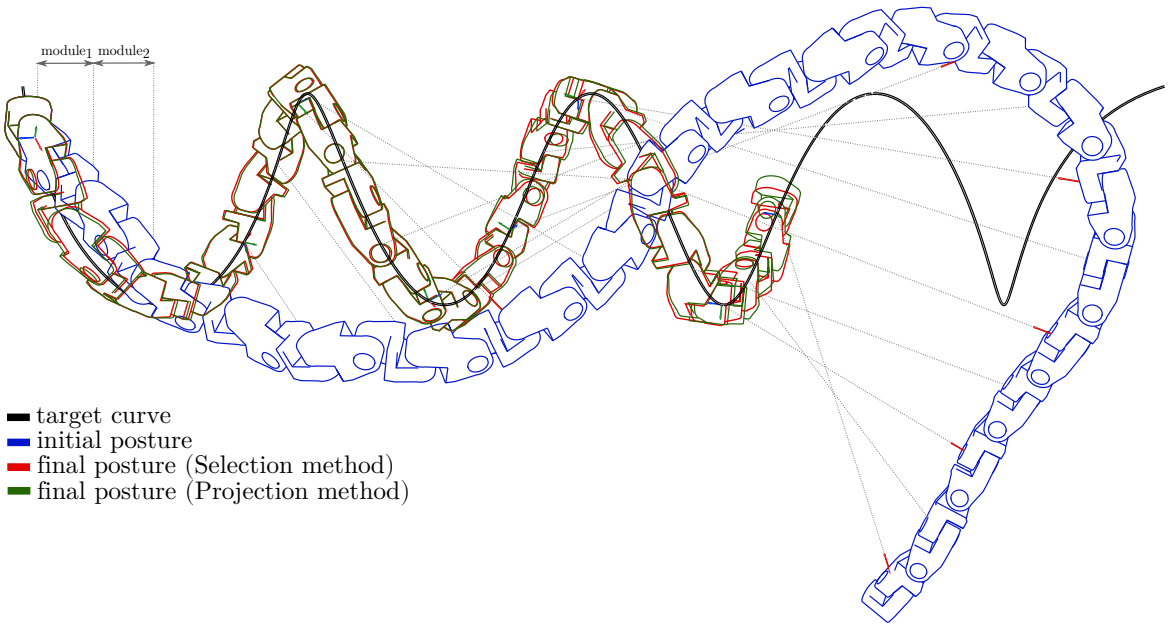
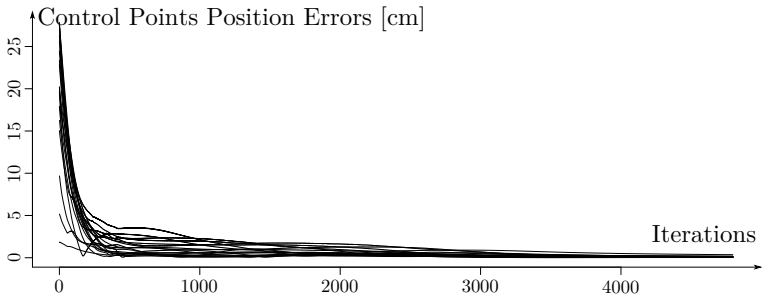


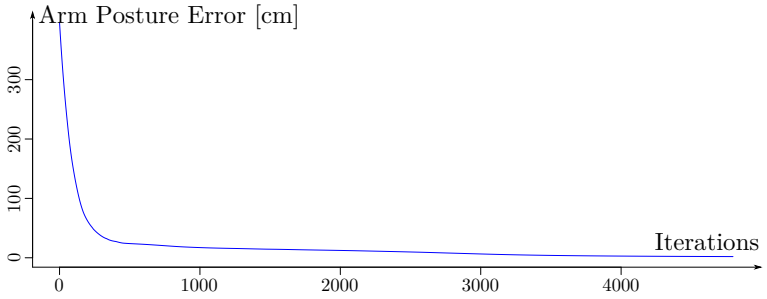
Fig. 6.3 Example of tracking an arbitrary non-singular posture from a non-singular initial posture for a spatial arm using the Selection and Projection methods separately

The results of the Selection method are presented first in Figures 6.4(a,b,c). The selection method has the property of consistently reducing the position error of the posture control points of the active control set with each iteration. Figure 6.4a shows how every posture control point position error fluctuates with each iteration. It can be seen that this error decrease is not monotonic since only the position errors of the posture control points in the active control set of a given iteration are guaranteed to decrease during that iteration and the active control set changes with each iteration as shown in Figure 6.4c. Furthermore, the position error of each posture control point is not necessarily minimized but their sum is (Figure 6.4b).

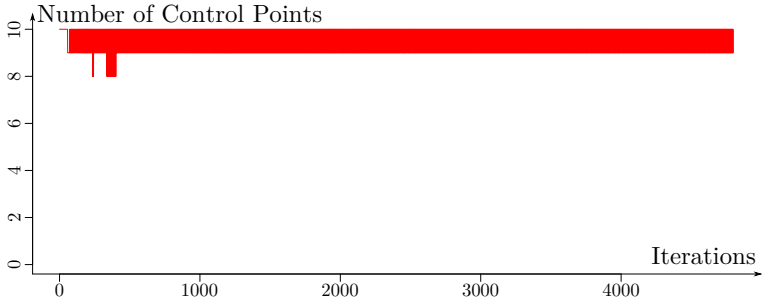
Next, the results of the Projection methods are presented in Figures 6.4(d,e). For the Projection method, the posture control point position errors change in a seemingly chaotic manner as shown in Figure 6.4d. Here, all control points are always used and participate in the overall motion of the arm. Since each posture control point is moved to follow the path of feasible motion for that control point as opposed to the specified motion (which is the only one capable of monotonically reducing the position error of the control point), the resulting errors are likely non-monotonic. As a result, the posture error shown in Figure 6.4e is not monotonically decreasing but it does eventually converge to zero with lower convergence error than the Selection method.



