Modeling, control and optimization of tree-type robotic systems using exponential coordinates

March 2021

Graduate School of Systems Engineering

Wakayama University

AMAR JULIEN SAMUEL

指数座標を用いたツリータイプロボットシステムのモデル化と制御および最適化

令和３年３月

和歌山大学大学院システム工学研究科

アマル　ジュリアン　サムエル

Abstract

In recent years, an increasing number of tasks are being performed by robots, and we can observe an increase in the use of automated machines at working stations in the industry to complete various tasks. Although robots can complete tasks at very high speeds that cannot be achieved manually with a high level of precision, their main weakness resides in their lack of versatility. Even if humans cannot perform tasks at the same level, the diversity of tasks that they can perform and their historic use of tools allows for a wide array of possible actions, which is a domain where robots fail to show significant progress. This gap is particularly apparent in the domain of object manipulation, where robots are limited to simple pick-and-place tasks or linear wielding. Although some progress has been made in recent years, for example, the use of robotic arms in factories, we can currently only mimic human behaviors for repetitive tasks. These tasks are becoming increasingly difficult, more varied, and we continue to seek better, more precise, and faster results from robots.

These constraints suggest that we need more flexible tools and a more general approach to task-based optimization schemes for developing such robotic structures. In order to overcome current robots weakness, the field of robotic modeling, design and optimization is the main vector of improvement. Some researches has been done, mainly involving Denavit-Hartemberg (DH) parameters and calculus-based optimization schemes, to search for more compact, more dexterous and in general more versatile robots to tackle the present challenges. However, those schemes possess a major flaw: the inherent nature of this strategy narrow the design possibilities. Indeed, DH parameters are not flexible for optimization schemes due to their inter-dependant nature, and while local optimization schemes yields some results, especially for bio-inspired robot designs, they are limited in their results since they are based on existing samples.

To answer those needs, we propose a rapid and convenient method to model and control tree-type architecture systems using exponential coordinates. Exponential coordinates have elegant rules on products and derivatives, which allows for a simple definition of joint movements; and are highly effective for modeling complex architectures. The grasping and manipulation of an object by robot manipulators is considered in order to illustrate our modeling and control process. Using the chain matrix representing the system architecture, we derive a unified framework for the manipulator and object dynamics in a closed form fashion. The key benefit of this methodology is its simplicity and flexibility. Using this newly derived form of dynamics, we can conveniently change the system configurations (add/delete joints and links, change direction of joint movements, etc.) from one design to another; this will more accurately satisfy the manipulation requirements and simplify the optimization process for the system and control designs. Simulators can be conveniently constructed by following the formulas derived in the paper. Numerical examples of an arm–hand system are conducted to illustrate the usage of the proposed formulas in modeling and control process.

Additionally, we also propose a simultaneous design optimization of the geometry (joint position and directions) and the topology (joint distribution and connection) of tree-type robotic systems based on the above exponential coordinate system expression. As stated above, tree-type systems represent a versatile system expression of mechanical systems comprising multiple serial link chains branching from the root. These results can be extended to floating base and closed-chain systems to enlarge the system framework. For the optimization, coding of the system parameters by using a genetic algorithm (GA) is demonstrated. The closed-form formulas of the kinematics and dynamics allow for cost evaluations through numerical simulations with feedback control. Design examples of a robotic platform and a grasping/manipulation system illustrate the proposed global optimization process.

論文要旨

近年，我々の生活の様々な場面でロボットが活躍しており，例えば，工場などでは様々な作業が，工作機械やロボットにより自動化されている．そのようなロボットは，人間には真似が出来ないような高速・高精度な動作を行うことができる一方，その作業は決められた単純な繰り返し動作によるものが多く，汎用性に乏しいことがロボットの適用範囲の拡大の妨げとなっている．一方，人間は速度や精度の面ではロボットに及ばないが，工具など，これまでに開発された様々な道具を利用して作業することで，ロボットには難しい複雑な作業を実現している．そのような，物体の把握と操りの操作を伴う作業が，ロボットが特に苦手とする分野であり，現在実現されている作業の多くは，ピックアンドプレースや直線的な溶接作業など，比較的単純なものに限定されている．近年，人間の動作を模倣して繰り返し作業を行うロボットなど，より複雑な作業を行うロボットもいくつか見られるようになってきているが，その作業はいまだ限定的である．今後，ロボットに要求される作業は，より複雑で多様化することが予想され，ロボットは，さらなる高度な要求に応えていく必要がある．

そのような要求に応えるロボットを開発していくためには，より柔軟かつ汎用的な，タスク指向の最適設計手法の確立が重要と予想される．そのためには，ロボットのモデル化や制御における新しい運動表現の導入と，それに基づく，最適化手法の枠組みの構築が必要と考えられる．ロボットの運動表現の分野では，従来，Denavit-Hartemberg (DH) パラメータが多く用いられている．また，最適化手法としては，勾配法など，関数の微分情報に基づくものが多く提案されている．しかしながら，DH パラメータは，各関節の特性を根元から順に相対的に定義していくパラメータであるため，途中の関節の特性変更が以降の関節にも影響を与え，そのため，ロボットの機構の最適設計には用いづらい．また．勾配法を用いた最適解の探索は，基本的に局所的なものとなるため，最適設計では，生物の運動機構など既存の機構をベースとして用い，その機構に対し，関節の長さや向きなどを調整するに留まるものが多い．

上記の問題を解決するため，本論文の前半部では，まず，ロボットマニピュレータなどの運動機構を抽象化した，ツリータイプロボットシステムに対し，指数座標と呼ばれる運動表現を用いたモデル化と制御の方法を提案する．指数座標は，積や微分に優れた演算規則を有しているため，関節の運動を簡潔に定義でき，複雑な構造の多リンク系をモデル化するのに有用な運動表現である．各関節の特性（関節位置や運動方向）の表現には，指数パラメータと呼ばれる変数が用いられる．議論が過度に抽象的になるのを避けるため，本論文では，特に，ロボットマニピュレータによる物体の把握・操りの操作を取り上げ，そのモデル化と制御のプロセスを通して，提案手法の枠組みを説明していく．関節の分配や結合関係（システム構造）の表現には，チェーン行列と呼ばれる (0,1) 要素の行列を新たに提案する．指数パラメータとチェーン行列を用いることで，ロボットと物体の運動方程式，および，ロボットと物体間の拘束条件を，閉公式の形で陽に表現することができる．提案する運動表現の主な利点は，複雑な機構のロボットを簡潔に定義できること，および，それによりロボットの機構をある設計から他のものへ容易に変更できる点にある．実際，提案する運動表現を用いれば，関節やリンクの追加・削除，関節の結合関係や関節の位置と運動方向の変更も容易に行うことが出来る．動力学シミュレータの構築も，与えられた指数パラメータとチェーン行列から容易に行える．提案するモデル化と制御系設計のプロセスは，アーム・ハンドシステムを用いた物体の把握・操りの制御の数値例を通して検証する．

加えて，本論文では，上記指数座標を用いた運動表現を利用して，ツリータイプシステムの幾何構造（関節位置と運動方向）と位相構造（関節の分配と結合関係）を同時に最適化する，大域的な最適設計法の提案も行う．上述したように，ツリータイプシステムは，ロボットマニピュレータなどの運動機構を抽象化したもので，根元から分岐する複数のシリアルリンクチェーンから構成される．根元から分岐するシリアルリンクチェーンという制約は強いものではなく，論文の後半部では，まず，上記前半部で得られた結果が，若干の変更により，固定面を有しない浮動ベース系やリンクの先端が結合した閉リンク系へ拡張できることを示す．次に，数値最適化のための枠組みとして，遺伝的アルゴリズム (GA) の利用を想定した，指数パラメータとチェーン行列の符号化の方法を示す．運動学や動力学モデルが閉公式の形で得られていることを利用すれば，最適化におけるコストの評価を，フィードバック制御を用いた数値シミュレーションを利用して行うことも可能となる．提案する大域的最適化法のプロセスは，プラットフォームシステムとマニピュレーションシステムに対する最適構造設計の数値例を通して検証する．

Contents

List of Figures

List of Tables

Chapter 1

Introduction and preliminaries

1.1 Introduction

1.1.1 Background

Recent decades have witnessed an exponential increase in the robotic influence on human society. Robots are now employed everywhere, e.g., in industries, businesses, shops, and even in our houses. This infatuation with robots is fueled by their seemingly bottomless potential: they can work without rest, ensure high precision, and can even cooperate because of the recent breakthroughs in artificial intelligence. Indeed, this exponential growth is attributed to the massive scientific research backup in several fields connected to robotics: mechanical engineering, electronic engineering, informatics, control theories, artificial intelligence, computer vision, and overall improvements in computing capacities.

Historically, robots started as a science-fiction entity from a play written by Czech playwright Karel Čapek in 1920. Then, a master-slave architecture—remotely controlled mechanical manipulator—was developed for the first time after World War II. In 1949, force feedback was added to maintain the slave manipulator from crushing glass. For the same type of manipulator, numeric computation was improved to enhance the performance of these robots. The first robot to be implemented in the industry was the Unimation robot in a General Motors plant in 1961, where the key innovation was the possible programming of the machine. One of the major basic systems was designed in 1965, when the Stewart platform [2]–a positional platform–allowed for precision positioning. Overall, in an attempt to automate tasks in the industry, manipulator-type robots such as the Unimation PUMA or the Cincinnati Milacron T^3 received considerable attention.

However, a drastic improvement in robotic abilities was achieved only in the beginning of the 21st century. In 2000, the humanoid robot ASIMO [3] from Honda was the first humanoid robot that could run, thus closing the gap between human behaviors and robots. In 2002, the vacuum cleaner robot ROOMBA from iRobot found his way into our houses and helped with basic chores. In 2006, another humanoid robot named NAO was developed by Aldebaran robotics to assist patients with rehabilitation [4]. In 2013, another humanoid robotic evolution was performed with the ATLAS [5] robot from Boston dynamics with even more precise movements, even allowing for backflips. Finally, in more recent years, the robotic dog

SPOT [6] developed by Boston dynamics was another big step in introducing robots into our everyday lives, aiming for cooperative work in disaster areas. From a manipulator point of view, basic manipulators were replaced with generic 6-DOF manipulator systems or robotic hands. However, some problems remain in their design, especially in the dexterity department. A limited number of possible grasps, inefficiency of arm movements, and lack of fine force control are the massive challenges to overcome to improve their usefulness.

Although these robots are an advanced form of structure that we have designed, the tasks imposed on these robots are still considerably basic. The ROOMBA robot can perform some cleaning, ASIMO can run, and the robotic arms can pick up parts and move them around. However, the tasks demanded for robots are becoming increasingly complex, and these tasks need more precision and speed; in some cases, the tasks require cooperation between robots and humans or even cooperation between robots themselves. Most of the aforementioned robots use the biomimetic approach; this means that their design is inspired by existing species in nature or by humans. In the future, however, with the ever-increasing demand imposed on robots, these designs are bound to change and become more complex, adapting to the new challenges of robotic design. One of the main research challenges that has emerged in recent years is the lack of doubt about robotic construction [1]. Robotic construction, or more generally collective robotic construction, is the combination of multiple engineering skills accumulated by humans in recent years to fulfill one of humanity's oldest dream: complete automation of the construction process. Indeed, the construction process varies vastly from the construction of a house to the assembly of a car; however, the underlying skills and theories are considerably similar, and they are summarized in Fig. 1.1 [1]. In this figure, the skills required by robots are shown on the outer circle, and the sub-categories for robotic construction are shown in the inner circle.

Among these fields, we focus our attention on the combination of robotic design, robotic control, and robotic optimization in this thesis. The flagships of these fields are the robotic manipulator and robotic arms, which quickly took over the industries in recent years in manufacturing sectors, allowing for faster and more efficient manufacturing processes. Their abilities are still limited, and one of the recent challenges of these manipulators is their improvement in several aspects. To grasp objects and manipulate them on the same level as human hands, their precision must be improved

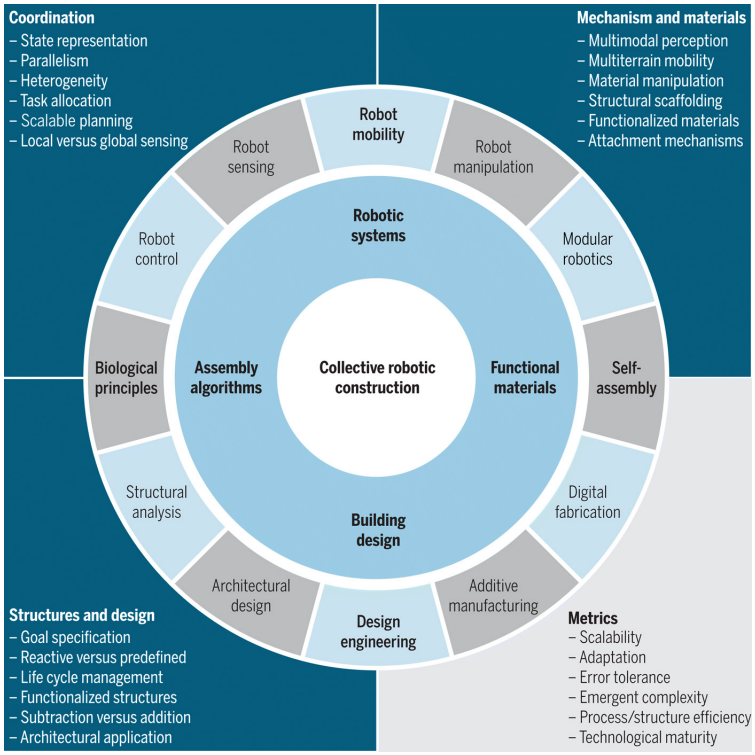


Figure 1.1: Different fields of robotic construction (From [1])

to allow them to cover more difficult tasks. To improve their cost effectiveness, they must become more versatile and complete several types of different tasks. Finally, to implement them in an embedded system and mimic the behavior of humans, these improvements must occur while limiting the load on the physical parameters of the systems such as the total mass or size of the system. To achieve these improvements, we need to ask ourselves one key question: **"How can we improve existing robotic design methods?"** In this thesis, we focus on the development of new tools for robotic design; in particular, we focus on robotic manipulators for object grasping using a combination of exponential coordinates and genetic algorithms.

1.1.2 Previous Works

1.1.2.1 Background overview and research direction of the thesis

With the rapid progress of robotic technology, robots have been implemented in our everyday life to perform various tasks autonomously or with the aid of humans [7, 8, 9]. The analysis, design, and control of robotic systems have attracted increasing attention in several domains [7, 8, 9, 10, 11].

Thus far, several robots have been designed for industrial and domestic applications [12, 13, 14, 15, 16]; such robots can perform various tasks efficiently and are therefore finding increasing use in daily life. However, if we need the robots to complete various tasks autonomously and as precisely as humans, a considerable challenge needs to be overcome to replicate human skills. Since humans perform most tasks with their hands and/or with tools specifically designed for human hands, robotic manipulation has been attracting significant attention in many fields.

The modeling and control of multifingered robot hands for grasping an object has been studied extensively in the past several decades. A kinematic relation between a finger and an object was developed in [17]. The modeling and control of the systems were proposed separately depending on the contact types: point contact [18], rolling contact [19], and rolling/sliding contact [20]. The pinching motion achieved with a set of dual fingers with soft tips was investigated in [21] and the representations of internal forces were reported in [22]. Surveys of related topics are available in [23, 24, 25].

These papers addressed the manipulation problem at the hand level. However, to reproduce human skills, the modeling and control of more complex architectures such as arm–hand systems should be investigated further. An intuitive approach toward applying the above results to the arm–hand systems is to divide the system into two (hand and arm) parts in the modeling and control design process and then combining them in an approximate manner [26, 27]. This approach is simple, and in practice, it yields a certain solution. However, we will require a more accurate and thorough design framework for more complex architectures that are beyond the arm–hand architectures such as legged vehicles and humanoid robots. These systems usually comprise chains of links and joints branching at several points, and they are referred to as tree-type systems in some studies [28].

In this thesis, inspired by the tensegrity principle of structural networks [29, 30, 31, 32], we introduced a (0,1) element matrix called the chain matrix and derived closed-form formulas of the kinematics and dynamics of tree-type systems on exponential coordinates. Based on this formulation, the system geometry (joint position and directions) was determined from the exponential parameters, and the system topology (joint connection and distribution) was determined from the chain matrix. This parametrization is considerably useful for optimization as a simple coding rule is formulated

because of the independence of the exponential parameters and the (0,1)-valued elements of the chain matrix. Physical parameters such as the mass and link length can be determined from the joint position, and thus, from the exponential parameters. These results are summarized in [33].

Based on the above results, we also propose a simultaneous (or global) geometry and topology optimization scheme for tree-type robotic systems based on exponential coordinates with the chain matrix expression. To enlarge the system framework, the results for tree-type systems are first extended to floating-base systems (for humanoids, mobile robots, etc. [34, 35]) and closed-chain systems (for parallel link manipulators, robotic platforms, etc. [36, 37]). Global optimization problems are large-scale and highly nonlinear, and they involve mixed continuous and discrete variables. In this thesis, we employ a genetic algorithm (GA) [38, 39, 40] because gradient-based approaches [41, 42, 43, 44] may be inefficient for such problems. When coding the GA, the exponential parameters can be efficiently coded using binary strings with rate decoding in a unified manner. For the chain matrix coding, the possible (0,1) patterns for tree-type systems are investigated in detail to exclude unnecessary search spaces and a minimal set of binary strings is defined. The closed-form kinematics and dynamics for tree-type systems can be obtained from the strings, and a controller of any type can be constructed from them; this allows performing numerical simulations to evaluate various costs (e.g., workspace and work [43, 44, 45]) in the optimization process, for which analytical formulas are not available. Further, a cost evaluation process that uses kinematics and dynamics with feedback control is illustrated. Two numerical design examples of a robotic platform and robotic manipulator are presented to illustrate the proposed optimization process.