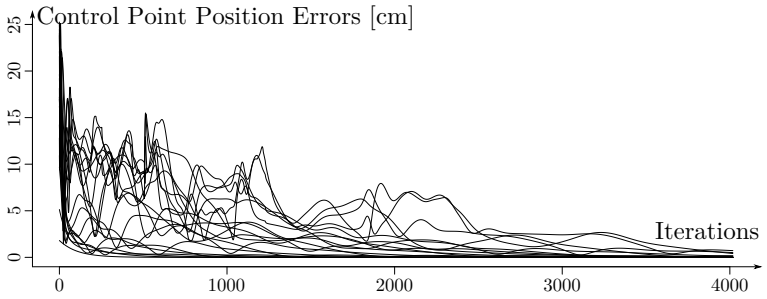
(a) Posture control point position errors in the Selection method



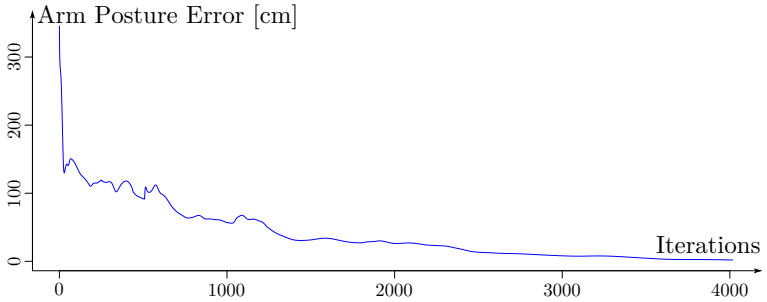
(b) Arm posture error in the Selection method



(c) Number of control points used in the Selection method



(d) Posture control point position errors in the Projection method



(e) Arm posture error in the Projection method

Fig. 6.4 Results of tracking an arbitrary non-singular posture from a non-singular initial posture for a spatial arm using the Selection and Projection methods separately

6.1.3 Singular Posture Tracking

Figure 6.5 illustrates an example of tracking a singular posture from a non-singular initial posture. The initial posture in the previous section is reused here as the initial posture of the arm in blue. The target singular posture is along the continuous line in black. This figure also contains the results of two experiments, arm postures depicted in red and green are the final postures of the arm when controlled with the Selection and Projection methods respectively.

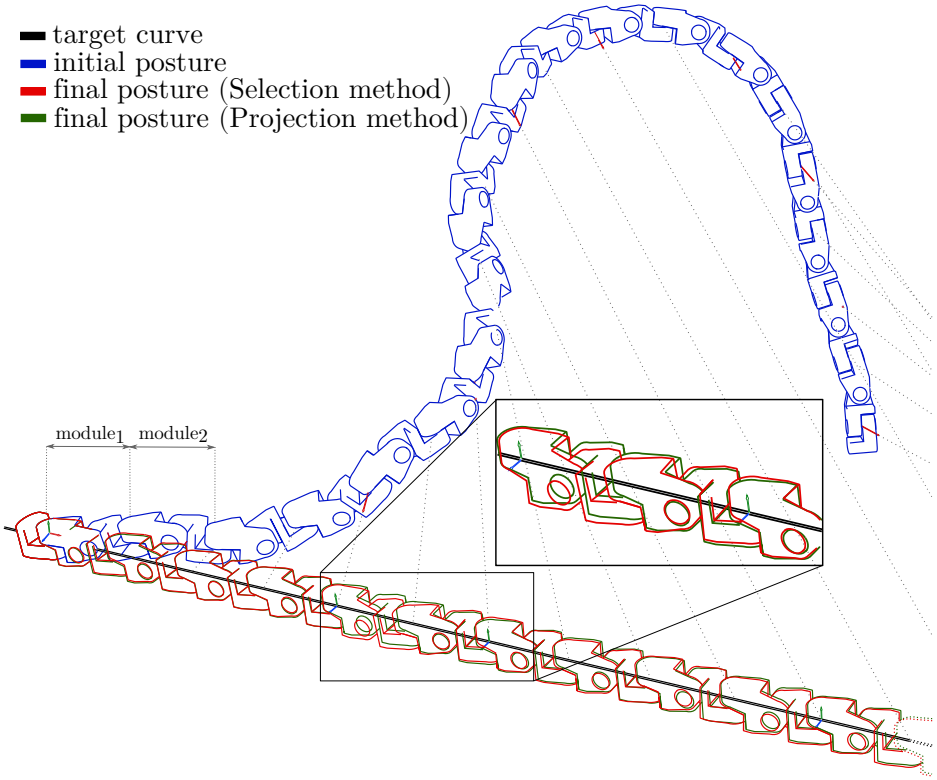
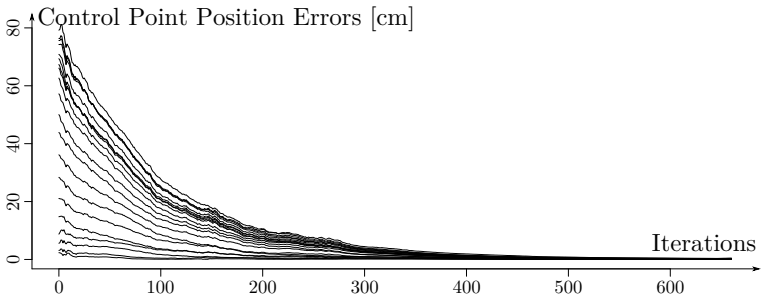
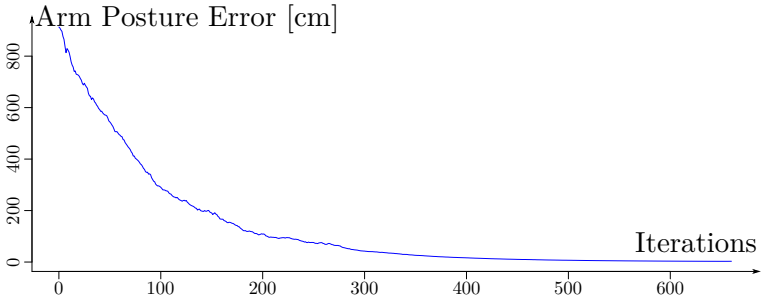


Fig. 6.5 Example of tracking a singular posture from a non-singular initial posture for a spatial arm using the Selection and Projection methods separately

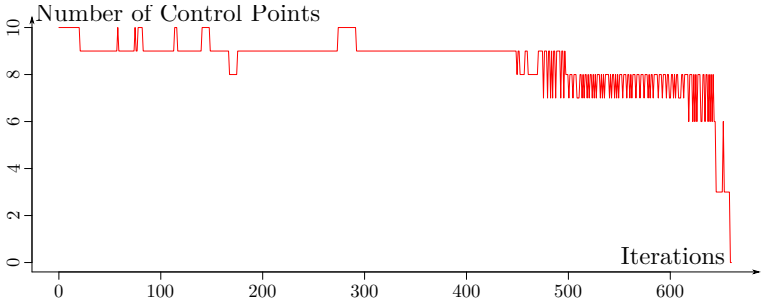
Results for the Selection method are shown in Figures 6.6(a,b,c). Since the Selection method attempts to select a set of posture control points that can be moved safely, it tends to run out of posture control points to use as the arm approaches a singular posture as show in Figure 6.6c. At some point, no posture control points qualifying for selection are found and the control is stranded. This method has the advantage of being able to safely steer the arm as close as possible to the singular target posture but its main drawback it might not achieve complete convergence as seen in Figure 6.6b and is ineffective afterwards. The Projection method on the other hand, retains its ability to guide the arm even as it approaches the singular posture (Figure 6.6e).