As seen above, the field of robotic design optimization is based on two fundamental choices to conduct optimizations. The choice of parameters describing the kinematics and dynamics of the robot, and the optimization strategy that will be employed. In the following, we will focus on these topics.

1.1.2.2 Choice of parameters

As the first choice, we choose how to derive the equation of motion of such systems. Several methods exist that involve different parameters and computation methods.

For robotic systems, the most widely used coordinates include the

Denavit–Hartenberg (DH) parameters [7, 46, 47, 48]. The DH parameters are the standard for industrial robots; several researchers employ the DH parameters for more complex tree-type systems [28], where they derived a recursive algorithm for the dynamics. However, these parameters are defined sequentially from the root joint to the tip and are not intuitive from a geometrical viewpoint, and therefore, it is difficult to use them in the optimization process due to the on relevant dependencies. Some researchers optimized the kinematic path and other cost functions based on the DH parameters [49, 50, 51, 52].

Another set of parameters is the dual quaternion, which provides a simultaneous representation of the position and attitude using a pair of quaternions [53]. Although they have been used to describe forward and inverse kinematics in robotics [54], their use for the dynamics of complex systems such as tree-type systems is still in the nascent stages [55]. Further, they have been used in computer-assisted 3D representations and spacecraft attitude control [55, 56]. A quaternion-based optimization scheme for simple serial link robots was proposed in [57]. Because the quaternion was originally developed for the description of the attitude of a rigid body in free space without singularity, their application to complex robotic structures and optimization is minimal because of the large number of parameters involved and the complexity of the computation required.

Another representation of rigid body motion, called exponential coordinates [8], can be considered an alternative to these schemes. Exponential parameters include four parameters that define joint properties: initial position, translational movement axis, rotational movement axis, and displacement along the axis. These parameters are defined with respect to a single base frame, and thus, it is easy to change their properties without affecting other joint parameters. The position/orientation and velocity of a frame can be described by the exponential of the exponential parameters. The exponential function has elegant rules regarding the product and the derivative, which is beneficial for deriving closed-form formulas of the kinematics and dynamics for serial link chains. One such optimization based on the product of exponentials was described in [58], in which only the kinematics of some specific structures were considered.

1.1.2.3 Optimization methods

The second choice covers the optimization strategy used for the mechanical structure. Historically, most of these studies have a fairly narrow

application field because they consider structural parameters specific to the system at hand and the gradient-based optimization approach is employed. While these methods are effective in their own right, they consider only case-by-case problems and do not offer flexible overall structural designs [42, 43, 44]. In [41, 59], some general guidelines were provided; however, they were still restricted to simple systems. In [60], which covers a classic optimization scheme, the need for more general optimization schemes is highlighted by the authors. In addition, in [61, 12, 13], several optimization methods were presented, and the need for more global methods involving an increasing number of parameters is hinted.

Because DH parameters are the most prevalent choice of coordinate representations, it is only natural that most optimization schemes are based around them. In [49], a scheme was proposed to optimize the manipulability and torque of a 6-DOF classic manipulator. In [50], a more general scheme of DH parameter optimization was presented. Further, the optimization of the kinematic path and other cost functions based on the DH parameters can be found in [51, 52]. For quaternions, as stated above, their introduction to robotic kinematics and dynamics description is rather recent, and as such, optimization schemes based on them are rather sparse. A good example of an optimization scheme of a simple serial link robot based on quaternions can be found in [50].

In general robot designs, the system architecture—the joint distribution and connection (topology) and their position and movement directions (geometry)—is first determined by considering intuitive or biologically inspired approaches. Then, for the choice of architecture, physical parameters such as mass/inertia and link length are designed or optimized, as reported in [62, 60, 63]. These designs are simple, and several satisfactory solutions can be obtained; however, bio-inspired designs might be inefficient or intractable if more specific or complex design specifications [60, 59, 64] are required. Therefore, a scheme involving systematic simultaneous optimization of geometrical and topological properties is required [12, 61].

In recent years, evolutionary algorithms such as genetic algorithms (GA) have been introduced to robotic design problems, and they show the most promise for solving optimization problems in a complex environment that composes a large number of parameters. Evolutionary algorithms have been introduced long ago [38, 65, 66, 67]; however, their applications in robotic designs are at their premices because one of the major obstacles

to overcome is to properly include the robotic design parameters into those algorithms to yield some significant results. In [64, 39], they were introduced to optimize a robotic gripper, which is the first step to a general approach for overall robotic structures. In [40], an application to a 6-DOF classic manipulator was also reported, which proved their efficiency compared to classical schemes such as [49]. Finally, a more general approach was attempted in [68]; however, the results are still not provided in a global manner.

Thus, we can conclude that the field of optimal robotic structure lacks two fundamental points: a generalized approach that can encompass a wide array of problems and the maximization of the number of general parameters to include in the optimization problem.

1.1.3 Objective of the thesis

1.1.3.1 Kinematics and dynamics of tree-type systems

Primarily motivated by the works of [8, 38], the global objective of this thesis is to propose a general approach to robotic structure optimization based on a combination of exponential parameters and GAs.

Because exponential parameters allow for a flexible choice of design parameters, the first step is to generalize the exponential parameter theory to more complex structures, mainly to tree-type structures. By observing these structures, the parameters we consider in our study are geometrical (joint positions and orientation) and topological properties (joint distribution and connection). To the goal of the development of optimization tools for these complex structures, the control strategies as well as closed-form formulas the kinematics and dynamics of these systems will be provided.

1.1.3.2 Combination of exponential coordinates and genetic algorithms

When this generalization is acquired, the second step is to adapt the GA process and code the exponential parameters and the chain matrix with a smart conversion between the binary approach of the GAs and the continuous nature of the parameters describing the exponential parameters. For the chain matrix, possible (0,1) patterns are precisely investigated to code the possible structure. This approach allows us to conduct a simultaneous optimization of the system geometry and topology.

Finally, the results yielded by those designs and optimizations are pre-

sented and analyzed, and we provide prospects of the proposed solutions into the general field of robotic structure optimization as a whole. To illustrate our theories, we consider the robotic manipulation approach; the proposed theories can be applied to other robotic tasks as well.

1.1.4 Contribution of the thesis

The contributions of this thesis in chapters 2 and 3 are listed below.

- Chapter 2
  - Definition of the chain matrix that defines the kinematic path taken within the structure of a branching robotic architecture.
  - Description of tree-type systems kinematics and dynamics using the chain matrix that allows for a unified framework for the manipulator and object dynamics in a closed-form manner.
  - Extension of the constraint equation governing the contact relation between manipulator fingertips and object contact points, thereby allowing for improved control covering both object motion control and manipulator redundancy control.
  - Numerical example of an arm-hand robotic systems grasping an object using the proposed theories.
- Chapter 3
  - Extension of the exponential parameters and chain matrix combination to cover more complex robotic designs including floating-base systems, closed-loop systems, and platform systems.
  - Adaptation of the binary coding of the GAs to exponential parameters and the chain matrix, thereby allowing for wide-scale parameters robotic design optimization including simultaneous geometric and topological optimizations.
  - Proposition of costs evaluation by kinematics and dynamics control for the optimization process
  - Design optimization examples based on the arm-hand robotic systems observed in Chapter 2

1.1.5 Organization of the thesis

This thesis is divided in three major parts:

In the first part, we introduce the main concepts used in this thesis. In section 1.2, we focus on several parameter schemes used for robotic kinematic analysis including DH parameters, dual quaternions, and the main

topic of the thesis: exponential coordinates. In section 1.3, we introduce the basic concepts of GAs and their usefulness in robotic design optimization schemes.

In chapter 2 after describing our motivation and a brief summary of rigid body motion in exponential coordinates in section 2.1-2.3, we first define tree-type systems, the chain matrix, and the simplified chain matrix in section 2.4. Using these definitions, we extend the exponential coordinate theories to define tree-type system kinematics and dynamics in section 2.5. In section 2.6, we describe the control variables of such a system and extend the previously used constraint equation to account for redundancy in the mechanisms, which is a common feature of complex robotic manipulators. In section 2.7, we focus on the control strategies used for those systems and summarize the total design approach in section 2.8. Using this approach, numerical examples are provided in section 2.9 focusing on the development of an arm-hand system, and we present the benefits and drawbacks of the proposed method in section 2.10.

In chapter 3, we briefly review tree-type structures and introduce their implementation into the optimization scheme in section 3.1. Before focusing on the optimization, we extend the previous theories to other complex architectures—floating-base systems, closed-loop systems, and platform systems—in section 3.2. The combination of GAs and exponential coordinates with the chain matrix is explored in sections 3.3, 3.4, and 3.5, where we focus on the coding of the different exponential parameters in binary expressions, the coding of the chain matrix describing the architecture of the structure, and we summarize the overall genetic algorithm procedure. We propose a cost evaluation process for the GA-driven robotic design optimization in section 3.6. Finally, the optimization results and design examples are observed and discussed in sections 3.7 and 3.8, and we conclude the GA approach in section 3.9.

Finally, we present all contributions, the efficiency of our overall procedures, and future prospects in the conclusion in chapter 4.

For preliminary, we review the parameters for the rigid body motion and genetic algorithms.

1.2 Rigid body analysis motion parameters

1.2.1 Rigid body motion

The basics of kinematics, dynamic analysis, and control of mechanical systems have been studied over several decades. Several books treat the matter from a basic to an advanced level [69] [70] [7].

In robotic systems dynamical analysis, we study rigid body motion. A rigid motion of an object is a continuous movement of the particles in the object such that the distance between any two particles always remains fixed. The net movement of a rigid body from one location to another via rigid motion is called a rigid displacement. A rigid displacement may consist of both translation and rotation of the object. The two components of this displacement are denoted by $p \in \mathbb{R}^3$ for translation and $R \in \mathbb{R}^{3\times3}$ for rotation. The interpretation of the translation is straightforward (it is simply the vector linking the start position and end position of the movement); therefore, we focus more on the rotational aspect of the motion.

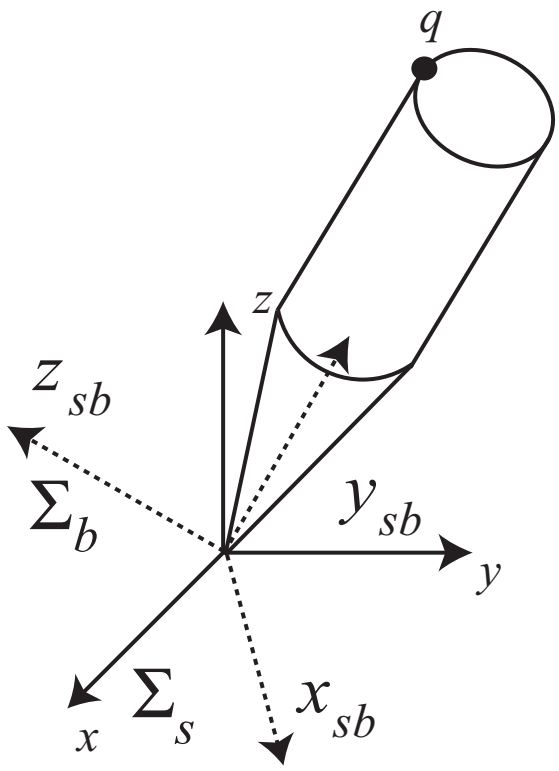


Figure 1.2: Rotation of a rigid body

To observe rotational movement, we first define a base coordinate frame (all coordinate frames are assumed to be right-handed) and we attach another coordinate frame to the center of mass of the observed object. In

Fig. 1.2, we define Σ_s as the spatial frame (an arbitrary frame in $\mathbb{R}^3$) and Σ_b as the body frame, where x_{sb}, y_{sb}, and $z_{sb} \in \mathbb{R}^3$ are the coordinates of the principal axes of Σ_b relative to Σ_s. By stacking these coordinate vectors next to each other, we define a 3×3 matrix as

$$R_{sb} = \begin{bmatrix} x_{sb} & y_{sb} & z_{sb} \end{bmatrix}. \tag{1.1}$$

Such a matrix is called a rotation matrix: every rotation of the object relative to the ground corresponds to a matrix of this form. This matrix exhibits two important properties:

$$\begin{aligned} RR^{\mathrm{T}} &= R^{\mathrm{T}}R = I \\ \det(R) &= 1. \end{aligned} \tag{1.2}$$

The set of all 3×3 matrices that satisfy these two properties is denoted as $SO(3)$, which is an abbreviation for a special orthogonal. We define the space of rotation matrices in $\mathbb{R}^{3\times3}$ as

$$SO(3) = \{R \in \mathbb{R}^{3\times3} : RR^{\mathrm{T}} = I, \det(R) = 1\}. \tag{1.3}$$

Finally, a rotation matrix also serves as a transformation, taking the coordinates of a point from one frame to another. Consider the point q shown in Fig. 1.2. Let $q_b = (x_b, y_b, z_b)$ be the coordinates of q relative to the frame Σ_b. The coordinates of q described in frame Σ_s are given by

$$\begin{aligned} q_s &= x_{sb}x_b + y_{sb}y_b + z_{sb}z_b \\ &= \begin{bmatrix} x_{sb} & y_{sb} & z_{sb} \end{bmatrix} \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} \\ &= R_{sb}q_b. \end{aligned} \tag{1.4}$$

That is, R_{sb} rotates the coordinates of a point from frame Σ_b to frame Σ_s.

Further, the rotation matrix can be combined to link several bases, wherein we use the Chasles relation given by

$$R_{ac} = R_{ab}R_{bc}. \tag{1.5}$$

In addition, we define vector cross products using skew-symmetric matrices such as

$$a\times = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \times \equiv \widehat{a} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}. \tag{1.6}$$

To describe motion in a 3D space, the rotation matrix is thoroughly analyzed and several representations for a 3D rotation are provided, as described in the following section.

1.2.2 Representations of rigid body rotation

Several methods exist for the representation of rigid body rotation matrix, and they all have their advantages and drawbacks. In this section, we briefly describe these methods and comment on their usefulness.

1.2.2.1 Euler angles

The first and most commonly used angles are Euler angles. The method is as follows: start with a frame Σ_b coincident with frame Σ_s. First, the Σ_b frame is rotated about the z-axis of frame Σ_b (at the time coincident with frame Σ_s) by an angle α, and then, rotate it about the new y-axis of frame Σ_b by an angle β; finally, rotate it once more about the new z-axis of angle γ. This yields a new orientation $R_{sb}(\alpha, \beta, \gamma)$, and the three angles represent the rotations. Indeed, it is possible to rotate the frame about any axis in this method, which allows several possibilities for the rotation definition (the one presented above is called the ZYZ Euler angles because we rotate it about the Z, Y, and Z axes of the evolving Σ_B frame). Mathematically, this rotation is noted as

$$\begin{aligned} R &= R_z(\alpha)R_y(\beta)R_z(\gamma) \\ &= \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} c_\alpha c_\beta c_\gamma - s_\alpha s_\gamma & -c_\alpha c_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta \\ s_\alpha c_\beta c_\gamma + c_\alpha s_\gamma & -s_\alpha c_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta \\ -s_\beta c_\gamma & s_\beta s_\gamma & c_\beta \end{bmatrix}. \end{aligned} \tag{1.7}$$

Using the time derivation rotation matrix $\dot{R}$, it is possible to relate those angles to the angular velocity vector ω as

$$\omega = T \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix}, \tag{1.8}$$

where T is called the *transformation matrix*. The problem here lies in the fact that the inverse of T is not always guaranteed, which can lead to

singularities in the mathematical expression.

Advantages:

- Easy to understand and very intuitive.
- Minimalistic because they require only three parameters.

Drawbacks:

- The expression of the translation and rotation are de-coupled.
- Euler angle suffers from singularities, which means there exists mathematically impossible configurations for the movement where it is actually feasible in reality because of the absence of the matrix inverse in certain cases.
- The derivation process involving Euler angles can be tedious, and therefore, it is not well-suited for general optimization schemes.

1.2.2.2 Quaternions

Another representation of the rigid body motion is a quaternion. A basic quaternion is defined by

$$\tilde{q} = q_0 + q = q_0 + iq_1 + jq_2 + kq_3, \tag{1.9}$$

where i, j, and k are units with the property $i^2 = j^2 = k^2 = ijk = -1$. The quaternion can be identified by a four-dimensional vector as $\tilde{q} = \begin{bmatrix} q_0 & | & q_1 & q_2 & q_3 \end{bmatrix}^{\mathrm{T}}$. A position vector $r \in \mathbb{R}^3$ can be associated with a quaternion as $\tilde{r} = \begin{bmatrix} 0 & | & r^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$. The sum, product, and conjugate operations are defined as

$$\tilde{q} + \tilde{p} = (q_0 + p_0) + i(q_1 + p_1) + j(q_2 + p_2) + k(q_3 + p_3) \ , \tag{1.10}$$

$$\tilde{q} \otimes \tilde{p} = q_0 p_0 - q.p + q_0 p + p_0 q + q \times p \ , \tag{1.11}$$

$$\tilde{q}^* = q_0 - iq_1 - jq_2 - kq_3, \tag{1.12}$$

and the norm and inverse are given respectively by

$$\|\tilde{q}\| = \sqrt{\tilde{q} \otimes \tilde{q}^*} = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} \ , \tag{1.13}$$

$$\tilde{q}^{-1} = \frac{\tilde{q}^*}{\|\tilde{q}\|^2} \ . \tag{1.14}$$

The orientation of a rigid body (or frame) can be represented by a unit quaternion. Using the axis of rotation n and the angle of rotation θ, the orientation can be specified by a unit quaternion

$$\tilde{q} = \cos\frac{\theta}{2} + n\sin\frac{\theta}{2} \ . \tag{1.15}$$

If we consider a rotation of r by $\tilde{q}$, i.e., if we rotate r around the n axis by θ, the resultant vector r' is given from its quaternion counterpart $\tilde{r}'$ as

$$\tilde{r}' = \tilde{q} \otimes \tilde{r} \otimes \tilde{p^*}. \tag{1.16}$$

Similarly, a rotated vector r'' of r by another unit quaternion $\tilde{q}'$ is given by $r'' = \tilde{q}' \otimes \tilde{r} \otimes \tilde{p'}^*$.

Finally, the time derivative of $\tilde{q}$ is related to the angular velocity vector ω as

$$\dot{\tilde{q}} = \frac{1}{2} \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \begin{bmatrix} 0 \\ \omega \end{bmatrix}, \tag{1.17}$$

and from $\|\tilde{q}\| = 1$, its inverse always exists and is given by

$$\begin{bmatrix} 0 \\ \omega \end{bmatrix} = 2 \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \\ -q_1 & q_0 & q_3 & -q_2 \\ -q_2 & -q_3 & q_0 & q_1 \\ -q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \dot{\tilde{q}} \ . \tag{1.18}$$

Advantages:

- No singularities because the inverse is always available.
- Unified representation of the translation and rotation.

Drawbacks:

- Complex formulation of the mathematics involved.
- Difficulty of their implementation in a complex structure involving several paths or closed loops.

1.2.2.3 Denavit–Hartenberg parameters

Finally, the industry standards for the representation of rigid body motion are the DH parameters. These parameters are shown in Fig. 1.3, and they describe the relative position from one joint in the system to the next one in a successive manner.

In order to define the D-H parameters, we first attach coordinates frames to every rigid body i in the system:

- The z axis is in the direction of the joint axis.
- The x axis is parallel to the common normal ($x_i = z_i \times z_{i-1}$). If there is no common normal (e.g. translational joint), d is a free parameter. The x axis is a free choice for the first joint of the system.

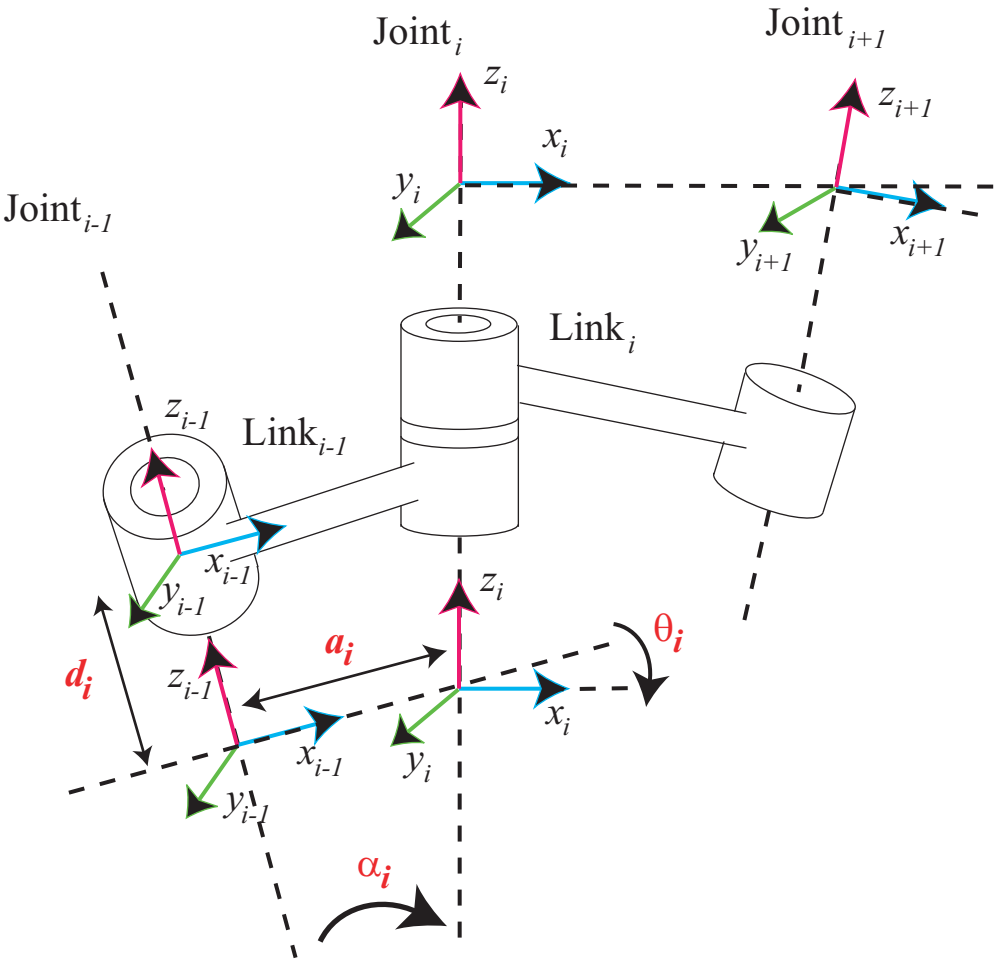


Figure 1.3: Denavit–Hartenberg parameters definition

- The y axis is taken with the x and z axis to complete a right-handed coordinate system.

The following four transformation parameters are known as the D-H parameters.

- d is the offset along the previous z_{i-1} axis to the common normal (the difference in height on z between the coordinate frame of joint $i-1$ and joint $i+1$).
- θ is the angle about the previous z_{i-1} axis, from the old x_{i-1} axis to the new x axis.
- a is the length of the common normal.
- α is the angle about the common normal, from the old z_{i-1} axis to the new z axis.

Using these parameters, it is possible to define the position of a body i with respect to i-1 using a transformation matrix as

$$
\begin{aligned}
{}^{i-1}T_i &= \begin{bmatrix} \cos(\theta) & -\sin(\theta)\cos(\alpha) & \sin(\theta)\sin(\alpha) & a\cos(\theta) \\ \sin(\theta) & \cos(\theta)\cos(\alpha) & -\cos(\theta)\sin(\alpha) & a\sin(\theta) \\ 0 & \sin(\alpha) & \cos(\alpha) & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix},
\end{aligned} \tag{1.19}
$$

where R and T are the rotation matrix and translation vector, respectively.

Advantages:

- Unified representation of the translation and rotation.

Drawbacks:

- Singularity expressions of the attitude remains.
- The relation between the joints becomes very non-intuitive if they are not situated at perfect right-angles.
- The parameters are defined from one base to another, meaning that they are inter-dependent.

1.2.3 Exponential coordinates

The principal motivation for this thesis is the extension of the exponential coordinates theory to more complex architectures. As such, this section serves as a reminder of the basic theories used for exponential coordinates.

1.2.3.1 Rigid body motion

As stated above, a common motion encountered in robotics is the rotation of a body about a given axis by some amount. In this case, we define $\omega \in \mathbb{R}^3$ (different from the angular acceleration vector) as a unit vector that specifies the direction of rotation, and $\theta \in \mathbb{R}$ as the angle of rotation in radians. Because every rotation of the object corresponds to some $R \in SO(3)$, we would like to write R as a function of ω and θ.

For our derivation, we consider the velocity of a point q attached to the rotating body. If we rotate the body at a constant unit velocity about the axis ω as shown in Fig. 1.4, the velocity of point $\dot{q}$ is given by

$$\dot{p}(t) = \omega \times p(t) = \widehat{\omega} p(t). \tag{1.20}$$

This is a time-invariant linear differential equation that can be integrated to provide

$$p(t) = e^{\widehat{\omega} t} p(0), \tag{1.21}$$

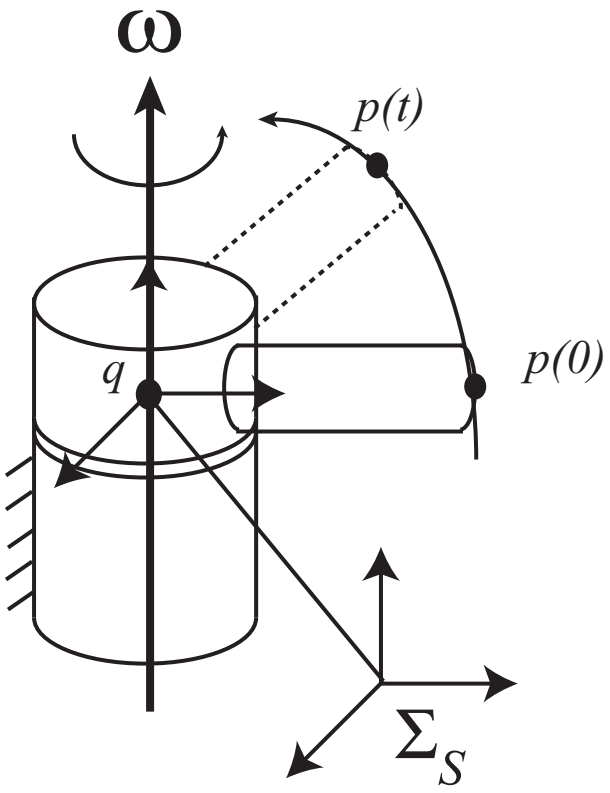


Figure 1.4: Description of the rotation axis ω

where $q(0)$ is the initial ($t = 0$) position of the point, and $e^{\widehat{\omega}t}$ can be developed using Taylor's development

$$e^{\widehat{\omega}t} = I + \widehat{\omega}t + \frac{(\widehat{\omega}t)^2}{2!} + \cdots . \tag{1.22}$$

It follows that, if we rotate about the axis ω at unit velocity for θ units of time, then the net rotation is given by

$$R(\omega, \theta) = e^{\widehat{\omega}\theta}. \tag{1.23}$$

We denote the vector space of all 3×3 skew matrices such as ω by $so(3)$. Finally, by developing the infinite series for the exponential and using the Rodrigues formula, we can rewrite Eq. (1.22) as

$$e^{\widehat{\omega}\theta} = I + \widehat{\omega}sin\theta + \widehat{\omega}^2(1 - cos\theta) \tag{1.24}$$

$$= \begin{bmatrix} \omega_1^2 v_\theta + c_\theta & \omega_1\omega_2 v_\theta - \omega_3 s_\theta & \omega_1\omega_3 v_\theta + \omega_2 s_\theta \\ \omega_1\omega_2 v_\theta + \omega_3 s_\theta & \omega_2^2 v_\theta + c_\theta & \omega_2\omega_3 v_\theta - \omega_1 s_\theta \\ \omega_1\omega_3 v_\theta - \omega_2 s_\theta & \omega_2\omega_3 v_\theta + \omega_1 s_\theta & \omega_3^2 v_\theta + c_\theta \end{bmatrix}, \tag{1.25}$$

where c_θ and s_θ are the cosine and sine of θ and $v_\theta = 1 - c_\theta$, respectively.

We can describe the configuration of the movement of the rigid body frame by associating $p_{sb} \in \mathbb{R}^3$ and $R_{sb} \in \mathbb{R}^3$, respectively, i.e., the translation and rotation components from the fixed base frame Σ_s to the moving body frame Σ_b. These configurations are represented by the pair (p_{sb}, R_{sb}), and they belong to a group denoted by the special Euclidian or $SE(3)$ group such that

$$SE(3) = \{(p, R) : p \in \mathbb{R}^3, R \in SO(3)\} = \mathbb{R}^3 \times SO(3). \tag{1.26}$$

As such, given q_b (the point viewed from the body), we can find q_s (the point viewed from the base) with

$$q_s = p_{sb} + R_{sb} q_b. \tag{1.27}$$

By representing the points and vector in $\mathbb{R}^4$ instead of $\mathbb{R}^3$ using what is called homogeneous coordinates, we transform points such that $\overline{q} = \begin{bmatrix} q_1 & q_2 & q_3 & 1 \end{bmatrix}^{\mathrm{T}}$. Using this new representation, Eq. (1.27) becomes

$$\overline{q}_s = \begin{bmatrix} q_s \\ 1 \end{bmatrix} = \begin{bmatrix} R_{sb} & p_{sb} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} q_b \\ 1 \end{bmatrix} = g_{sb} \overline{q}_b, \tag{1.28}$$

where

$$g_{sb} = \begin{bmatrix} R_{sb} & p_{sb} \\ 0 & 1 \end{bmatrix}, \; g_{sb}^{-1} = \begin{bmatrix} R_{sb}^{\mathrm{T}} & -R_{sb}^{\mathrm{T}} p_{sb} \\ 0 & 1 \end{bmatrix}. \tag{1.29}$$

g_{sb} is a 4×4 matrix in 3D, which is called the homogeneous representation of the transformation, and represents rigid body motion. Rigid body transformations are linked by the Chasles relation. If we introduce an intermediate coordinate frame Σ_a, we can link those rigid body transformations by the relation $g_{sb} = g_{sa} g_{ab}$.

1.2.3.2 Twists and exponential coordinates

By applying twist theories to the rotational motion shown in Fig. 1.4, we can transform the velocity of a point p such that

$$\dot{p}(t) = \omega \times (p(t) - q), \tag{1.30}$$

which becomes

$$\begin{bmatrix} \dot{p} \\ 0 \end{bmatrix} = \begin{bmatrix} \widehat{\omega} & -\omega \times q \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ 1 \end{bmatrix} = \widehat{\xi} \begin{bmatrix} p \\ 1 \end{bmatrix}, \tag{1.31}$$

where

$$\widehat{\xi} = \begin{bmatrix} \widehat{\omega} & -\omega \times q \\ 0 & 0 \end{bmatrix} \tag{1.32}$$

is called the *twist* of the transformation. Similar to those for the skew-symmetric matrices, the twists are defined using the rotation vector ω and the translation vector v in their vector form as

$$\xi = \begin{cases} \begin{bmatrix} \omega \\ -\omega \times q \end{bmatrix} & \text{if the joint is a pure rotation} \\ \begin{bmatrix} 0 \\ v \end{bmatrix} & \text{if the joint is pure translation,} \end{cases} \tag{1.33}$$

or, as in its matrix version,

$$\widehat{\xi} = \begin{cases} \begin{bmatrix} \widehat{\omega} & -\omega \times q \\ 0 & 0 \end{bmatrix} & \text{if the joint is pure rotation} \\ \begin{bmatrix} 0 & v \\ 0 & 0 \end{bmatrix} & \text{if the joint is pure translation,} \end{cases} \tag{1.34}$$

where it holds $(\widehat{\xi})^V = \xi$. $(\cdot)^V$ is called the *vee* operation, and is similar to the opposite of the skew transformation, as it transforms the matrix form of a twist into its vector form.

By observing the two special cases of pure translational motion and pure rotational motion, and resolving Eq. (1.31) in a manner similar to Eq. (1.21), we define the exponential of twists as

$$e^{\widehat{\xi}\theta} = \begin{bmatrix} I & v\theta \\ 0 & 1 \end{bmatrix}, \tag{1.35}$$

for pure translational motion, and

$$e^{\widehat{\xi}\theta} = \begin{bmatrix} e^{\widehat{\omega}\theta} & (I - e^{\widehat{\omega}\theta})(\omega \times v) + \omega\omega^{\mathrm{T}} v\theta \\ 0 & 1 \end{bmatrix}, \tag{1.36}$$

for the general case. Note that the exponential function

$$R = e^{\widehat{\omega}\theta} \tag{1.37}$$

represents the rotation matrix. In the case of pure rotational motion viewed from Σ_b, because rotational motion also produces a translation viewed from the base, v in Eq. (1.36) is given by $v = -\omega \times q$. The exponential of a twist given in Eq. (1.36) has the same form as the rigid body transformation matrix given in Eq. (1.29).

Finally, we can derive the forward kinematics of a system composed of a serial link chain using the exponential coordinate as

$$g_{sb}(\theta) = e^{\widehat{\xi}_1\theta_1} \cdots e^{\widehat{\xi}_n\theta_n} g_{sb}(0), \tag{1.38}$$

where $e^{\xi_i \theta_i}$ are the exponential coordinate matrices for every joint i, $g_{sb}(0)$ is the initial configuration of base Σ_b as viewed from Σ_s, and $g_{sb}(\theta)$ is the rigid body motion.