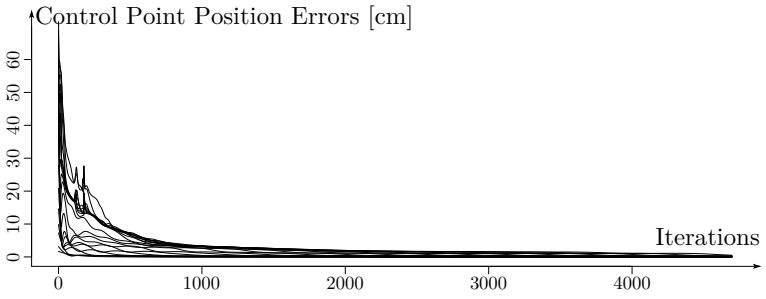
(a) Posture control point position errors in the Selection method



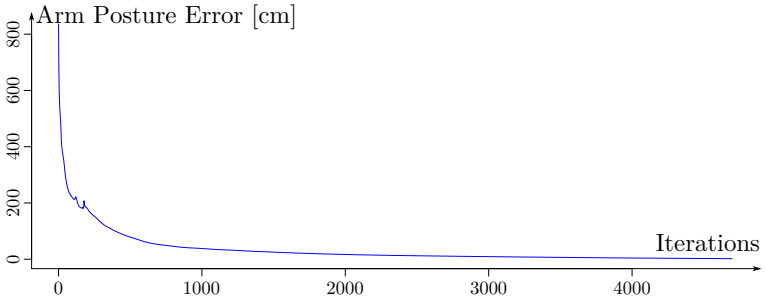
(b) Arm posture error in the Selection method



(c) Number of control points used in the Selection method



(d) Posture control point position errors in the Projection method



(e) Arm posture error in the Projection method

Fig. 6.6 Results of tracking a singular posture from a non-singular initial posture for a spatial arm using the Selection and Projection methods separately

6.1.4 Tracking from a Singular Posture

Figure 6.7 illustrates an example of tracking an arbitrary posture from a singular initial posture. The singular initial posture is the arm in blue and the target posture is along the line in black. The Projection method is the only method applicable to this scenario. Results in Figure 6.8 proves that the Projection method has the ability to break free from the singular initial posture and successfully converge toward the target posture.

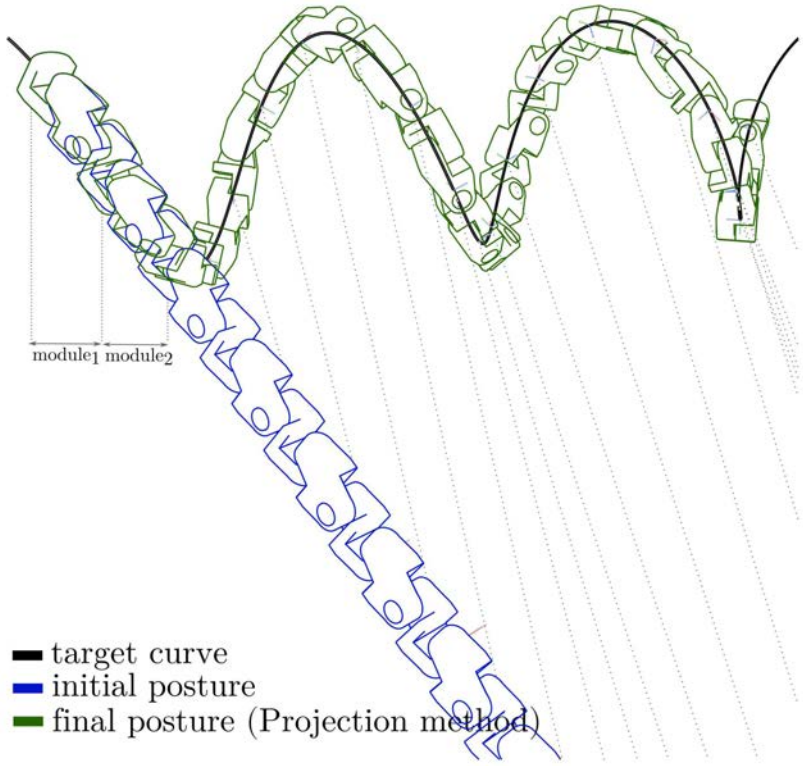
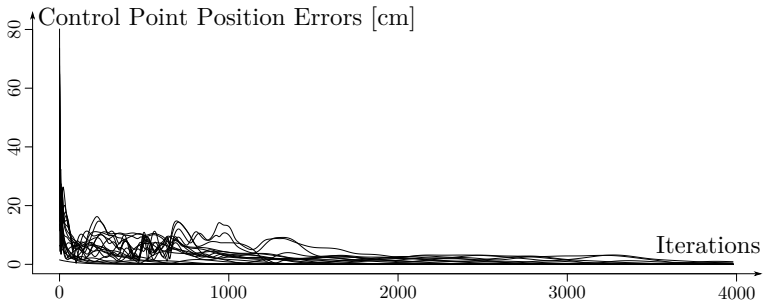
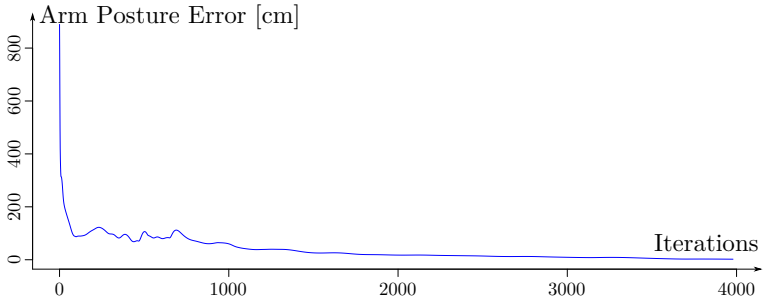


Fig. 6.7 Example of tracking an arbitrary posture from a singular initial posture for a spatial arm using the Projection method



(a) Posture control point position errors in the Projection method



(b) Arm posture error in the Projection method

Fig. 6.8 Results of tracking an arbitrary posture from a singular initial posture for a spatial arm using the Projection method

6.2 Application to a Planar Arm

In this section, the proposed methods will be tested on a planar arm. The first three sections present tracking simulations designed to test the performance of each control methods individually. The last section shows a tracking example where the two control methods are used alternatively. A 45-DOF hyper-redundant planar arm with revolute joints and a link length of 15cm is used as example.