1.2.3.3 Rigid body velocity using exponential coordinates

We now consider the rotational velocity of rigid body motion. We start by observing the rotational velocity of a single point and then extend it to a rigid body velocity. We have seen that the path followed by a rotating point in space is given by

$$q_s(t) = R_{sb}(t) q_b. \tag{1.39}$$

After derivation, we obtain

$$v_{q_s}(t) = \frac{d}{dt} q_s(t) = \dot{R}_{sb}(t) q_b. \tag{1.40}$$

However, this representation suggests that we need nine numbers to describe the velocity of a rotating body. A smarter approach is to transform the above equation as

$$\begin{aligned} v_{q_s}(t) &= \dot{R}_{sb}(t) q_b \\ &= \dot{R}_{sb}(t) R_{sb}^{-1}(t) R_{sb}(t) q_b \\ &= \widehat{\omega}_{sb}^{s} R_{sb}(t) q_b, \end{aligned} \tag{1.41}$$

where $\widehat{\omega}_{sb}^{s} = \dot{R}_{sb}(t) R_{sb}^{-1}(t)$ is defined as the instantaneous spatial angular velocity. This vector corresponds to the instantaneous angular velocity of the object, as seen from the spatial coordinate frame Σ_s. In a similar manner, we define $\widehat{\omega}_{sb}^{b} = R_{sb}^{-1}(t) \dot{R}_{sb}(t)$ as the instantaneous body angular velocity, which is the description of the angular velocity viewed from the body frame Σ_b. The two velocities are linked to each other as

$$\widehat{\omega}_{sb}^{b} = R_{sb}^{-1} \widehat{\omega}_{sb}^{s} R_{sb} \quad \text{or} \quad \omega_{sb}^{b} = R_{sb}^{-1} \omega_{sb}^{s}. \tag{1.42}$$

We finally see that we end up with two different velocity definitions: spatial velocity and body velocity, as

$$\begin{aligned} v_{q_s}(t) &= \widehat{\omega}_{sb}^{s} R_{sb}(t) q_b = \omega_{sb}^{s}(t) \times q_s(t) \\ v_{q_b}(t) &= R_{sb}^{T}(t) v_{q_s}(t) = \omega_{sb}^{b}(t) \times q_b. \end{aligned} \tag{1.43}$$

Let us now consider the general case where $g_{sb}(t) \in SE(3)$ represents the rigid body motion. As in the case of simple rotation, $\dot{g}_{sb}(t)$ is not useful

by itself; however, the terms $\dot{g}_{sb}g_{sb}^{-1}$ and $g_{sb}^{-1}\dot{g}_{sb}$ are interesting. Given that $g_{sb}(t) = \begin{bmatrix} R_{sb}(t) & p_{sb}(t) \\ 0 & 1 \end{bmatrix}$, we have

$$\begin{aligned} \dot{g}_{sb}g_{sb}^{-1} &= \begin{bmatrix} \dot{R}_{sb} & \dot{p}_{sb} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} R_{sb}^{\mathrm{T}} & -R_{sb}^{\mathrm{T}}p_{sb} \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \dot{R}_{sb}R_{sb}^{\mathrm{T}} & -\dot{R}_{sb}R_{sb}^{\mathrm{T}}p_{sb} + \dot{p}_{sb} \\ 0 & 0 \end{bmatrix}, \end{aligned} \tag{1.44}$$

which has the form of a twist in Eq. (1.34). By analogy to the rotational velocity, we define the spatial velocity $\widehat{V}_{sb}^{s} \in se(3)$ as

$$\widehat{V}_{sb}^{s} = \dot{g}_{sb}g_{sb}^{-1}, \tag{1.45}$$

or in vector form

$$V_{sb}^{s} = \begin{bmatrix} v_{sb}^{s} \\ \omega_{sb}^{s} \end{bmatrix} = \begin{bmatrix} -\dot{R}_{sb}R_{sb}^{\mathrm{T}}p_{sb} + \dot{p}_{sb} \\ (\dot{R}_{sb}R_{sb}^{\mathrm{T}})^{V} \end{bmatrix}. \tag{1.46}$$

In a similar manner, we define the body velocity $\widehat{V}_{sb}^{b} \in se(3)$ as

$$\widehat{V}_{sb}^{b} = g_{sb}^{-1}\dot{g}_{sb} \tag{1.47}$$

or in vector form

$$V_{sb}^{b} = \begin{bmatrix} v_{sb}^{b} \\ \omega_{sb}^{b} \end{bmatrix} = \begin{bmatrix} R_{sb}^{\mathrm{T}}\dot{p}_{sb} \\ (R_{sb}^{\mathrm{T}}\dot{R}_{sb})^{V} \end{bmatrix}. \tag{1.48}$$

These two velocities are linked together by the adjoint transformation, such that

$$V_{sb}^{s} = \mathrm{Ad}_{g_{sb}}V_{sb}^{b} \tag{1.49}$$

where

$$\mathrm{Ad}_{g_{sb}} = \begin{bmatrix} R_{sb} & \widehat{p}_{sb}R_{sb} \\ 0 & R_{sb} \end{bmatrix} \quad \left(\mathrm{Ad}_{g_{sb}}^{-1} = \begin{bmatrix} R_{sb}^{\mathrm{T}} & -R_{sb}^{\mathrm{T}}\widehat{p}_{sb} \\ 0 & R_{sb}^{\mathrm{T}} \end{bmatrix} \right). \tag{1.50}$$

These transformation rules can also be applied to constant twists defined by ξ. If ξ is a twist which represents the motion of a screw and we move the screw by applying a rigid body motion $g \in SE(3)$, the new twist ξ' can be described by

$$\xi' = \mathrm{Ad}_{g}\xi \quad \text{or} \quad \widehat{\xi'} = g\widehat{\xi}g^{-1}, \tag{1.51}$$

are the equations that allow us to transform twist vector expression into matrix expression and vice versa.

Using the properties of velocity, we can link the spatial and body velocity to the joint rotational speed using the derivation of the forward kinematics by applying the chain rule as

$$\begin{aligned}\widehat{V}_{sb}^{s} &= \sum_{i=1}^{n}(\frac{\partial g_{sb}}{\partial \theta_i}\dot{\theta}_i)g_{sb}^{-1}(\theta) \\ &= \sum_{i=1}^{n}(\frac{\partial g_{sb}}{\partial \theta_i}g_{sb}^{-1}(\theta))\dot{\theta}_i.\end{aligned} \tag{1.52}$$

By using the twist vector form expression, we can write

$$\begin{aligned}V_{sb}^{s} &= \left[(\tfrac{\partial g_{sb}}{\partial \theta_1}g_{sb}^{-1})^{V}\cdots(\tfrac{\partial g_{sb}}{\partial \theta_n}g_{sb}^{-1})^{V}\right]\dot{\theta} \\ &= J_{sb}^{s}\dot{\theta}.\end{aligned} \tag{1.53}$$

and

$$\begin{aligned}V_{sb}^{b} &= \left[(g_{sb}^{-1}\tfrac{\partial g_{sb}}{\partial \theta_1})^{V}\cdots(g_{sb}^{-1}\tfrac{\partial g_{sb}}{\partial \theta_n})^{V}\right]\dot{\theta} \\ &= J_{sb}^{b}\dot{\theta}.\end{aligned} \tag{1.54}$$

The Jacobians J_{sb}^{s} and J_{sb}^{b} can be described more specifically by applying the time derivative of the forward kinematic formula in Eq. (1.38) as

$$\begin{aligned}(\frac{\partial g_{sb}}{\partial \theta_i})g_{sb}^{-1} &= e^{\widehat{\xi}_1\theta_1}\cdots e^{\widehat{\xi}_{i-1}\theta_{i-1}}\frac{\partial}{\partial \theta_i}(e^{\widehat{\xi}_i\theta_i})e^{\widehat{\xi}_{i+1}\theta_{i+1}}\cdots e^{\widehat{\xi}_n\theta_n}g_{sb}(0)g_{sb}^{-1} \\ &= e^{\widehat{\xi}_1\theta_1}\cdots e^{\widehat{\xi}_{i-1}\theta_{i-1}}(\widehat{\xi}_i)e^{\widehat{\xi}_i\theta_i}e^{\widehat{\xi}_{i+1}\theta_{i+1}}\cdots e^{\widehat{\xi}_n\theta_n}g_{sb}(0)g_{sb}^{-1} \\ &= e^{\widehat{\xi}_1\theta_1}\cdots e^{\widehat{\xi}_{i-1}\theta_{i-1}}(\widehat{\xi}_i)e^{-\widehat{\xi}_{i-1}\theta_{i-1}}\cdots e^{-\widehat{\xi}_1\theta_1}.\end{aligned} \tag{1.55}$$

By converting into twist coordinates,

$$(\frac{\partial g_{sb}}{\partial \theta_i}g_{sb}^{-1})^{V} = \mathrm{Ad}_{(e^{\widehat{\xi}_1\theta_1}\cdots e^{\widehat{\xi}_{i-1}\theta_{i-1}})}\xi_i, \tag{1.56}$$

and the *spatial manipulator Jacobian* becomes

$$J_{sb}^{s}(\theta) = \begin{bmatrix}\xi_1 & \xi_2^{'} & \cdots & \xi_n^{'}\end{bmatrix} \tag{1.57}$$

where

$$\xi_i^{'} = \mathrm{Ad}_{(e^{\widehat{\xi}_1\theta_1}\cdots e^{\widehat{\xi}_{i-1}\theta_{i-1}})}\xi_i. \tag{1.58}$$

In a similar manner, the *body manipulator Jacobian* becomes

$$J_{sb}^{b}(\theta) = \begin{bmatrix}\xi_1^{\dagger} & \xi_2^{\dagger} & \cdots & \xi_n^{\dagger}\end{bmatrix} \tag{1.59}$$

where

$$\xi_i^{\dagger} = \mathrm{Ad}_{(e^{\widehat{\xi}_i\theta_i}\ldots e^{\widehat{\xi}_n\theta_n}g_{sb}(0))}\xi_i. \tag{1.60}$$

Using the above exponential rigid body expressions, we extend the application of exponential coordinates to more complex structures by smartly defining the kinematic path taken in these complex structures and maintaining the advantages of exponential and twists formulations.

1.3 Introduction to genetic algorithms

1.3.1 Different type of optimization algorithms

Historically, several types of optimization algorithms have been used based on their strengths and weaknesses; however, all optimization algorithms can be categorized into calculus-based methods, enumerative methods, and random methods.

Calculus-based methods have been by far the most explored and studied optimization methods. They rely on solving linear or nonlinear sets of equations by computing the gradient of the functions to search where the function gradient becomes null, which indicates a local maxima or minima. The problems with these methods are apparent. They are very powerful tools to obtain the extrema of a known function; however, they might breakdown when the problem becomes difficult to express from a mathematical point of view. Further, the obtained results might be only local extrema and not global ones.

The second method is a very human-like approach to solving the problem of enumeration. As the name implies, the process is to enumerate every single possible solution of the problem to assess their quality. This straightforwardness can seem attractive; however, as expected, in the case of complex system analysis, it is simply a widely inefficient method because of the number of useless answers obtained in the process.

The final optimization method, which has attracted the most attention in recent years, is the random method. Within this method, we found several nature-inspired optimization algorithms such as GA [71, 72], particle swarm optimization (PSO) [73], artificial bee colony (ABC) [74, 75], and adaptive firefly algorithm (AFA) [76]. The most popular and widely used would be GA because they can solve multiobjective problems with relatively simple algorithms involving a minimal amount of mathematics. The tradeoff is the birth of deceptive results if the problem is not tackled

wisely and the uncertainty of computational time.

In our case, we adapt the optimization of the mechanical design using exponential coordinates to GA because exponential parameters are flexible, describe the robotic system in an independent manner, and are easy to code into binary strings.

1.3.2 Overview on genetic algorithms

First, we provide a quick reminder on GA. The GA is a type of optimization algorithm based on natural selection, first proposed by John Holland in the 1970s. Similar to the natural process, GAs observe the evolution of a given population at a given time. First, we define the several parts of GAs to clarify their components:

- The total number of samples in a study are referred as the *population* of the study and the size of the population is given by N.
- A *structure* or *genotype* in an artificial genetic system is one member of the population. They represent a parameter set in the solution space and are referred to as S_i.
- A *string* or *chromosome* is the coding of one parameter within the parameter set. In the simplest cases (when parameter sets are only composed of one parameter), the structure and string are the same entity. We refer to the strings as s_i
- These chromosomes are composed of *genes* that are the elementary brick of the parameter coding within the set and are denoted as b_i. These genes contain two pieces of information: their position in the chromosome, which we will call the *locus*, and their value, which we will call *alleles*.
- The function value or *fitness* of a structure or string defines its aptitude to compel to the objective of the optimization, the aim being the maximization of the average fitness of a population, which will mean that we are closer to achieving the objective.

For simplicity, we show an example to explain GAs by representing the population as a string of Boolean characters representing numbers. The set of Boolean is defined as $\mathbb{B} = \{b \,|\, b \in \{0, 1\}\}$. A string is a vector with members in $\mathbb{B}$; for instance,

$$s = \begin{bmatrix} b_1 & \cdots & b_n \end{bmatrix} \in \mathbb{B}^{n_s}. \tag{1.61}$$

However, these strings cannot be used in their Boolean form most of the time because they do not carry the information (or parameter value)

explicitly. To extract the information from those strings, we decode them to obtain a value in the natural number set such that the natural equivalent defined in $\mathbb{N}$ is given by

$$s \equiv b_1 \times 2^0 + \cdots + b_1 \times 2^{n-1}. \tag{1.62}$$

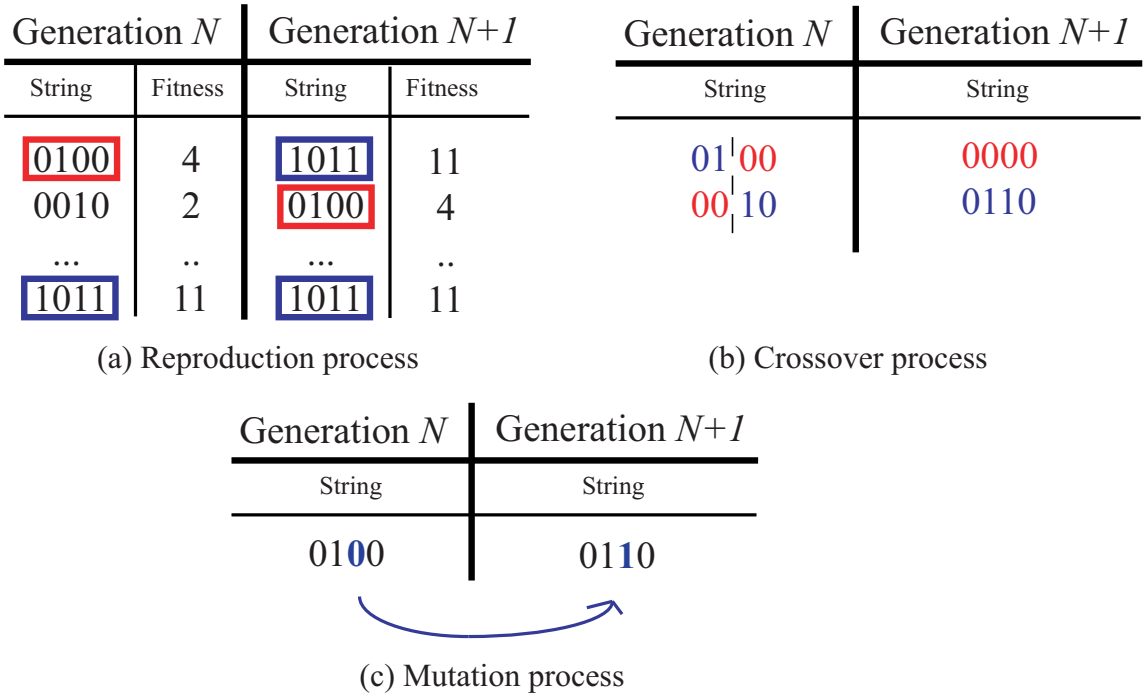
Once the population is defined, we must observe their evolution over time and generate successive populations that will hopefully improve over time. To achieve evolution, the three main operators of genetic algorithms—*Reproduction*, *Crossover* and *Mutation*—are used.

Reproduction is the process of copying existing strings or structures based on their fitness. Since the fitness of these structures are a direct measure of their utility or goodness that we want to maximize, it is only natural to attach higher importance to the structure possessing a high fitness. As such, the reproduction process will first aim to classify each structure in the current population according to their fitness, attributing to every structure a percentage of the total fitness. When this percentage is attributed, we will build a roulette wheel representing the different structures according to their fitness, and we reproduce the structure attached to them where the marble ends up in the wheel. This operator simply represents the concept of the survival of the fittest, because higher the fitness of one structure, the larger is their portion in the wheel, and thus, the higher is the probability that the marble will fall out in their position. We spin the wheel n times to end up with the same number of structures as we had in the beginning.

Crossover is the process of mixing two structures information to form a new one containing part of both parent structure information. The crossover process can be divided into two steps. The first step is to decide where we will "cut" the string or structures to mate them together. The second step is to select two structures at random within the population to ensure that mating occurs. This operator mimics the natural mating process between two subjects of a population.

Finally, mutation is a process that transforms information possessed by one structure at random. Mutation is the operator that is completely random in a genetic algorithm and thus keeps the algorithms from stagnating for too long in a dead end if their initial population is biased or if the population structures are similar but still fail to achieve the objective. The several operations are shown on Fig. 1.5.

The overall process has, however, proven to be a very effective method to achieve optimization, especially when we do not have numerical equations fully describing a system, or when the number of optimization parameters is too high to attempt enumeration optimization methods. Therefore, the main challenge for the design optimization is to code the system parameters in binary for the GA process, as shown in Eq. (1.62). In our exponential formulation, the parameters we need to code into the strings are the exponential parameters $\{v_i, \omega_i, q_i\}$ and the chain matrix $\{C_h\}$. The length of the links l_i is defined from the positions of the joints q_i, and it is not a design parameter.



(a) Reproduction process

(b) Crossover process

(c) Mutation process

Figure 1.5: Basic operations of genetic algorithms

Chapter 2

Kinematics, Dynamics, and control of tree-type systems

2.1 Purpose of this chapter

In this chapter, we will focus on the derivation of the kinematics and dynamics of tree-type systems grasping an object, propose control strategies to apply for those systems and illustrate our theories with a well-known example: an arm-hand system.

We will start by describing the properties of tree-type systems. Those systems possess branching in the kinematic path, which we will describe by separating every path from the base to one extremity of the system in an entity which we will call a chain. We then propose a matrix regrouping the joints in the system and distributing them within the chains of the system. We will call this matrix the chain matrix, and it will describe the connection between the joints to understand the architecture of a tree-type system. Using this matrix, we will then propose a unified closed form of the kinematics and dynamics of those systems based solely on the exponential parameters q, v, ω, their transformations ξ and $e^{\xi\theta}$ and the chain matrix C_h, allowing for an automated design process.

Next, we will focus on the grasping part, deriving the object equation of motion and the constraint linking the manipulator and the object in terms of exponential coordinates. When the whole system equations have been defined, we will observe that such systems often contains redundancy on the manipulator part. There, we will propose control strategies adapted to such situations, summarize the hole automated design process and conclude with an example.

2.2 Motivation

To understand the necessity and implications of the choice of parameters for rigid body motion analysis, we will observe the most simple robotic architecture: the two-link finger seen in Fig. 2.1. In this example, the two joints are rotating around the Z-axis, and the study is limited to the X-Y 2D plane.