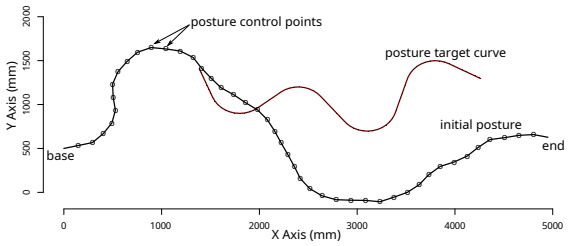
6.2.1 Arbitrary Non-Singular Posture Tracking

First, tracking an arbitrary non-singular target posture from a non-singular initial posture is attempted in Figure 6.9. Figure 6.9a depicts the initial setup with the 45-DOF arm along with its posture target curve. Figures 6.9(b,c) show the simulation results when using the Selection method and Figures 6.9(d,e) when using the Projection method. It can be observed from the results in Figures 6.9(b,d) that both methods register a lot of posture control points at each iteration step. This results in a stable motion and a convergence with little residual error. Figure 6.9c confirms that the posture error is reduced monotonically for the Selection method and convergence is achieved within a few iterations. For the projection method on

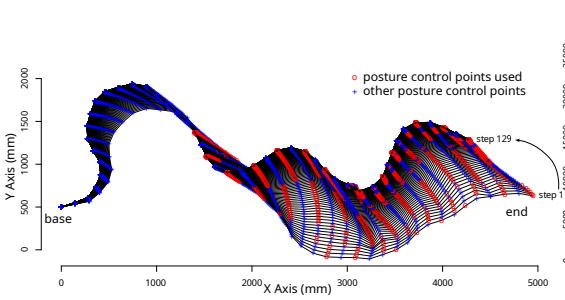
the other hand, the posture error fluctuates seemingly arbitrarily but tends to reduce as the number of iterations increases (Figure 6.9e). As a result, arm postures from the projection method appear to be chaotic and arbitrary (Figure 6.9d). Figures 6.9(f,g) show the final posture of the arm.

6.2.2 Singular Posture Tracking

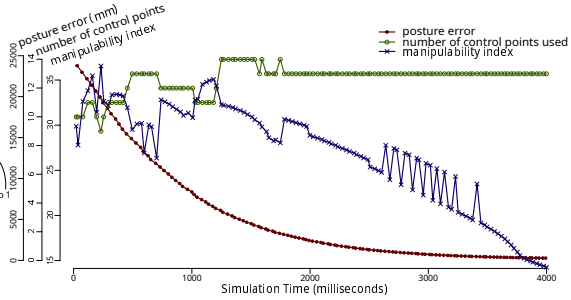
Similarly, the next set of results in Figure 6.10 presents the case when the arm starts at an arbitrary non-singular initial posture but has a near-singular target posture. As the posture of the arm approaches the target posture, the number of posture control points decreases rapidly in the Selection method (Figure 6.10c). As a result, the posture is less or not controllable at all once it reaches the target posture. Moreover, residual posture errors tend to be larger as this method loses posture control points near the target posture. The Projection method in contrast always registers a lot of posture control points (Figure 6.10e) which means that the posture is still controllable even after it converges to the near-singular target posture.



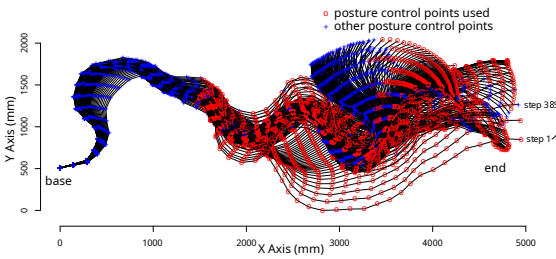
(a) Initial posture and posture target curve



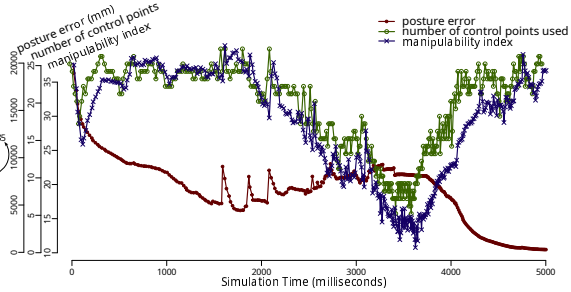
(b) Posture control points used at each iteration step in the Selection method



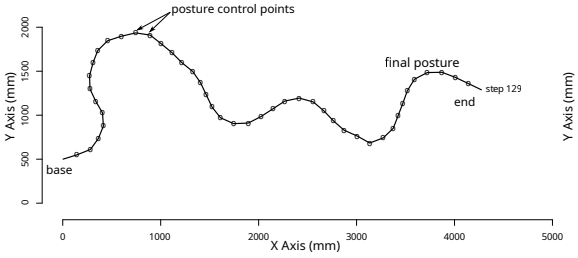
(c) Posture error, number of posture control points used and manipulability index in the Selection method



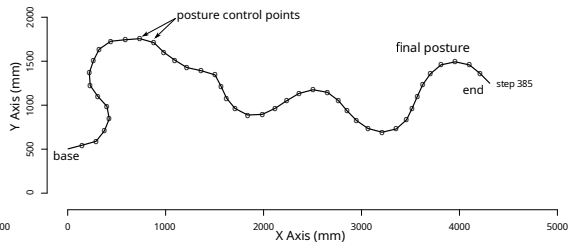
(d) Posture control points used at each iteration step in the Projection method



(e) Posture error, number of posture control points used and manipulability index in the Projection method

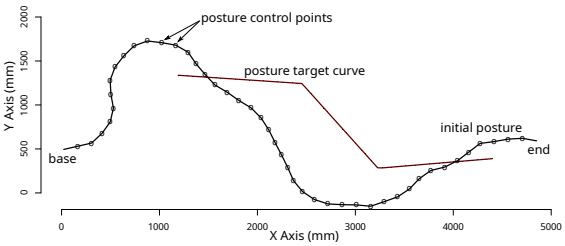


(f) Final posture in the Selection method

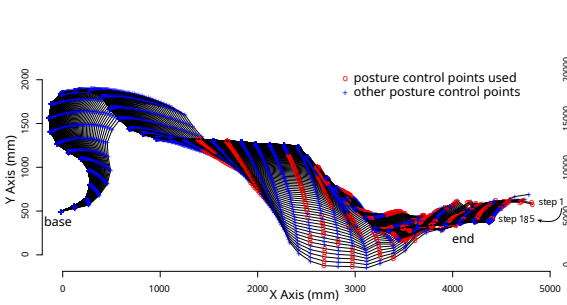


(g) Final posture in the Projection method

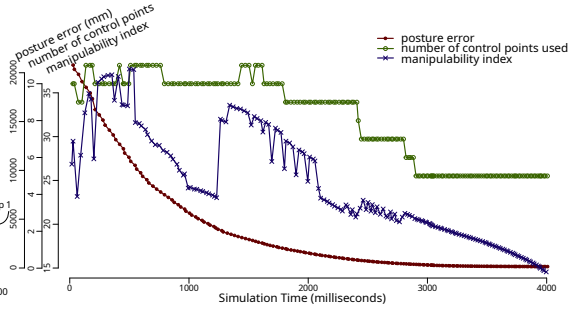
Fig. 6.9 Example of tracking an arbitrary non-singular posture from a non-singular initial posture for a planar arm using the Selection and Projection methods separately



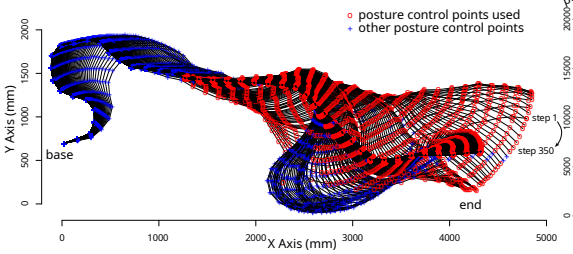
(a) Initial posture and posture target curve



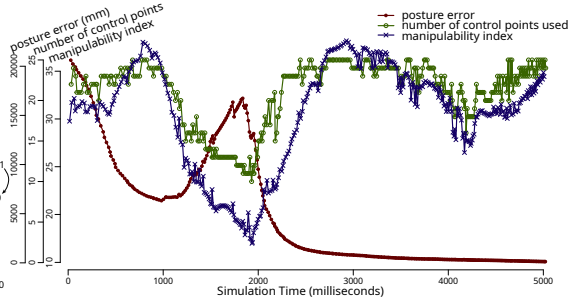
(b) Posture control points used at each iteration step in the Selection method



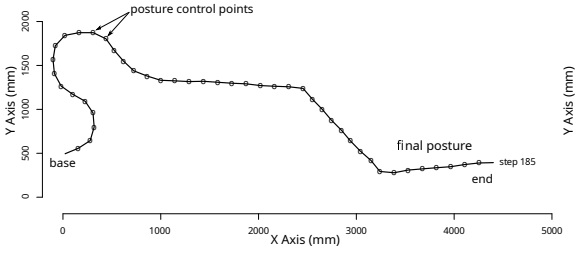
(c) Posture error, number of posture control points used and manipulability index in the Selection method



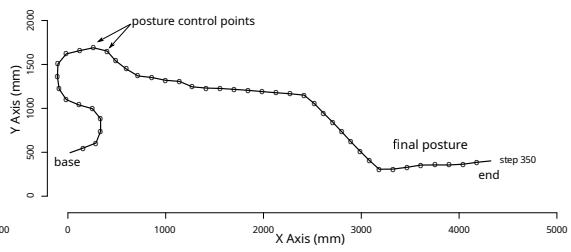
(d) Posture control points used at each iteration step in the Projection method



(e) Posture error, number of posture control points used and manipulability index in the Projection method



(f) Final posture in the Selection method

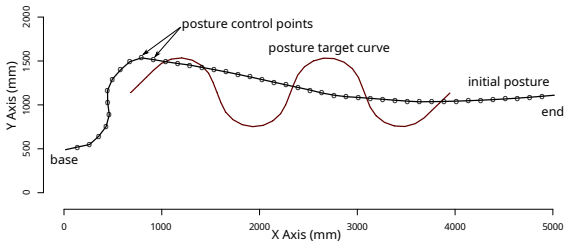


(g) Final posture in the Projection method

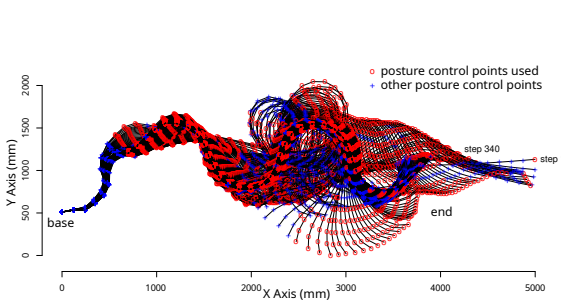
Fig. 6.10 Example of tracking a near-singular posture from a non-singular initial posture for a planar arm using the Selection and Projection methods separately

6.2.3 Tracking from a Singular Posture

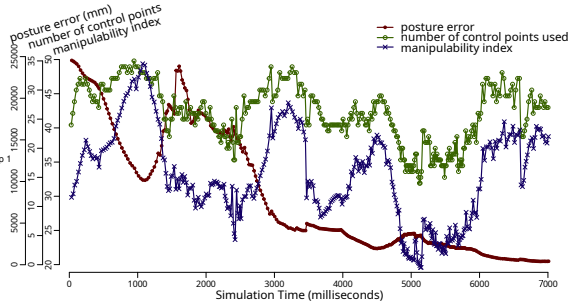
The next set of results in Figure 6.11 is for the case when the arm starts at a singular initial posture. In this case, the Selection method is ineffective as it does not register any posture control points so the results for the Projection method alone are shown. The results confirm that arm can effectively depart from a singular initial posture if the Projection method is used.



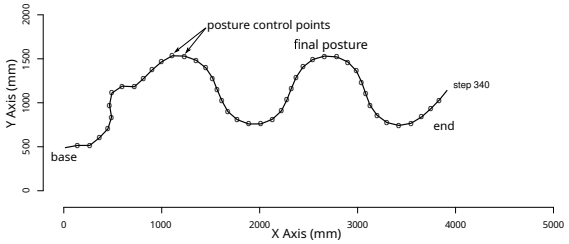
(a) Initial posture and posture target curve



(b) Posture control points used at each iteration step in the Projection method



(c) Posture error, number of posture control points used and manipulability index in the Projection method