In traditional mechanics, we select the joint angles θ_i as a generalized coordinate and represent the link position and velocity in Cartesian coordinates. In the reference frame Σ_s the position of the gravity center of each link is expressed as

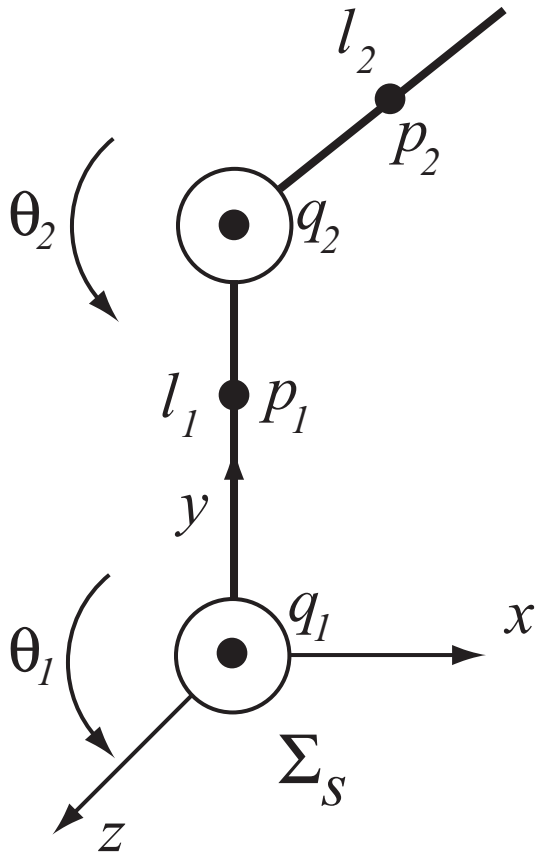


Figure 2.1: Two-link finger system

$$p_1 = \begin{bmatrix} \frac{l_1}{2}\cos\theta_1 \\ \frac{l_1}{2}\sin\theta_1 \\ 0 \end{bmatrix}, \; p_2 = \begin{bmatrix} l_1\cos\theta_1 + \frac{l_2}{2}\cos(\theta_1+\theta_2) \\ l_1\sin\theta_1 + \frac{l_2}{2}\sin(\theta_1+\theta_2) \\ 0 \end{bmatrix}. \tag{2.1}$$

By using a combination of the kinetic energy ($T = \frac{1}{2}mv^2$) and the potential energy ($V = mgh$), we can compute the Lagrangian of the system as

$$\begin{aligned} L &= T - V \\ &= \frac{1}{2}m_1\dot{p}_1^2 + \frac{1}{2}m_2\dot{p}_2^2 + \frac{1}{2}I_1\dot{\theta}_1^2 + \frac{1}{2}I_2(\dot{\theta}_1+\dot{\theta}_2)^2 - g(m_1h_1 + m_2h_2), \end{aligned} \tag{2.2}$$

from there, the equation of motion is obtained by a series of derivation such that

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_i} - \frac{\partial L}{\partial \theta_i} = \Upsilon_i, \tag{2.3}$$

where Υ_i is the external force term.

As is apparent, the main part of the derivation is to describe the link position p_i in Σ_s using the joint angle θ_i and to calculate its derivatives in Eq. (2.3) with respect to θ_i. This can be extended to systems with more

joints in 3D motion; however, the calculation will be more complex and likely to be intractable as the system degrees become larger. This challenge arises from the description of the positions in a Cartesian space such as in Eq. (2.1). Another important aspect to observe is that after computing the Lagrangian, the equation of motion of the system are obtained by operating partial derivation, which is a rather huge drawback for computer driven algorithm which cannot support such computations. While the most basic method of computation is given in the form of Eq. (2.1), it is indeed not the most efficient method to compute dynamics when the systems are becoming more and more complex, which is where new computation schemes have been proposed.

2.3 Rigid body motion using exponential parameters

As seen in the introduction, the rigid body motion in exponential coordinates can be described by exponential coordinates, and they can readily apply to the motion of serial link chains. This section briefly summarize the key aspects of the results used in this chapter.

2.3.1 Exponential parameters

A classic approach to represent a serial link chain is to use parameters that fully describe the joint properties, as the links only act as bridges between these joints. Exponential parameters can be used to realize such an approach. One of the main benefits of using exponential parameters is that they are all defined with respect to a fixed base frame, which allows the modification of the properties of a joint without affecting other joints. Another benefit is the elegant rules of the exponential function pertaining to the products and derivatives, which enables the formulation of the closed-form expression of the kinematics and dynamics of serial link manipulators. Four exponential parameters exist for every joint i ($i = 1, \cdots, n$) in a system:

- The position parameter $q_i \in \mathbb{R}^3$ is the 3D vector representing the coordinates of the initial position of joint i with respect to the base frame Σ_s.
- The translation parameter $v_i \in \mathbb{R}^3$ is the 3D vector representing the translation axis of joint i with respect to Σ_s. For a translational or

Table 2.1: Parameters of exponential coordinates

Joint number	Joint 1	Joint 2
Translation parameter v	$v_1 = \begin{bmatrix}0\\0\\0\end{bmatrix}$	$v_2 = \begin{bmatrix}0\\0\\0\end{bmatrix}$,
Rotation parameter ω	$\omega_1 = \begin{bmatrix}0\\0\\1\end{bmatrix}$	$\omega_2 = \begin{bmatrix}0\\0\\1\end{bmatrix}$,
Joint position q	$q_1 = \begin{bmatrix}0\\0\\0\end{bmatrix}$	$q_2 = \begin{bmatrix}0\\l_1\\0\end{bmatrix}$,

prismatic joint, we set $\|v_i\| = 1$, whereas for a rotational joint, we set $v_i = 0$.

- The rotation parameter $\omega_i \in \mathbb{R}^3$ is the 3D vector representing the rotation axis of joint i with respect to Σ_s. For a rotational joint, we set $\|\omega_i\| = 1$, whereas for a prismatic joint, we set $\omega_i = 0$.
- The movement parameter $\theta_i \in \mathbb{R}$ represents the incremental movement of joint i along the direction v_i or ω_i. This parameter corresponds to the angle and translation for rotational and translational joints, respectively. θ_i is called the joint angle for clarity; however, θ_i can represent a displacement when the joint is translational.

We can locate several joints at the same location to describe a multi-DOF joint by setting $q_i = q_{i+1} = q_{i+2} = \cdots$. For example, exponential parameters for the two-link fingers in Fig. 2.1 is summarized in Table 1.1.

From the exponential parameters, the twist $\widehat{\xi_i}$ (in matrix form) associated with the i_{th} joint is defined as follows:

$$\widehat{\xi_i} = \begin{cases} \begin{bmatrix} \widehat{\omega}_i & -\omega_i \times q_i \\ 0 & 0 \end{bmatrix} & \text{if the joint is pure rotation} \\ \begin{bmatrix} 0 & v_i \\ 0 & 0 \end{bmatrix} & \text{if the joint is pure translation,} \end{cases} \tag{2.4}$$

where $\widehat{\omega}_i$ is the skew-symmetric matrix equivalent to the vector product. The rotation matrix of the i_{th} link relative to the base frame Σ_s is described by the exponential of $\widehat{\omega}_i$ as $R_{si} = e^{\widehat{\omega}_i \theta_i}$. By using $e^{\widehat{\omega}_i \theta_i}$, the exponential of the twist can be described as

$$e^{\widehat{\xi}_i \theta_i} = \begin{cases} \begin{bmatrix} e^{\widehat{\omega}_i \theta_i} & (I - e^{\widehat{\omega}_i \theta_i})(\omega_i \times v_i) + \omega_i \omega_i^{\mathrm{T}} v_i \theta_i \\ 0 & 1 \end{bmatrix} & (\omega \neq 0) \\ \begin{bmatrix} I & v_i \theta_i \\ 0 & 1 \end{bmatrix} & (\omega = 0). \end{cases} \tag{2.5}$$

2.3.2 Kinematics of rigid body

The motion of a rigid body is characterized by a frame attached to it. The configuration $g_{st} \in \mathbb{R}^{4\times4}$ of a tool frame Σ_t relative to a reference frame Σ_s is represented by the relative translation $p_{st} \in \mathbb{R}^3$ and the relative rotation $R_{st} \in \mathbb{R}^{3\times3}$ (orthogonal matrix) as

$$g_{st} = \begin{bmatrix} R_{st} & p_{st} \\ 0 & 1 \end{bmatrix}. \tag{2.6}$$

From (2.5), the exponential of a twist represents a configuration. For serial rink robots, the configuration of the tool frame Σ_t relative to the base frame Σ_s becomes a function of $\theta = [\theta_1, \theta_2, \cdots, \theta_n]^{\mathrm{T}} \in \mathbb{R}^n$. The rigid body motion $g_{st}(\theta)$, or the forward kinematics, is related to the reference configuration $g_{st}(0)$ in exponential parameters as

$$g_{st}(\theta) = e^{\widehat{\xi}_1\theta_1} \dots e^{\widehat{\xi}_n\theta_n} g_{st}(0). \tag{2.7}$$

To calculate the rigid body velocity of a frame from its configuration g, the adjoint transformation $\mathrm{Ad}_g \in \mathbb{R}^{6\times6}$ and its inverse are defined as follows:

$$\mathrm{Ad}_g = \begin{bmatrix} R & \hat{p}R \\ 0 & R \end{bmatrix}, \quad \mathrm{Ad}_g^{-1} = \begin{bmatrix} R^{\mathrm{T}} & -R^{\mathrm{T}}\hat{p} \\ 0 & R^{\mathrm{T}} \end{bmatrix} \tag{2.8}$$

($\widehat{p}$ is the skew-symmetric matrix equivalent to the vector product). We define two types of velocities of the frame Σ_t from the configuration g_{st} as follows. The spatial velocity $V_{st}^s = [(v_{st}^s)^{\mathrm{T}}\ (\omega_{st}^s)^{\mathrm{T}}]^{\mathrm{T}}$, where v_{st}^s and ω_{st}^s respectively denote the translational and rotational components, is the velocity of Σ_t as observed from Σ_s, and it is defined by the (1,1) and (1,2) blocks of $\widehat{V}_{st}^s = \dot{g}_{st}g_{st}^{-1}$. When g_{st} is obtained using (2.7), V_{st}^s can be expressed as

$$V_{st}^s = J_{st}^s\dot{\theta}, \tag{2.9}$$

with

$$J_{st}^s = \begin{bmatrix} \xi_1 & \xi_2' & \dots & \xi_n' \end{bmatrix}, \text{ where } \xi_\ell' = \mathrm{Ad}_{\left(e^{\hat{\xi}_1\theta_1}\dots e^{\hat{\xi}_{\ell-1}\theta_{\ell-1}}\right)}\xi_\ell. \tag{2.10}$$

The body velocity $V_{st}^b = [(v_{st}^b)^{\mathrm{T}}\ (\omega_{st}^b)^{\mathrm{T}}]^{\mathrm{T}}$ represents the velocity of Σ_t as observed from Σ_t, and it is defined by the (1,1) and (1,2) blocks of $\widehat{V}_{st}^b = g_{st}^{-1}\dot{g}_{st}$. When g_{st} is obtained using (2.7), V_{st}^b can be expressed as

$$V_{st}^b = J_{st}^b\dot{\theta}, \tag{2.11}$$

with

$$J_{st}^b = \begin{bmatrix} \xi_1^\dagger & \xi_2^\dagger & \dots & \xi_n^\dagger \end{bmatrix}, \text{ where } \xi_\ell^\dagger = \mathrm{Ad}^{-1}_{\left(e^{\hat{\xi}_\ell \theta_\ell} \dots e^{\hat{\xi}_n \theta_n} g_{st}(0)\right)} \xi_\ell. \tag{2.12}$$

The spatial and body velocities are related to each other by the adjoint transformation as

$$V_{st}^s = \mathrm{Ad}_{g_{st}} V_{st}^b, \quad \text{or} \quad V_{st}^b = \mathrm{Ad}_{g_{st}}^{-1} V_{st}^s. \tag{2.13}$$

When the frames interchange their role, the following relation holds:

$$V_{ab}^b = -V_{ba}^s, \quad V_{ab}^b = -\mathrm{Ad}_{g_{ba}} V_{ba}^b. \tag{2.14}$$

It is also feasible to describe the spatial or body velocity from Σ_a to Σ_c by using an intermediate frame Σ_b:

$$V_{ac}^s = V_{ab}^s + \mathrm{Ad}_{g_{ab}} V_{bc}^s \tag{2.15}$$

$$V_{ac}^b = \mathrm{Ad}_{g_{bc}}^{-1} V_{ab}^b + V_{bc}^b. \tag{2.16}$$

As observed in Eq. (2.7), the forward kinematics are derived only from the exponential parameters (q_i, v_i, ω_i, θ_i) and the initial position $g_{st}(0)$, which is conveniently determined by observing the initial configuration with respect to a single base frame. The initial configuration should be selected appropriately to simplify the definitions of the exponential parameters. In addition, the forward kinematics are composed of the product of the exponentials, which yields a simple formula to express the partial derivatives with respect to the joint angles θ_i. These properties provide a significant benefit to derive a closed-form formula for the dynamics of complex and large-scale systems such as the tree-type systems.

2.4 Tree-type systems and chain matrix

2.4.1 Definition of Tree-type systems

In robotic manipulators such as those in Fig. 2.1, the systems are generally composed of serial chains of links with no closed path, and each link is driven by a joint. We call those types of systems the tree-type manipulators systems, or simply tree-type systems (illustrated in Fig. 2.2), as defined below

Definition 1 *A tree-type system is a mechanical system composed of links and joints characterized by the elements below:*

- The base frame Σ_s is fixed, and the tree-type system starts from the root joint fixed in Σ_s.

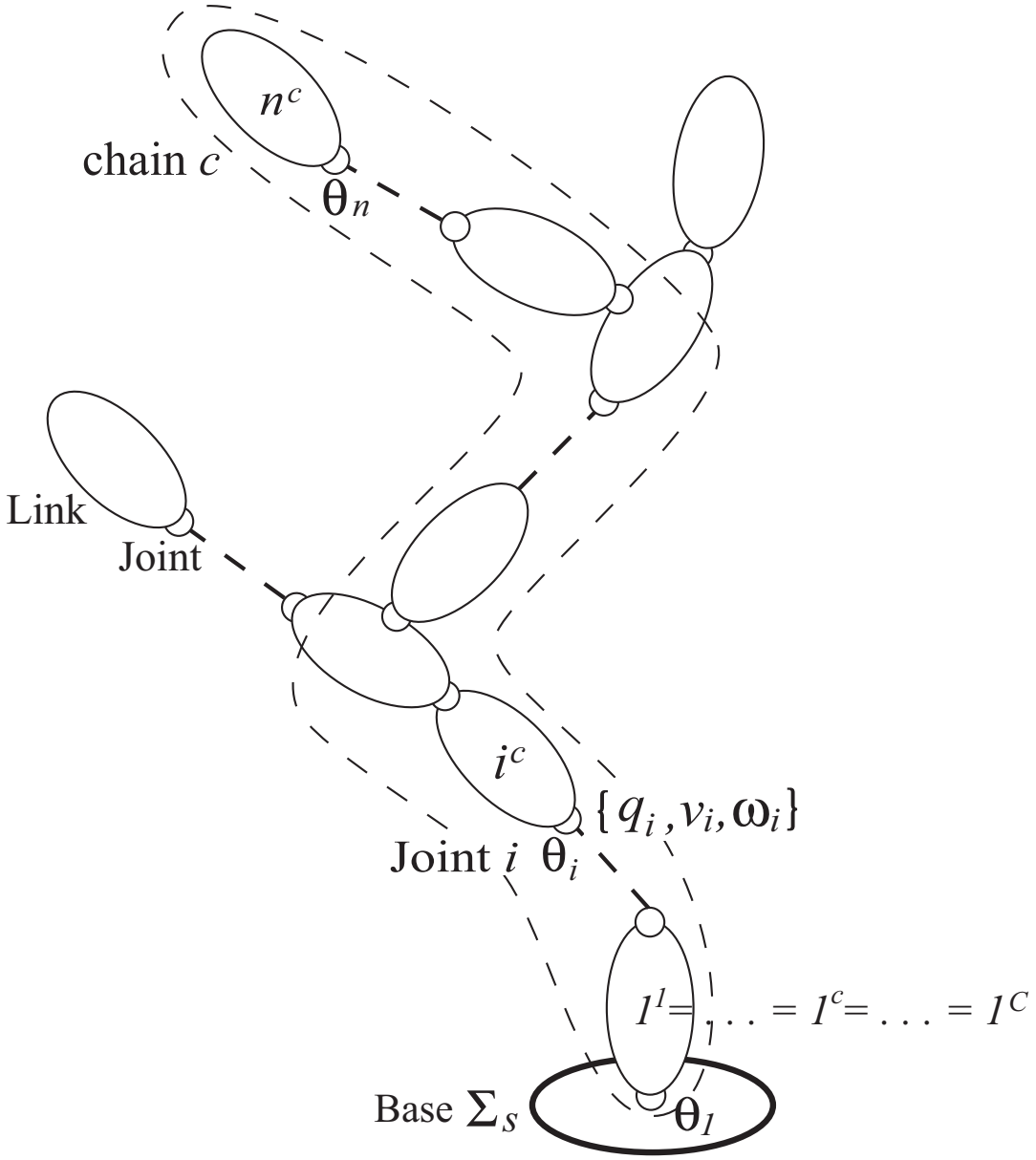


Figure 2.2: Tree-type robotic system architecture

- A joint is attached to each link forming a one-to-one relation. Starting from the base, the joint is connected to its parent (lower-side) link.
- The chain is a set of serial links and joints starting from the root at Σ_s to one of the extremities of the system. The chains within the system are denoted by $c = 1, \cdots, C$.
- The joint are characterized by their constant exponential parameters (q_i,v_i,ω_i) and a variable exponential parameter θ_i representing the joint angle, where $i = 1, \cdots, n$ is the joint number in the complete system. The joint number is assigned to increase from the base in each chain.
- The rigid links are denoted by i^c (a symbol representing the position in the system), where i is the joint number in the complete system and c is the chain to which the joint belongs. A link can be a member of different chains, e.g. similar to the base link ($i = 1$),$1^1 = 1^2 = \cdots = 1^C$.

The above definitions are marginally different from the definitions by Shah. et al. [28] which were used to develop formulas using the Denavit-Hartemberg parameters to define the joints properties. The main difference is that we define the chains to divide the complex tree-type structures into several serial kinematic chains to introduce the chain matrix defined in the next section.

2.4.2 Definition of the kinematic path: the Chain matrix

To describe the system connection, we introduce the connectivity matrices called the chain matrix C_h and the simplified chain matrix $\widehat{C}_h$:

$$C_h = \begin{bmatrix} C_h(1,1) & \cdots & C_h(1,n) \\ \vdots & C_h(c,i) & \vdots \\ C_h(C,1) & \cdots & C_h(C,n) \end{bmatrix} \in \mathbb{R}^{C \times n}, \tag{2.17}$$

where the element $C_h(c,i)$ is set to be one if a joint i exists in a chain c; otherwise, we set $C_h(c,i) = 0$. The ones in the c-th row represent the joints in chain c. Because $e^{0 \cdot \widehat{\xi}_i \theta_i} = I_4$, we can describe the configuration of a link i^c in a unified manner by using C_h as, for example,

$$g_{si^c}(\theta) = \prod_{j=1}^{i} \left(e^{C_h(c,j)\widehat{\xi}_j \theta_j} \right) g_{si^c}(0). \tag{2.18}$$

The ones in the same column in C_h represent the multiplicity of the links in the total chains. To remove this multiplicity, we additionally define the

simplified chain matrix by replacing the ones in a column, except for the topmost to zero:

$$\widehat{C}_h = \begin{bmatrix} \widehat{C}_h(1,1) & \cdots & \widehat{C}_h(1,n) \\ \vdots & \widehat{C}_h(c,i) & \vdots \\ \widehat{C}_h(C,1) & \cdots & \widehat{C}_h(C,n) \end{bmatrix} \in \mathbb{R}^{C \times n}, \tag{2.19}$$

where $\widehat{C}_h(c,i)$ is set to be one if $C_h(c,i)$ is the topmost one in the i_{th} column of C_h; otherwise, we set $\widehat{C}_h(c,i) = 0$.

The simplified chain matrix can be used to calculate the total kinetic and potential energies; for example, the kinetic energy can be defined by

$$T = \frac{1}{2} \sum_{c=1}^{C} \sum_{i=1}^{n} \widehat{C}_h(c,i) \left(J^b_{si^c} \dot{\theta} \right)^{\mathrm{T}} \mathcal{M}_{i^c} J^b_{si^c} \dot{\theta}, \tag{2.20}$$

where $\mathcal{M}_{i^c}$ is the generalized inertia matrix of the link i^c and $V^b_{si^c} = J^b_{si^c} \dot{\theta}$ is its body velocity. Note that it is unnecessary to calculate the terms for $\widehat{C}(c,i) = 0$.

2.5 Dynamical equations

2.5.1 Typical forms of dynamics for manipulation

When we consider deriving the equation of motion for robotic systems interacting with the environment, one of the most prominent examples is a robotic manipulator holding an object with the fingertips. For simplicity, we assume the type of contact to be a point contact with friction without slipping. In the following, we focus on the dynamical equations for those systems for clarity; however, the method can be applied to other types of the tree-type systems (legged robots, humanoid robots, etc.) as well as other mechanical systems (vehicles, tensegrity structures etc.) with marginal modifications.

It is well established that the dynamics of a robotic manipulator holding an object is represented by the following set of equations:

$$M_f \ddot{\theta}_f + C_f \dot{\theta}_f + N_f = \tau - J_h^{\mathrm{T}} \lambda \tag{2.21}$$

$$M_o \dot{V}^b_{so} + C_o V^b_{so} + N_o = G\lambda \tag{2.22}$$

$$J_h \dot{\theta}_f = G^{\mathrm{T}} V^b_{so}, \tag{2.23}$$

where Eq. (2.21) represents the dynamics of the manipulator in terms of the joint angles $\theta_f \in \mathbb{R}^n$, Eq. (2.22) represents the dynamics of the object in terms of the body velocity of the object $V_{so}^b \in \mathbb{R}^6$, and Eq. (2.23) is the geometrical constraint between $\dot{\theta}_f$ and V_{so}^b caused by the contact.

In the equations, $M_f > 0 \in \mathbb{R}^{n\times n}$ and $M_o > 0 \in \mathbb{R}^{6\times 6}$ are the generalized inertia matrices, and $C_f \in \mathbb{R}^{n\times n}$ and $C_o \in \mathbb{R}^{6\times 6}$ are the Coriolis and centrifugal terms matrices. $N_f \in \mathbb{R}^n$ and $N_o \in \mathbb{R}^6$ include the gravity terms, and $\tau \in \mathbb{R}^n$ is the joint torque. Corresponding to the constraint in Eq. (2.23), the Lagrange multipliers $\lambda \in \mathbb{R}^k$ (where k is the number of constraints) are introduced to represent the contact forces, which are mapped to the joint and the object spaces by J_h^{T} and G, respectively. $J_h \in \mathbb{R}^{k\times n}$ and $G \in \mathbb{R}^{6\times k}$ are called the manipulator Jacobian and the grasp map, respectively.

As observed, the system dynamics are determined by $\{M_f, C_f, N_f\}$, $\{M_o, C_o, N_o\}$, and $\{J_h, G\}$. While those equations gives us the general form of the dynamics for every single type of robotic structure, the method to obtain their several components can change quite drastically depending on the basic theories involved in the computations.

2.5.2 Manipulator dynamics for tree-type systems in exponential coordinates

We will start by deriving the mathematical expressions for the manipulator dynamics, in the case of tree-type structure manipulators.

2.5.2.1 Manipulator inertia matrix

The generalized inertia matrix of a link i^c is expressed as

$$\mathcal{M}_{i^c} = \begin{bmatrix} m_{i^c} I_3 & 0 \\ 0 & \mathcal{I}_{i^c} \end{bmatrix}, \tag{2.24}$$

where m_{i^c} and $\mathcal{I}_{i^c}$ are the mass and inertia tensors of the link. From Eq. (2.11), the body velocity $V_{si^c}^b$ of link i^c in chain c relative to the base frame Σ_s can be related to the joint velocity $\dot{\theta}_f$ by the body Jacobian $J_{si^c}^b \in \mathbb{R}^{6\times n}$ as

$$V_{si^c}^b = J_{si^c}^b \dot{\theta}_f. \tag{2.25}$$

In the following, we omit the superscript b in $J_{si^c}^b$ for notational simplicity. Using the body velocity, the kinetic energy of a link is expressed as

$$T_{i^c} = \frac{1}{2} \dot{\theta}_f^{\mathrm{T}} J_{si^c}^{\mathrm{T}} \mathcal{M}_{i^c} J_{si^c} \dot{\theta}_f. \tag{2.26}$$

The total kinetic energy is expressed by summing up these kinetic energies and exhibits the following form

$$T = \frac{1}{2}\dot{\theta}_f^{\mathrm{T}} M_f \dot{\theta}_f, \tag{2.27}$$

where $M_f \in \mathbb{R}^{n\times n}$ is the manipulator inertia matrix appearing in Eq. (2.21) and is expressed by a formula in the next theorem.

Theorem 1 *Consider a tree-type system with the chain matrices $C_h \in \mathbb{R}^{C\times n}$ and $\widehat{C}_h \in \mathbb{R}^{C\times n}$, where C is the total number of chains and n is the total number of joints in the system. Let ξ_i, $\theta_i (= \theta_{f_i})$ be the exponential parameters of joint i and $\mathcal{M}_{i^c}$ be the generalized inertia matrix of the corresponding link i^c. The manipulator inertia matrix $M_f \in \mathbb{R}^{n\times n}$ in Eq. (2.21) is expressed by*

$$M_f = \sum_{c=1}^{C}\sum_{i=1}^{n} \widehat{C}_h(c,i) J_{si^c}^T \mathcal{M}_{i^c} J_{si^c}, \tag{2.28}$$

where $J_{si^c} \in \mathbb{R}^{6\times n}$ is the body Jacobian of link i^c in chain c relative to the base frame Σ_s and is expressed as

$$J_{si^c} = \begin{bmatrix} J_{si^c}(1) & \cdots & J_{si^c}(\ell) & \cdots & J_{si^c}(n) \end{bmatrix}, \tag{2.29}$$

where

$$J_{si^c}(\ell) = \begin{cases} Ad^{-1}_{g^{\dagger}_{\ell i^c}} \xi_\ell & C_h(c,\ell) = 1 \text{ and } \ell \le i \\ 0 & \text{otherwise,} \end{cases} \tag{2.30}$$

where $g^{\dagger}_{\ell i^c}$ is the configuration of link i^c related to link ℓ and is expressed as

$$g^{\dagger}_{\ell i^c} = \prod_{j=l}^{i} (e^{C_h(c,j)\widehat{\xi}_j \theta_j}) g_{si^c}(0). \tag{2.31}$$

$g_{si^c}(0)$ is the configuration of the frame attached to the center of mass of link i^c relative to the base frame Σ_s when the joints are in the reference position $\theta = 0$.

Proof: From the definition of $\widehat{C}_h$, it is apparent that Eq. (2.28) yields the sum of the kinetic energies of all the links. Therefore, the remainder of the proof is to establish that the body Jacobian J_{si^c} of link i^c is expressed by Eqs. (2.29)–(2.30). From Eq. (2.25), $J_{si^c}(\ell)$ relates the joint velocity $\dot{\theta}_\ell$ to the body velocity of link i^c. Therefore, $J_{si^c}(\ell)$ is non-zero only when the joint ℓ is present in chain c and the joint is within the chain from the base

to joint i; this is equivalent to the condition in Eq. (2.30). For the formula of $J_{si^c}(\ell)$ for the non-zero case, note that $J_{si^c}(\ell)$ corresponds to $\xi_i^\dagger$ in Eq. (2.12) with tip link i^c ($B \to i^c$ in Eq. (2.12)). The configuration matrix in the adjoint transformation (in the parentheses in Eq. (2.12)) in this case is expressed by the product of the exponential coordinate matrices of the joints that are present in the chain from ℓ to i in chain c. Because

$$e^{C_h(c,j)\widehat{\xi}_j\theta_j} = \begin{cases} e^{\widehat{\xi}_j\theta_j} & C_h(c,j) = 1 \\ 1 & C_h(c,j) = 0 \end{cases}, \tag{2.32}$$

we observe that this configuration matrix is expressed by Eq. (2.31). □

Note that for calculating M_f in Eq. (2.28), we only need to calculate the body Jacobian J_{si^c} for link i^c for which $\widehat{C}_h(c,i)$ is non-zero.

2.5.2.2 Manipulator Coriolis matrix

Using the expression in Theorem 1, we can also derive a closed form formula for the Coriolis matrix C_f.

Let $M_{\alpha\beta}$ and $C_{\alpha\beta}$ be (α,β) elements of M_f and C_f. From Eq. (2.28), we have

$$M_{\alpha\beta} = \sum_{c=1}^{C}\sum_{i=1}^{n} \widehat{C}_h(c,i) J_{si^c}(\alpha)^{\mathrm{T}} \mathcal{M}_{i^c} J_{si^c}(\beta), \tag{2.33}$$

Using $M_{\alpha\beta}$, $C_{\alpha\beta}$ is expressed by [8]:

$$C_{\alpha\beta} = \frac{1}{2}\sum_{\gamma=1}^{n}\left(\frac{\partial M_{\alpha\beta}}{\partial\theta_\gamma} + \frac{\partial M_{\alpha\gamma}}{\partial\theta_\beta} - \frac{\partial M_{\gamma\beta}}{\partial\theta_\alpha}\right). \tag{2.34}$$

From Eq. (2.33), Eq. (2.28), and Eq. (2.30), for a non-zero $J_{si^c}(\cdot)$, the partial derivatives in Eq. (2.34) are expressed in the form:

$$\begin{aligned}
\frac{\partial M_{\alpha\beta}}{\partial\theta_\gamma} &= \sum_{c=1}^{C}\sum_{i=1}^{n} \widehat{C}_h(c,i)\left(\frac{\partial J_{si^c}^{\mathrm{T}}(\alpha)}{\partial\theta_\gamma}\mathcal{M}_{i^c}J_{si^c}(\beta) + J_{si^c}^{\mathrm{T}}(\alpha)\mathcal{M}_{i^c}\frac{\partial J_{si^c}(\beta)}{\partial\theta_\gamma}\right) \\
&= \sum_{c=1}^{C}\sum_{i=1}^{n} \widehat{C}_h(c,i)\,\xi_\alpha^{\mathrm{T}}\left[\left(\frac{\partial \mathrm{Ad}^{-1}_{g^\dagger_{\alpha i^c}}}{\partial\theta_\gamma}\right)^{\mathrm{T}}\mathcal{M}_{i^c}\mathrm{Ad}^{-1}_{g^\dagger_{\beta i^c}}\right. \\
&\qquad \left. + (\mathrm{Ad}^{-1}_{g^\dagger_{\alpha i^c}})^{\mathrm{T}}\mathcal{M}_{i^c}\left(\frac{\partial \mathrm{Ad}^{-1}_{g^\dagger_{\beta i^c}}}{\partial\theta_\gamma}\right)\right]\xi_\beta.
\end{aligned} \tag{2.35}$$

Therefore, for our purpose, it suffices to have a formula for the partial derivative of the inverse adjoint transformation matrix $\mathrm{Ad}^{-1}_{g^{\dagger}_{\ell i^c}}$ $(\ell = \alpha, \beta, \gamma)$ with respect to θ_k $(k = \alpha, \beta, \gamma)$.

From Eq. (2.6), $g^{\dagger}_{\ell i^c}$ has the following form

$$g^{\dagger}_{\ell i^c} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}, \tag{2.36}$$

and thus,

$$\begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} I & 0 \end{bmatrix} g^{\dagger}_{\ell i^c} \begin{bmatrix} I \\ 0 \end{bmatrix} & \begin{bmatrix} I & 0 \end{bmatrix} g^{\dagger}_{\ell i^c} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ 0 & 1 \end{bmatrix}. \tag{2.37}$$

The corresponding inverse adjoint transformation has the form of Eq. (2.8), and its derivative is expressed by

$$\frac{\partial \mathrm{Ad}^{-1}_{g^{\dagger}_{\ell i^c}}}{\partial \theta_k} = \begin{bmatrix} \left(\dfrac{\partial R}{\partial \theta_k}\right)^{\mathrm{T}} & -\left(\dfrac{\partial R}{\partial \theta_k}\right)^{\mathrm{T}} \widehat{p} - R^{\mathrm{T}} \left(\dfrac{\partial \widehat{p}}{\partial \theta_k}\right) \\ 0 & \left(\dfrac{\partial R}{\partial \theta_k}\right)^{\mathrm{T}} \end{bmatrix}, \tag{2.38}$$

where

$$\left(\frac{\partial R}{\partial \theta_k}\right)^{\mathrm{T}} = \begin{bmatrix} I & 0 \end{bmatrix} \left(\frac{\partial g^{\dagger}_{\ell i^c}}{\partial \theta_k}\right)^{\mathrm{T}} \begin{bmatrix} I \\ 0 \end{bmatrix} \tag{2.39}$$

$$\frac{\partial \widehat{p}}{\partial \theta_k} = \begin{bmatrix} 0 & -p_3 & p_2 \\ p_3 & 0 & -p_1 \\ -p_2 & p_1 & 0 \end{bmatrix}. \tag{2.40}$$

p_1, p_2, p_3 are elements of $\dfrac{\partial p}{\partial \theta_k}$, and are expressed by

$$\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} I & 0 \end{bmatrix} \frac{\partial g^{\dagger}_{\ell i^c}}{\partial \theta_k} \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{2.41}$$

From Eq. (2.31), $\dfrac{\partial g^{\dagger}_{\ell i^c}}{\partial \theta_k}$ in the above equations is conveniently derived as

$$\begin{aligned}\frac{\partial g^{\dagger}_{\ell i^c}}{\partial \theta_k} &= \frac{\partial}{\partial \theta_k}\left\{\prod_{j=\ell}^{i}\left(e^{C_h(c,j)\widehat{\xi}_j\theta_j}\right) g_{si^c}(0)\right\} \\ &= \begin{cases} \displaystyle\prod_{j=\ell}^{k-1}\left(e^{C_h(c,j)\widehat{\xi}_j\theta_j}\right)\ \widehat{\xi}_k \prod_{j=k}^{i}\left(e^{C_h(c,j)\widehat{\xi}_j\theta_j}\right) g_{si^c}(0) & , C_h(c,k)=1 \\ 0 & , C_h(c,k)=0 \end{cases}\end{aligned} \tag{2.42}$$

The result for the Coriolis matrix is summarized in the next theorem.

Theorem 2 *Let $M_{\alpha\beta}$ and $C_{\alpha\beta}$ be the (α,β) elements of the manipulator inertia matrix M_f and the Coriolis matrix C_f, respectively, appearing in Eq. (2.21). Under the same assumption of Theorem 1, the element of the Coriolis matrix $C_{\alpha\beta}$ is expressed by Eq. (2.34), where the partial derivatives, e.g., $\frac{\partial M_{\alpha\beta}}{\partial \theta_\gamma}$, are expressed by Eq. (2.35); here, $\frac{\partial}{\partial \theta_k}\left(Ad^{-1}_{g^{\dagger}_{\ell i^c}}\right)$ (ℓ,$k=\alpha,\beta,\gamma$) is expressed by Eq. (2.38) with Eqs. (2.39)–(2.42).*

2.5.2.3 Manipulator gravity vector

The gravitational force affecting the manipulator is computed from potential energy V by $N_f = \partial V/\partial\theta$. Using the chain matrix, the total potential energy of the manipulator is expressed by

$$V = \sum_{c=1}^{C}\sum_{i=1}^{n} \widehat{C}_h(c,i)\, m_{i^c}\, g\, h_{i^c}, \tag{2.43}$$

where g is the gravitational constant and h_{i^c} is the height of the link center of mass. From Eq. (2.6), the position of a frame is expressed by p_{st} in a configuration g_{st}. Thus, h_{i^c} is extracted from the configuration g_{si^c} of link i^c as

$$h_{i^c} = \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix} g_{si^c} \begin{bmatrix}0\\0\\0\\1\end{bmatrix}. \tag{2.44}$$

Note that g_{si^c} is expressed by the products of the exponential as in Eq. (2.7). Comparing it with $g^{\dagger}_{\ell i^c}$ in Eq. (2.31) with $\ell = 1$, we observe that $g_{si^c} = g^{\dagger}_{1i^c}$. Thus, the partial derivative of g_{si^c} appearing in $N_k = \partial V/\partial\theta_k$ is

expressed by Eq. (2.42) with $\ell = 1$. The result is summarized in the next theorem.