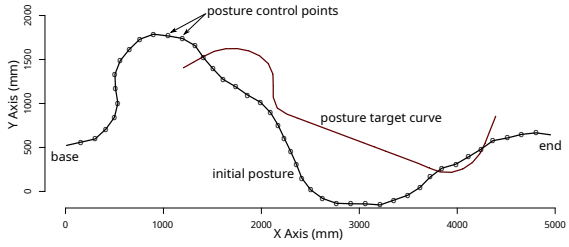


(d) Final posture in the Projection method

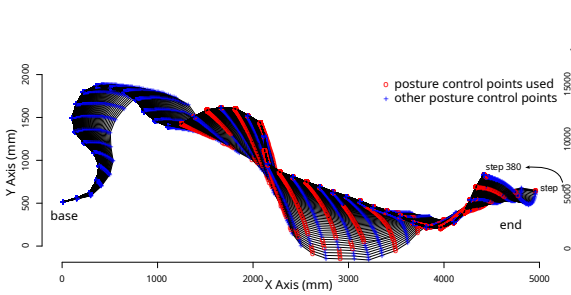
Fig. 6.11 Example of tracking an arbitrary posture from a singular initial posture for a planar arm using the Projection method

6.2.4 Tracking from a Near-Singular Posture by Switching Methods

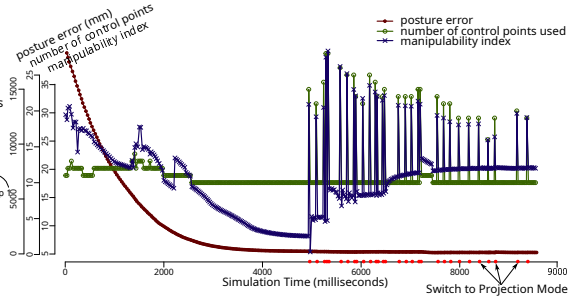
The final set of results in Figure 6.12 shows the case when both the Selection and Projection methods are integrated as in the algorithm of Figure 4.8. The Selection method is used whenever possible and the control method is switched to the Projection method if the posture error is not being substantially reduced after a few iterations of the Selection method. The aim of combining the two methods is to further reduce convergence error. This is an important topic that will also be explored for the spatial case in future works. Note that the iteration steps where the Projection method was used in place of the Selection method are marked with red dots on the X-axis of Figure 6.12c. It can be seen that a smaller residual error than in the case where only one of the control methods is used.



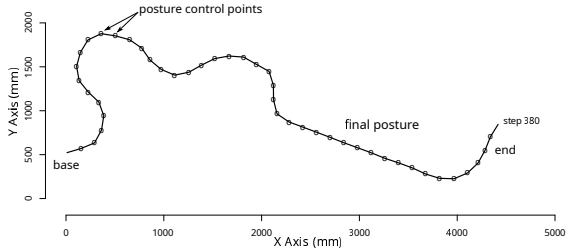
(a) Initial posture and posture target curve



(b) Posture control points used at each iteration step



(c) Posture error, number of posture control points used and manipulability index



(d) Final posture

Fig. 6.12 Example of tracking a near-singular posture from a non-singular initial posture for a planar arm by switching between the Selection and Projection methods

6.3 Discussion

This chapter applied the control formulations of Chapter 4 and Chapter 5 to some example planar and spatial hyper-redundant arms. A few tracking scenarios were considered and the characteristics and performance of the proposed formulations were reviewed. Similar findings were observed for both the planar and spatial case which are (1) both formulations can track non-singular target postures with similar performance, (2) both methods can also track singular target postures but the manipulability space projection method of Chapter 5 tends to achieve better convergence than the posture control point selection method of Chapter 4, (3) the manipulability space projection method can start from a singular initial posture or traverse one but the posture control point selection method can not, (4) the manipulability space projection method produces unpredictable intermediate postures, (5) a combination of both methods is a viable solution for keeping the predicable nature of the posture control point selection method while having the low convergence error of the the manipulability space projection method.

Chapter 7

Conclusion

In this thesis, a new posture control framework for hyper-redundant modular serial arms was developed and tested. It is based on the fundamental premise that the posture of hyper-redundant arms could be shaped by controlling the velocity of some strategic points located along the arm termed posture control points. The work in this thesis was motivated by the desire to provide an easier task specification for hyper-redundant tasks by designing a control framework that could track user-specified posture target curves. Thus, a suitable framework must be able to handle arbitrary target postures so a particular emphasis was placed on a framework that can track posture curves and that works well around singularities. In pursuing that goal, two control schemes were developed, termed the Selection method and the Projection method respectively. Both are techniques for constructing a guaranteed full rank posture control matrix that maps joint velocities to posture control points velocities. These methods differ mainly in the way they select and move posture control points. Several numerical simulations on a planar and spatial hyper-redundant arm were performed using these methods in order to understand their individual properties and gauge their performances. A strategy that combined the two methods under a single control framework were also proposed and tested on the planar hyper-redundant arm.

The Selection method selects movable posture control points at each iteration step and moves them in the directions that decrease their position errors. The Projection method selects a large number of posture control points at each iteration step but moves them only in directions of feasible velocities. Simulation results demonstrated that both methods can track arbitrary target postures even singular ones. The Selection method is preferable to the Projection method because it generates somewhat predictable postures but its major drawbacks are: (1) it can not depart from or near a singular posture, (2) it can not traverse or pass near a singular posture, and (3) it has the tendency of leaving larger residual errors when tracking singular postures. The Projection method on the other hand, does not have these

limitations but has the undesirable characteristic of generating unpredictable intermediate postures.

7.1 Future Works

The work done in this thesis provided a glimpse of the potential for effectively using posture control points to safely control the posture of hyper-redundant arms. The ability to fashion the posture of hyper-redundant arms into any desired shape opens a world of new interesting applications. The work presented in this thesis is a small step towards that goal but further work is however necessary.

The first aspect that deserves further investigations is finding a strategy that best seamlessly integrates the two control methods under a single more efficient posture control framework. A simple strategy was given for the planar case in Section 4.6.2 but a more detailed treatment is necessary. A rigorous study on the applicability of the control framework to arbitrary target postures is also needed in order to better understand the limitations of the proposed methods. The analysis of the computational cost of the control methods is also necessary in order to assess their viability in real-time control situations. Finally, tracking a sequence of target postures and applications to real-world hyper-redundant manipulation tasks are another candidates for future works.

References

[1] Asl, F. M., Ashrafiuon, H., and Nataraj, C. (2002). A general solution for the position, velocity, and acceleration of hyperredundant planar manipulators. *Journal of Robotic Systems*, 19(1):1–12.

[2] Aupetit, B. (1991). *A Primer on Spectral Theory*. Springer-Verlag, New York.

[3] Baillieul, J. (1985). Kinematic programming alternatives for redundant manipulators. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 722–728.

[4] Baillieul, J., Hollerbach, J., and Brockett, R. (1984). Programming and control of kinematically redundant manipulators. In *The 23rd IEEE Conference on Decision and Control*, pages 768–774.

[5] Chen, Y.-C. and Walker, I. (1993). A consistent null-space based approach to inverse kinematics of redundant robots. In *[1993] Proceedings IEEE International Conference on Robotics and Automation*, pages 374–381 vol.3.

[6] Chirikjian, G. and Burdick, J. (1990). An obstacle avoidance algorithm for hyper-redundant manipulators. In *Proceedings., IEEE International Conference on Robotics and Automation*, pages 625–631 vol.1.

[7] Chirikjian, G. and Burdick, J. (1991). Kinematics of hyper-redundant robot locomotion with applications to grasping. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pages 720–725 vol.1.

[8] Chirikjian, G. and Burdick, J. (1994). A modal approach to hyper-redundant manipulator kinematics. *IEEE Transactions on Robotics and Automation*, 10(3):343–354.

[9] Chirikjian, G. and Burdick, J. (1995). The kinematics of hyper-redundant robot locomotion. *IEEE Transactions on Robotics and Automation*, 11(6):781–793.

[10] Date, H. and Takita, Y. (2005). Control of 3d snake-like locomotive mechanism based on continuum modeling. In *Proceedings ASME 2005 International Design Engineering Technical Conferences*, pages 1351–1359.

[11] Egeland, O. (1987). Task-space tracking with redundant manipulators. *IEEE Journal on Robotics and Automation*, 3(5):471–475.

[12] Fahimi, F., Ashrafiuon, H., and Nataraj, C. (2003). Obstacle avoidance for spatial hyper-redundant manipulators using harmonic potential functions and the mode shape technique. *Journal of Robotic Systems*, 20:23 – 33.

[13] Hirose, S. and Ma, S. (1989). Redundancy decomposition control for multi-joint manipulator. In *Proceedings, 1989 International Conference on Robotics and Automation*, pages 119–124 vol.1.

[14] Hu, Z., Xiao, L., Li, K., Li, K., and Li, J. (2021). Performance analysis of nonlinear activated zeroing neural networks for time-varying matrix pseudoinversion with application. *Applied Soft Computing*, 98:106735.

[15] Jamali, A., Khan, R., and Rahman, M. M. (2011). A new geometrical approach to solve inverse kinematics of hyper redundant robots with variable link length. In *2011 4th International Conference on Mechatronics (ICOM)*, pages 1–5.

[16] Jin, L., Li, S., Hu, B., and Liu, M. (2019). A survey on projection neural networks and their applications. *Applied Soft Computing*, 76:533–544.

[17] Jin, L., Li, S., Wang, H., and Zhang, Z. (2018). Nonconvex projection activated zeroing neurodynamic models for time-varying matrix pseudoinversion with accelerated finite-time convergence. *Applied Soft Computing*, 62:840–850.

[18] Khatib, O. (1985). Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 500–505.

[19] Kim, J.-O. and Khosla, P. (1992). Real-time obstacle avoidance using harmonic potential functions. *IEEE Transactions on Robotics and Automation*, 8(3):338–349.

[20] Klein, C. A. and Blaho, B. E. (1987). Dexterity measures for the design and control of kinematically redundant manipulators. *The International Journal of Robotics Research*, 6(2):72–83.

[21] Klein, C. A. and Huang, C.-H. (1983). Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(2):245–250.

[22] Liljeback, P., Pettersen, K. Y., Stavdahl, O., and Gravdahl, J. T. (2012). A control framework for snake robot locomotion based on shape control points interconnected by bézier curves. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3111–3118.

[23] Liljeback, P., Pettersen, K. Y., Stavdahl, O., and Gravdahl, J. T. (2014). A 3d motion planning framework for snake robots. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1100–1107.

[24] Liégeois, A. (1977). Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(12):868–871.

[25] Ma, S. and Konno, M. (1997). An obstacle avoidance scheme for hyper-redundant manipulators-global motion planning in posture space. In *Proceedings of International Conference on Robotics and Automation*, volume 1, pages 161–166 vol.1.

[26] Maciejewski, A. and Klein, C. (1985). Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *The International Journal of Robotics Research*, 4(3):109–117.

[27] Martin, D., Baillieul, J., and Hollerbach, J. (1989). Resolution of kinematic redundancy using optimization techniques. *IEEE Transactions on Robotics and Automation*, 5(4):529–533.

[28] Mori, M. and Hirose, S. (2002). Three-dimensional serpentine motion and lateral rolling by active cord mechanism acm-r3. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 829–834 vol.1.

[29] Mu, Z., Yuan, H., Xu, W., Liu, T., and Liang, B. (2020). A segmented geometry method for kinematics and configuration planning of spatial hyper-redundant manipulators. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(5):1746–1756.

[30] Nakamura, Y. and Hanafusa, H. (1986). Inverse Kinematic Solutions With Singularity Robustness for Robot Manipulator Control. *Journal of Dynamic Systems, Measurement, and Control*, 108(3):163–171.

[31] Podhorodeski, R., Goldenberg, A., and Fenton, R. (1991). Resolving redundant manipulator joint rates and identifying special arm configurations using jacobian null-space bases. *IEEE Transactions on Robotics and Automation*, 7(5):607–618.

[32] Sciavicco, L. and Siciliano, B. (1987). A dynamic solution to the inverse kinematic problem for redundant manipulators. In *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, volume 4, pages 1081–1087.

[33] Sciavicco, L. and Siciliano, B. (1988). A solution algorithm to the inverse kinematic problem for redundant manipulators. *IEEE Journal on Robotics and Automation*, 4(4):403–410.

[34] Seraji, H. (1989a). Configuration control of redundant manipulators: theory and implementation. *IEEE Transactions on Robotics and Automation*, 5(4):472–490.

[35] Seraji, H. (1989b). Configuration control of redundant manipulators: theory and implementation. *IEEE Transactions on Robotics and Automation*, 5(4):472–490.

[36] Takeichi, M., Suzumori, K., Endo, G., and Nabae, H. (2017). Development of a 20-m-long giacometti arm with balloon body based on kinematic model with air resistance. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2710–2716.

[37] Volpe, R. and Khosla, P. (1990). Manipulator control with superquadric artificial potential functions: theory and experiments. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(6):1423–1436.

[38] Wright, C., Buchan, A., Brown, B., Geist, J., Schwerin, M., Rollinson, D., Tesch, M., and Choset, H. (2012). Design and architecture of the unified modular snake robot. In *2012 IEEE International Conference on Robotics and Automation*, pages 4347–4354.

[39] Yamada, H. and Hirose, S. (2006). Study on the 3d shape of active cord mechanism. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2890–2895.

[40] Yoshihiko, N., Hideo, H., and Tsuneo, Y. (1987). Task-priority based redundancy control of robot manipulators. *The International Journal of Robotics Research*, 6(2):3–15.