Theorem 3 *Let g be the gravitational constant. Under the same assumption of Theorem 1, the manipulator gravity vector in Eq. (2.21) is expressed by*

$$N_f = [N_{f_1} \ \cdots N_{f_k} \ \cdots \ N_{f_n}]^T, \tag{2.45}$$

where

$$N_{f_k} = \sum_{c=1}^{C} \sum_{i=1}^{n} \widehat{C}_h(c,i)\, m_{i^c}\, g \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \frac{\partial g^{\dagger}_{1i^c}}{\partial \theta_k} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \tag{2.46}$$

where $\partial g^{\dagger}_{1i^c}/\partial\theta_k$ is expressed by Eq. (2.42) with $\ell = 1$.

It is worth noting that we can also consider springs connecting links, joints or the environment by additionally introducing a potential energy of the springs. This term can be calculated separately and we just need to add the corresponding spring force to the original dynamics. This formulation will be useful for the structural analysis (vibration analysis) of structures composed of struts and cables like tensegrity structures.

2.5.3 Object dynamics in exponential coordinates

Now that the equation of motion have been derived for the manipulator part, we derive the object equation of motion using exponential coordinates, to obtain an unified formulation of the full system dynamics.

2.5.3.1 Object motion and orientation

Let p_{so} and r_{so} be the mass center position and rotation vector of the object. Vector r_{so} is defined to align with the rotation axis of the object and to have the magnitude of the rotation angle about it. The body velocity V^b_{so} in Eq. (2.22) is composed of the translational velocity v^b_{so} and the angular velocity ω^b_{so}, and they are calculated from the translational speed $\dot{p}_{so}$ and the rotation matrix R_{so} [8]:

$$V^b_{so} = \begin{bmatrix} v^b_{so} \\ \omega^b_{so} \end{bmatrix} = \begin{bmatrix} R^{\mathrm{T}}_{so}\dot{p}_{so} \\ (R^{\mathrm{T}}_{so}\dot{R}_{so})^{\mathrm{V}} \end{bmatrix}. \tag{2.47}$$

The superscript $(.)^{\mathrm{V}}$ denotes the operator which converts a skew-symmetric matrix into its vector counterpart (i.e., $\widehat{\omega}_i$ in Eq. (1.6) is transformed back

to ω_i by $\omega_i = (\widehat{\omega}_i)^{\mathrm{V}}$).

As in Eq. (1.37), the rotation matrix R_{so} is expressed by the exponential of $\widehat{r}_{so}$. Applying Taylor's development and noting $\widehat{r}_{so}^3 = -\theta_{so}^2 \widehat{r}_{so}$ with θ_{so} being the object rotation angle, we obtain

$$\begin{aligned} R_{so} = e^{\widehat{r}_{so}} &= I + \widehat{r}_{so} + \frac{1}{2!}\widehat{r}_{so}^2 + \frac{1}{3!}\widehat{r}_{so}^3 + \cdots \\ &= I + \frac{\sin(\|r_{so}\|)}{\|r_{so}\|}\widehat{r}_{so} + \frac{1-\cos(\|r_{so}\|)}{\|r_{so}\|^2}\widehat{r}_{so}^2. \end{aligned} \tag{2.48}$$

In Eq. (2.47), v_{so}^b is calculated from $\dot{p}_{so}$ and R_{so} in Eq. (2.48), whereas ω_{so}^b is calculated from R_{so} in Eq. (2.48) and its derivative. After some calculation using the properties of skew-symmetric matrices [77], ω_{so}^b is expressed by

$$\omega_{so}^b = T_{so}\dot{r}_{so}, \tag{2.49}$$

where

$$T_{so} = I - \frac{1-\cos(\|r_{so}\|)}{\|r_{so}\|^2}\widehat{r}_{so} + \frac{\|r_{so}\| - \sin(\|r_{so}\|)}{\|r_{so}\|^3}\widehat{r}_{so}^2. \tag{2.50}$$

2.5.3.2 Object dynamics matrices

The object equation in Eq. (2.22) is the Newton–Eular equation in terms of the body velocity of the object V_{so}^b. Using R_{so} and ω_{so}^b expressed by Eq. (2.48) and Eq. (2.49), respectively, and from the standard derivation process of the Newton–Eular equation, the object inertia and Coriolis matrices and gravity vector are expressed by

$$M_o = \begin{bmatrix} m_o I_3 & 0 \\ 0 & \mathcal{I}_o \end{bmatrix} \tag{2.51}$$

$$C_o = \begin{bmatrix} m_o \widehat{\omega}_{so}^b & 0 \\ 0 & \frac{1}{2}(\widehat{\omega}_{so}^b \mathcal{I}_o - \mathcal{I}_o \widehat{\omega}_{so}^b) \end{bmatrix} \tag{2.52}$$

$$N_o = \begin{bmatrix} R_{so}^{\mathrm{T}} m_o g \\ 0 \end{bmatrix}, \tag{2.53}$$

where m_o and $\mathcal{I}_o$ are the mass and inertia tensor of the object and $\widehat{\omega}_{so}^b$ is the skew-symmetric counterpart of ω_{so}^b.

2.5.4 Constraint equation

Because we consider the point contact with friction without slipping, the constraint can be represented by the velocity of the fingertip relative to the

object surface. We define Σ_{f_i} and Σ_{c_i} as frames fixed at the contact points on the manipulator and the object surface, respectively. Note that the subscript i corresponds to the chain number in this section. The constraint equation at the i_{th} contact point can be expressed as

$$B_{c_i}^{\mathrm{T}} V_{f_i c_i}^b = 0 \tag{2.54}$$

where $V_{f_i c_i}^b$ is the body velocity of Σ_{c_i} as observed from Σ_{f_i} and $B_{c_i} \in \mathbb{R}^{6\times p}$ is the wrench basis matrix composed of elements zero and one constraining the velocity in the selected direction. The dimension of p indicates the total number of constraints caused by the contact. In the case of a point contact with friction without slipping, $p = 3$.

From Eq. (2.16) with $(a = f_i, b = s, c = c_i)$ and $(a = f_i, b = s, c = f_i)$, we can divide the body velocity as

$$\begin{aligned} V_{f_i c_i}^b &= \mathrm{Ad}_{g_{sc_i}}^{-1} V_{f_i s}^b + V_{sc_i}^b \\ &= -\mathrm{Ad}_{g_{sc_i}}^{-1} \mathrm{Ad}_{g_{sf_i}} V_{sf_i}^b + V_{sc_i}^b. \end{aligned} \tag{2.55}$$

Both the velocities in Eq. (2.55) are observed from the base Σ_s and are derived from Eq. (2.8) and Eq. (2.11) as

$$V_{sf_i}^b = \mathrm{Ad}_{g_{sf_i}}^{-1} V_{sf_i}^s = \mathrm{Ad}_{g_{sf_i}}^{-1} J_{sf_i}^s \dot{\theta}_f, \tag{2.56}$$

and

$$V_{sc_i}^b = \mathrm{Ad}_{g_{oc_i}}^{-1} V_{so}^b + V_{oc_i}^b = \mathrm{Ad}_{g_{oc_i}}^{-1} V_{so}^b, \tag{2.57}$$

from Eq. (2.16) with $(a = s, b = o, c = c_i)$ and $V_{oc_i}^b = 0$ because Σ_{c_i} is fixed on the object. Substituting Eqs. (2.55)–(2.57) into Eq. (2.54), we obtain

$$J_{h_i} \dot{\theta}_f = G_i^{\mathrm{T}} V_{so}^b, \tag{2.58}$$

where

$$J_{h_i} = B_{c_i}^{\mathrm{T}} \mathrm{Ad}_{g_{sc_i}}^{-1} J_{sf_i}^s \tag{2.59}$$

$$G_i = \mathrm{Ad}_{g_{oc_i}}^{-T} B_{c_i}. \tag{2.60}$$

In Eq. (2.60), the configuration g_{oc_i} is constant and is derived from the position p_{oc_i} and the rotation R_{oc_i} of Σ_{c_i} relative to Σ_o as

$$g_{oc_i} = \begin{bmatrix} R_{oc_i} & p_{oc_i} \\ 0 & 1 \end{bmatrix}. \tag{2.61}$$

The configuration g_{sc_i} in Eq. (2.59) can be decomposed as

$$g_{sc_i} = g_{so} g_{oc_i}, \tag{2.62}$$

and g_{so}, the configuration of the object Σ_o relative to Σ_s, is derived from the object position p_{so} and rotation R_{so} in Eq. (2.48) as

$$g_{so} = \begin{bmatrix} R_{so} & p_{so} \\ 0 & 1 \end{bmatrix}. \tag{2.63}$$

The corresponding inverse adjoint matrices in Eq. (2.59) and Eq. (2.60) are expressed by Eq. (2.8) with Eqs. (2.61)–(2.63). Finally, $J^s_{sf_i}$ in Eq. (2.59) is the spatial Jacobian from the base to the tip of the chain corresponding to the contact. Observing Eq. (2.10) and using the chain matrix, we observe that $J^s_{sf_i}$ is expressed by

$$J^s_{sf_i} = \begin{bmatrix} J^s_{sf_i}(1) & \cdots & J^s_{sf_i}(\ell) & \cdots & J^s_{sf_i}(n) \end{bmatrix} \in \mathbb{R}^{6\times n} \tag{2.64}$$

where $J^s_{sf_i}(1) = \xi_1$, and for $\ell > 2$,

$$J^s_{sf_i}(\ell) = \begin{cases} \mathrm{Ad}_{g'_{s,\ell-1}}\xi_\ell & C_h(i,l) = 1 \\ 0 & C_h(i,l) = 0 \end{cases} \tag{2.65}$$

where

$$g'_{s,\ell-1} = \prod_{j=1}^{\ell-1} e^{C_h(i,j)\widehat{\xi}_j\theta_j}. \tag{2.66}$$

The total constraint equation (2.23) is conveniently obtained by stacking up Eq. (2.58) over the contacts. The result is summarized in the next theorem.

Theorem 4 *Let the constraint between the i_{th} finger and the object be expressed by Eq. (2.54). Let C be the number of contacts (chains) and k be the total number of constraints. Under the same assumption as in Theorem 1, the manipulator Jacobian $J_h \in \mathbb{R}^{k\times n}$ in Eq. (2.23) is expressed by*

$$J_h = \begin{bmatrix} J^{\mathrm{T}}_{h_1} & \cdots & J^{\mathrm{T}}_{h_i} & \cdots & J^{\mathrm{T}}_{h_C} \end{bmatrix}^{\mathrm{T}}, \tag{2.67}$$

where J_{h_i} is expressed by Eq. (2.59). The inverse adjoint transformation matrix $Ad^{-1}_{g_{sc_i}}$ is defined by Eq. (2.8) with Eq. (2.6), and the corresponding configuration matrix g_{sc_i} is expressed by Eq. (2.62) with Eq. (2.61) and Eq. (2.63). The spatial Jacobian $J^s_{sf_i}$ is expressed by Eq. (2.64) with Eq. (2.65) and Eq. (2.66).

The grasp map $G \in \mathbb{R}^{6\times k}$ is expressed by

$$G = \begin{bmatrix} G_1 & \cdots & G_i & \cdots & G_C \end{bmatrix}, \tag{2.68}$$

where G_i is expressed by Eq. (2.60). The inverse adjoint transformation matrix $Ad^{-1}_{g_{oc_i}}$ is calculated from the configuration g_{oc_i} in Eq. (2.61) using Eq. (2.8) with Eq. (2.6).

2.6 Selection of control variables

2.6.1 Augmented constraint equation and redundant motion

For regulating the object motion, $x_o = \begin{bmatrix} p_{so}^{\mathrm{T}} & r_{so}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ can be selected as control variables. From Eq. (2.47) and Eq. (2.49), $\dot{x}_o$ is related to V_{so}^b by

$$V_{so}^b = T_o \dot{x}_o, \tag{2.69}$$

where

$$T_o = \begin{bmatrix} R_{so}^{\mathrm{T}} & 0 \\ 0 & T_{so} \end{bmatrix}, \tag{2.70}$$

where R_{so} and T_{so} are given by Eq. (2.48) and Eq. (2.50).

In the tree-type systems, the number of joints n is generally higher than the number of required constraints k, and we can specify a redundant motion $v_r \in \mathbb{R}^{n-k}$ in addition to the object motion $\dot{x}_o \in \mathbb{R}^6$ because $J_h \in \mathbb{R}^{k \times n}$ in Eq. (2.23) is a fat matrix in this case. We can choose variables as v_r as far as the coefficient matrix of the augmented constraint equation

$$\underbrace{\begin{bmatrix} J_h \\ T_h \end{bmatrix}}_{\overline{J}_h} \dot{\theta}_f = \begin{bmatrix} G^{\mathrm{T}} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} V_{so}^b \\ v_r \end{bmatrix}, \tag{2.71}$$

that is,

$$\overline{J}_h = \begin{bmatrix} J_h \\ T_h \end{bmatrix} \tag{2.72}$$

is non-singular.

A typical selection of T_h is $T_h = K_h^{\mathrm{T}}$, where K_h is the null-space matrix of J_h (i.e., $J_h K_h = 0$). However, the corresponding velocity $v_n = K_h^{\mathrm{T}} \dot{\theta}_f$ called the internal velocity does not have intuitive physical significance and is not effective for control purpose. The only requirement for the selection of the redundant velocity $v_r = T_h \dot{\theta}$ is that $\overline{J}_h$ is non-singular, and it is not limited to $T_h = K_h^{\mathrm{T}}$. A general requirement for T_h is specified in the next theorem.

Theorem 5 *Consider the constraint equation of the tree-type system in Eq. (2.23). Suppose $J_h \in \mathbb{R}^{k \times n}$ is the row-full rank and $k < n$. Then, the singular value decomposition of J_h has the form*

$$J_h = U_1 \begin{bmatrix} \Sigma_r & 0 \end{bmatrix} \begin{bmatrix} V_1^{\mathrm{T}} \\ V_2^{\mathrm{T}} \end{bmatrix}. \tag{2.73}$$

Suppose we wish to select the redundant velocity v_r by

$$v_r = T_h \dot{\theta}_f. \tag{2.74}$$

The augmented coefficient matrix $\overline{J}_h$ defined by Eq. (2.72) is non-singular if and only if

$$\det(\mathrm{T}_h \mathrm{V}_2) \neq 0 \tag{2.75}$$

Proof: Necessity) Suppose $\overline{J}_h$ is non-singular. From Eq. (2.73), $J_h = U_1 \Sigma_r V_1^{\mathrm{T}}$ and $K_h = V_2$ is a null-space matrix. Note that

$$\widehat{J}_h = \begin{bmatrix} J_h \\ K_h^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} U_1 \Sigma_r V_1^{\mathrm{T}} \\ V_2^{\mathrm{T}} \end{bmatrix} \tag{2.76}$$

is non-singular, and

$$\widehat{J}_h^{-1} = \begin{bmatrix} V_1 \Sigma_r^{-1} U_1^{\mathrm{T}} & V_2 \end{bmatrix}. \tag{2.77}$$

From Eq. (2.72) and Eq. (2.76), there exists matrices X_J and X_k such that

$$\overline{J}_h = \begin{bmatrix} J_h \\ T_h \end{bmatrix} = \begin{bmatrix} I & 0 \\ X_J & X_k \end{bmatrix} \widehat{J}_h. \tag{2.78}$$

From Eq. (2.78) and the non-singular condition, we have

$$\det(\overline{J}_h) = \det(I)\det(X_k)\det(\widehat{J}_h) \neq 0, \tag{2.79}$$

and thus, $\det(X_k) \neq 0$. Meanwhile, from Eq. (2.78) and Eq. (2.77), we have

$$\begin{bmatrix} X_J & X_k \end{bmatrix} = T_h \widehat{J}_h^{-1} = \begin{bmatrix} T_h V_1 \Sigma_r U_1^{\mathrm{T}} & T_h V_2 \end{bmatrix} \tag{2.80}$$

Combining $\det(X_k) \neq 0$ and Eq. (2.80), we have $\det(T_h V_2) \neq 0$.

Sufficiency) Suppose $\det(T_h V_2) \neq 0$. $\det(\overline{J}_h) \neq 0$ is apparent from Eq. (2.79) and Eq. (2.80). □

In the above theorem, we can select the redundant variables v_r to have a more explicit physical significance than the internal velocity v_n by selecting them from the combinations of $\dot{\theta}_f$ specified by T_h as in Eq. (2.74).

From Eq. (2.78), Eq. (2.71) with $T_h = K_h^{\mathrm{T}}$ and $v_r = v_n$, and Eq. (2.80),

$$\begin{aligned} v_r = T_h \dot{\theta}_f &= \begin{bmatrix} X_J & X_k \end{bmatrix} \widehat{J}_h \dot{\theta}_f = \begin{bmatrix} X_J & X_k \end{bmatrix} \begin{bmatrix} G^{\mathrm{T}} V_{so}^b \\ v_n \end{bmatrix} \\ &= (T_h V_1 \Sigma_r U_1^{\mathrm{T}}) G^{\mathrm{T}} V_{so}^b + (T_h V_2) v_n. \end{aligned} \tag{2.81}$$

Therefore, the non-singular condition in Eq. (2.75) can be considered as a requirement to select v_r to span the space spanned by v_n. When we

encounter the case $\det(T_h V_2) = 0$, the selection of v_r is not appropriate, and we should re-select or modify v_r to reflect the internal motion v_n. A feasible method of the modification is described in the next corollary.