[41] Yoshikawa, T. (1985). Manipulability and redundancy control of robotic mechanisms. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 1004–1009.

[42] Yu, S., Ma, S., Li, B., and Wang, Y. (2011). An amphibious snake-like robot with terrestrial and aquatic gaits. In *2011 IEEE International Conference on Robotics and Automation*, pages 2960–2961.

[43] Zhang, Y. and Zhang, Z., editors (2013). *Repetitive Motion Planning and Control of Redundant Robot Manipulators*. Spinger-Verlag, Berlin (Germany).

Related Publications

[1] Dione, A.C.A and Hasegawa, S. and Mitake, H. (2017). Stable posture control for planar hyper-redundant arms using selective control points. *Advanced Robotics*, 31(23-24):1338–1348. doi: 10.1080/01691864.2017.1401958. Section 4.6, 6.2, Appendix A

[2] Dione, A.C.A and Hasegawa, S (2023). Stable posture tracking for modular spatial hyper-redundant serial arms using selective control points. *Advanced Robotics*, 0(0-0):1–15. doi: 10.1080/01691864.2023.2291715. Chapter 4, 5 and Section 6.1

Appendix A

Relative Manipulability Condition Proof

1-DOF Module Arm

Let us examine the rank of the matrix constructed by stacking J_k and J_{k+1} which are the Jacobian matrix at the posture control point k and joint $k+1$ respectively. J_k is padded with an extra column filled of zeros and we assume that J_k and J_{k+1} are both full rank $(2\times(k+1))$ matrices. Therefore, the row vectors $\{\mathbf{j}_{k1},\mathbf{j}_{k2}\}$ of J_k and $\{\mathbf{j}_{(k+1)1},\mathbf{j}_{(k+1)2}\}$ of J_{k+1} are linearly independent and the relationship in equation (A.1) holds.

$$\begin{cases}\mathbf{j}_{(k+1)1} = \mathbf{j}_{k1} - {}_kS_k\nu \\ \mathbf{j}_{(k+2)2} = \mathbf{j}_{k2} + {}_kC_k\nu\end{cases} \tag{A.1}$$

ν is a $((k+1)\times 1)$ vector with 1 in each element and $\{{}_kS_k,\ {}_kC_k\}$ are shorthand defined as follows.

$${}_vC_u = \sum_{i=u}^{v} l_i \cos\sum_{j=0}^{i} q_j, \quad {}_vS_u = \sum_{i=u}^{v} l_i \sin\sum_{j=0}^{i} q_j \tag{A.2}$$

Therefore, it can be shown with equation (A.3) that there always exists a set of values $[a\ \ b\ \ c\ \ d]^T \in \mathbb{R}^4 - \{\mathbf{0}\}$ for which $a\mathbf{j}_{k1} + b\mathbf{j}_{k2} + c\mathbf{j}_{(k+1)1} + d\mathbf{j}_{(k+1)2} = \mathbf{0}$. That is, for an arm subdivided into a 1-DOF modules, a posture control matrix including any 2 adjacent posture control points is always singular.

$$(a+c)\mathbf{j}_{k1} + (b+d)\mathbf{j}_{k2} + (d_kC_k - c_kS_k)\nu = \mathbf{0} \tag{A.3}$$

2-DOF Module Arm

Similarly, we now use J_k and J_{k+2} to construct a $4\times(k+2)$ matrix. The row vectors between the two matrices have the following relationship.

$$\begin{cases} \mathbf{j}_{(k+2)1} = \mathbf{j}_{k1} - \begin{bmatrix} {}_{k+1}S_k & \cdots & {}_{k+1}S_k & {}_{k+1}S_{k+1} \end{bmatrix} \\ \mathbf{j}_{(k+2)2} = \mathbf{j}_{k2} + \begin{bmatrix} {}_{k+1}C_k & \cdots & {}_{k+1}C_k & {}_{k+1}C_{k+1} \end{bmatrix} \end{cases} \tag{A.4}$$

If the condition in equation (A.5) is true, then an analysis identical to Appendix A can be made which leads to the same conclusion that $\{\mathbf{j}_{k1}, \mathbf{j}_{k2}, \mathbf{j}_{(k+2)1}, \mathbf{j}_{(k+2)2}\}$ are linearly dependent.

$$\exists \quad \alpha \in \mathbb{R} \quad \setminus \quad \begin{cases} {}_{k+1}S_k = \alpha \times {}_{k+1}S_{k+1} \\ {}_{k+1}C_k = \alpha \times {}_{k+1}C_{k+1} \end{cases} \tag{A.5}$$

Note that condition (A.5) is true whenever the 2-DOF module between between k and $k+2$ is in a singular posture. This corresponds to $\alpha = 2$ and $\alpha = 0$ for the two possible singular postures (Table A.1). This means that for an arm composed of 2-DOF modules, a posture control matrix including 2 adjacent posture control points is always singular if the 2-DOF module between the adjacent posture control point is singular.

Table A.1 Values of α for which the stacked matrix is rank deficient

joint k	joint $k+1$	α
$*$	0	2
$*$	$\pm\pi$	0

$*$ arbitrary value

3-DOF Module Arm

Similarly, a matrix constructed from J_k and J_{k+3} is rank deficient when the condition in equation (A.6) is true. This also coincides with the singular postures of the 3-DOF module between k and $k+3$ (Table A.2). This means that for an arm composed of 3-DOF modules, a posture control matrix including 2 adjacent posture control points is always singular if the 3-DOF module between the adjacent posture control point is singular.

$$\exists \quad (\alpha \quad \beta) \in \mathbb{R}^2 \quad \setminus \quad \begin{cases} {}_{k+2}S_k = \alpha \times {}_{k+2}S_{k+2} \\ {}_{k+2}C_k = \alpha \times {}_{k+2}C_{k+2} \end{cases} \quad \bigcap \begin{cases} {}_{k+2}S_{k+1} = \beta \times {}_{k+2}S_{k+2} \\ {}_{k+2}C_{k+1} = \beta \times {}_{k+2}C_{k+2} \end{cases} \tag{A.6}$$

Table A.2 Values of α and β for which the stacked matrix is rank deficient

joint k	joint $k+1$	joint $k+2$	α	β
$*$	0	0	3	2
$*$	0	$\pm\pi$	-1	0
$*$	$\pm\pi$	$\pm\pi$	1	0
$*$	$\pm\pi$	0	1	2

$*$ arbitrary value

n-DOF module Arm

The above analysis can be generalized to a n-DOF module. Thus, if the section of a hyper-redundant arm in between 2 adjacent posture control points used in a posture control matrix is in a singular posture then the posture matrix is singular. This is equivalent to the relative manipulability condition stated in Section 4.3.