Corollary 1 *Consider the same assumptions as in Theorem 5. Suppose we wish to select the redundant velocity v_r as in Eq. (2.74); however, $det(T_h V_2) = 0$. For arbitrary $\varepsilon > 0$, we can select a modified redundant variable $\widehat{v}_r$ by*

$$\widehat{v}_r = \widehat{T}_h \dot{\theta}_f, \tag{2.82}$$

which satisfies the norm condition

$$\frac{\|v_r - \widehat{v}_r\|}{\|\dot{\theta}_f\|} \leq \varepsilon. \tag{2.83}$$

One such option for $\widehat{T}_h$ is expressed by

$$\widehat{T}_h = T_h + \varepsilon Y_2 Z_2^{\mathrm{T}} V_2^{\mathrm{T}}, \tag{2.84}$$

where Y_2, Z_2 are components of the unitary matrices in the SVD of $T_h V_2$, i.e.,

$$T_h V_2 = \begin{bmatrix} Y_1 & Y_2 \end{bmatrix} \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} Z_1^{\mathrm{T}} \\ Z_2^{\mathrm{T}} \end{bmatrix}. \tag{2.85}$$

Proof: Because $\det(T_h V_2) = 0$, the SVD has the form of Eq. (2.85). From Eq. (2.84) and Eq. (2.85), we observe that

$$\begin{aligned} \widehat{T}_h V_2 &= (T_h + \varepsilon Y_2 Z_2^{\mathrm{T}} V_2^{\mathrm{T}}) V_2 \\ &= T_h V_2 + \varepsilon Y_2 Z_2^{\mathrm{T}} = \begin{bmatrix} Y_1 & Y_2 \end{bmatrix} \begin{bmatrix} \Sigma_r & 0 \\ 0 & \varepsilon I \end{bmatrix} \begin{bmatrix} Z_1^{\mathrm{T}} \\ Z_2^{\mathrm{T}} \end{bmatrix} \end{aligned} \tag{2.86}$$

is non-singular. Therefore, the condition in Eq. (2.75) is satisfied, and we can select $\widehat{v}_r$ in Eq. (2.82) with $\widehat{T}_h$ in Eq. (2.84). The norm condition is satisfied because

$$\begin{aligned} \frac{\|v_r - \widehat{v}_r\|}{\|\dot{\theta}_f\|} &= \frac{\|(T_h - \widehat{T}_h)\dot{\theta}_f\|}{\|\dot{\theta}_f\|} \leq \overline{\sigma}(T_h - \widehat{T}_h) \\ &= \sqrt{\overline{\lambda}\Big((\varepsilon Y_2 Z_2^{\mathrm{T}} V_2^{\mathrm{T}})^{\mathrm{T}} (\varepsilon Y_2 Z_2^{\mathrm{T}} V_2^{\mathrm{T}})\Big)} \\ &= \sqrt{\overline{\lambda}(\varepsilon^2 I)} = \varepsilon, \end{aligned} \tag{2.87}$$

where $\overline{\sigma}(\cdot)$ and $\overline{\lambda}(\cdot)$ represent the maximum singular value and the maximum eigen value, respectively. □

2.6.2 Manipulating force and internal force

For control purposes, it is convenient to decompose the contact force λ into the manipulating force F_o (the total force applied to the object) and the magnitude of the internal force f_N (typically, a set of a couple of forces acting along the line connecting the contact points [22]). From Eq. (2.22), we have

$$F_o = G\lambda, \tag{2.88}$$

and the decomposition is obtained by solving the above equation:

$$\lambda = G^+ F_o + K_G f_N, \tag{2.89}$$

where $G^+ \in \mathbb{R}^{k\times 6}$ is the pseudo-inverse of G and $K_G \in \mathbb{R}^{k\times(n-6)}$ represents the matrix whose columns span the null space of G ($GK_G = 0$).

To maintain the constraint in Eq. (2.23), λ should be regulated so that the fingertip force F_f lies in the friction cone $\mathcal{FC}$, i.e.,

$$F_f \in \mathcal{FC}. \tag{2.90}$$

Several contact model exists (Frictionless point contact, soft finger..). When the constraint comes from the point contact with friction without slipping, the corresponding contact force is equivalent to the fingertip force, i.e. $\lambda = F_f$.

2.7 Control design

2.7.1 Control objectives and assumptions

For grasping and manipulation, we make the following assumptions:
(A1) $\overline{J}_h \in \mathbb{R}^{n\times n}$ in Eq. (2.72) is nonsingular.
(A2) For any F_o, there exists f_N to make F_f satisfy Eq. (2.90).
These assumptions ensure that the grasping is manipulable and force closure [8]. These conditions can be satisfied by appropriately selecting the contact points.

For regulating the system motion, we select the object motion $x_o = \left[p_{so}^{\mathrm{T}} \;\; r_{so}^{\mathrm{T}}\right]^{\mathrm{T}}$ and the redundant motion $x_r = \int v_r dt$ as the control variables and let

$$\overline{x}_o = \left[x_o^{\mathrm{T}} \;\; x_r^{\mathrm{T}}\right]^{\mathrm{T}}. \tag{2.91}$$

To regulate the contact force to satisfy Eq. (2.90), we select the internal force f_N as a control variable. The control objectives are

(C1) For the system motion, control $\overline{x}_o$ to follow its desired trajectories $\overline{x}_{o_d}$.
(C2) For the contact force, control f_N to follow its desired trajectory f_{N_d}.

2.7.2 Linear compensator (computed torque control)

The control objectives can be accomplished by numerous control strategies. A simple strategy for robot control is the computed torque control which is described in the following.

We first re-express the constraint in terms of $\overline{x}_o$ in Eq. (2.91). Using $\overline{x}_o$, Eq. (2.69) is re-expressed as

$$V_{so}^b = \widehat{T}_o \dot{\overline{x}}_o, \tag{2.92}$$

where $\widehat{T}_o = \begin{bmatrix} T_o & 0 \end{bmatrix}$. Similarly, from Eq. (2.69), the constraint equation in Eq. (2.71) is re-expressed as

$$\overline{J}_h \dot{\theta}_f = \overline{T}_o \dot{\overline{x}}_o, \tag{2.93}$$

where $\overline{T}_o = \text{block diag}(G^{\mathrm{T}} T_o, I)$. Considering the time derivative in Eq. (2.92) and Eq. (2.93), we have

$$\dot{V}_{so}^b = \widehat{T}_o \ddot{\overline{x}}_o + \dot{\widehat{T}}_o \dot{\overline{x}}_o, \tag{2.94}$$

$$\overline{J}_h \ddot{\theta}_f = \overline{T}_o \ddot{\overline{x}}_o + \dot{\overline{T}}_o \ddot{\overline{x}}_o - \dot{\overline{J}}_h \dot{\theta}_f. \tag{2.95}$$

Because $F_o = G\lambda$, substitution of Eq. (2.22) into Eq. (2.89) yields

$$\lambda = G^+ (M_o \dot{V}_{so}^b + C_o V_{so}^b + N_o) + K_G f_N. \tag{2.96}$$

The total system dynamics is obtained by substituting Eq. (2.96) into Eq. (2.21). By eliminating $\ddot{\theta}_f$, $\dot{\theta}_f$, $\dot{V}_{so}^b$, and V_{so}^b from the equation by using Eq. (2.92)–(2.95), we have

$$\overline{M}_o \ddot{\overline{x}}_o + \overline{C}_o \dot{\overline{x}}_o + \overline{N}_o + J_h^{\mathrm{T}} K_G f_N = \tau, \tag{2.97}$$

where

$$\overline{M}_o = M_f \overline{J}_h^{-1} \overline{T}_o + J_h^{\mathrm{T}} G^+ M_o \widehat{T}_o. \tag{2.98}$$

$$\overline{C}_o = M_f \overline{J}_h^{-1} (\dot{\overline{T}}_o - \dot{\overline{J}}_h \overline{J}_h^{-1} \overline{T}_o) + C_f \overline{J}_h^{-1} \overline{T}_o + J_h^{\mathrm{T}} G^+ (M_o \dot{\widehat{T}}_o + C_o \widehat{T}_o). \tag{2.99}$$

$$\overline{N}_o = N_f + J_h^{\mathrm{T}} G^+ N_o. \tag{2.100}$$

By selecting the control input

$$\tau = \overline{M}_o u_o + \overline{C}_o \dot{\overline{x}}_o + \overline{N}_o + J_h^{\mathrm{T}} K_G u_{f_N}, \tag{2.101}$$

and substituting Eq. (2.101) into Eq. (2.97), we observe that the closed loop system becomes

$$\begin{bmatrix} \overline{M}_o & J_h^{\mathrm{T}} K_G \end{bmatrix} \begin{bmatrix} \ddot{\overline{x}}_o - u_o \\ f_N - u_{f_N} \end{bmatrix} = 0. \tag{2.102}$$

Therefore, as long as the coefficient matrix $\begin{bmatrix} \overline{M}_o & J_h^{\mathrm{T}} K_G \end{bmatrix}$ is non-singular, we can directly regulate $\ddot{\overline{x}}_o$ and f_N by u_o and u_{f_N}. A simple option of u_o and u_{f_N} to achieve the control objectives (C1) and (C2) is PD and PI control such as

$$u_o = \ddot{\overline{x}}_{o_d} - K_{d_o}(\dot{\overline{x}}_o - \dot{\overline{x}}_{o_d}) - K_{p_o}(\overline{x}_o - \overline{x}_{o_d}), \tag{2.103}$$

$$u_{f_N} = f_{N_d} - K_{i_f} \int (f_N - f_{N_d}) dt, \tag{2.104}$$

where K_{d_o}, K_{p_o}, $K_{i_f} > 0$ are positive definite matrices. Closed loop stability follows from the standard PID control theory [8].

2.8 Summary of the modeling and control process

To sum up all the steps to derive the dynamics and the controller of a tree-type manipulator, this is the proposed procedure to follow:

(i) Define the systems configuration by specifying the joint parameters: (q_i, v_i, ω_i) as in Table 2.1, rigid links parameters (m_{i^c}, l_{i^c}, $\mathcal{I}_{i^c}$), object parameters (m_o, l_o, $\mathcal{I}_o$), architecture of the system (C_h,$\widehat{C}_h$) in Eq. (2.17), contact type B_{c_i} in Eq. (2.54), and configurations (the position of the contacts g_{oc_i} in Eq. (2.61), and the initial link position $g_{si^c}(0)$ in Eq. (2.31)).

(ii) Derive the manipulator and the object dynamics by determining the terms (M_f, C_f, N_f) and (M_o, C_o, N_o) as well as the terms of the constraint (J_h, G) by following the procedure in section 2.5.

(iii) Select the redundant variable v_r by selecting T_h in Eq. (2.74), and construct the control input τ in Eq. (2.101).

A flow chart to construct a simulator is shown in Fig. 2.3. The labels (i)-(iii) indicate the steps above from which the parameters and variables in the boxes are obtained.

Note that the only part which must be modified to change the system configuration is the first part, whereas the following parts can be automatized. Owing to the simplicity of the system description resulting from the

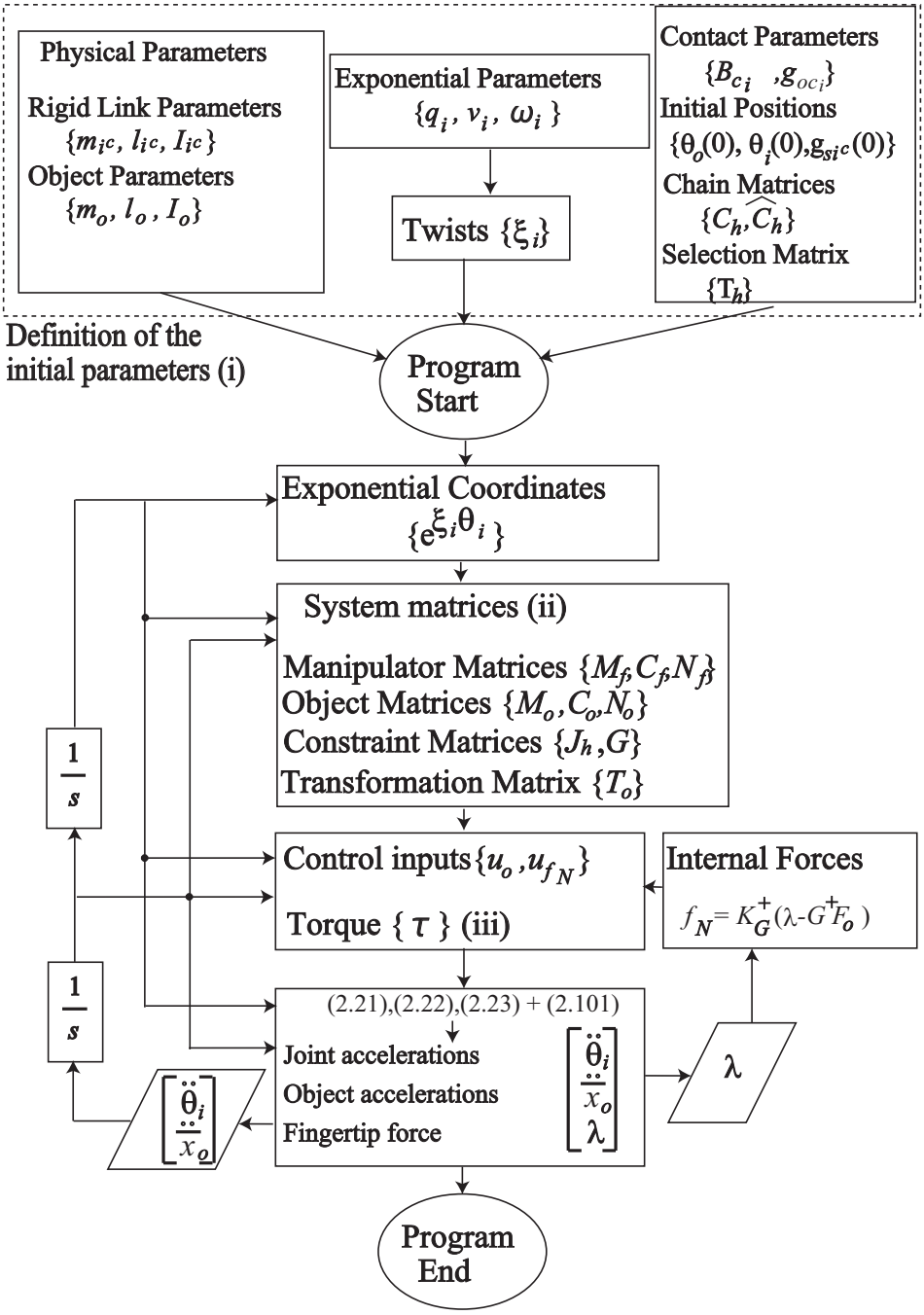


Figure 2.3: Simulation flowchart for modeling and control process

use of the exponential coordinates and the chain matrix, it will be convenient to simulate various types of complex tree-type architecture manipulators in a short amount of time. Together with the closed form expression of the system, this algorithm will be effective for the optimization process.

2.9 Numerical Simulations

2.9.1 Comparison between dynamics from proposed and conventional methods

2.9.1.1 Simulation settings

Before beginning to construct a complex structure using the proposed theories, we shall compare the results yielded by the dynamics obtained from the classical method (Cartesian coordinate representation) and the proposed method (exponential coordinate representation). To compare both the results, we will study a simple example of two two-link fingers ($n = 4$) grasping an object in 2D plane (Y–Z plane) as shown in Fig. 2.4. The contact between the fingers and the object is assumed to be a point contact with friction.

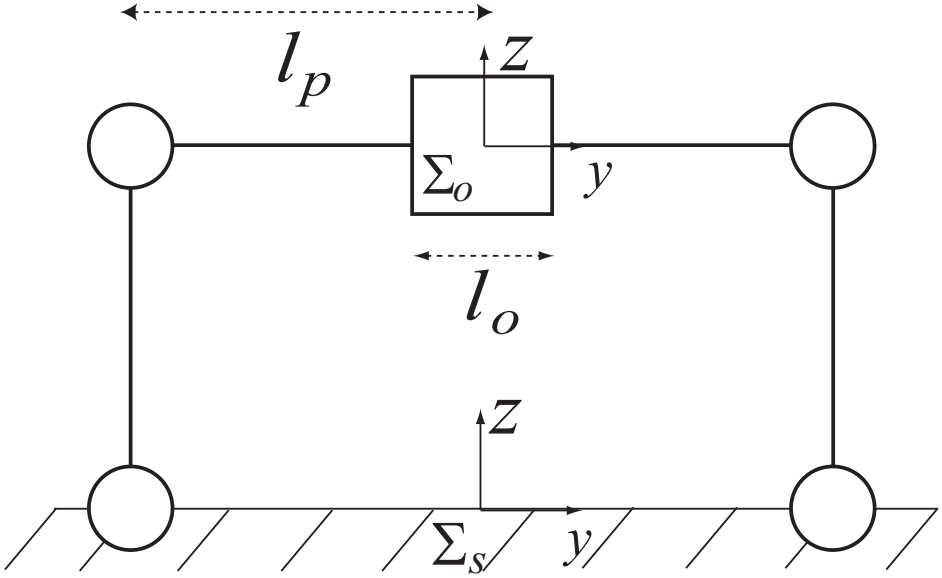


Figure 2.4: Two-link fingers for planar grasp (initial position)

We define the system base frame Σ_s at the middle of the base of the two fingers (the center of the "palm" of the hand) and the object frame Σ_o at the center of mass of the grasped object. For simplicity in this example, the object is a cube of length $l_o = 0.3$ [m] and mass $m_o = 0.1$ [kg]. All the links of the fingers are of identical length $l_f = 1$ [m] and mass $m_f = 0.1$ [kg], and the length of the palm of the hand is $l_p = 1.15$ [m]. In the following, the lengths will be in meter and the angles in radian. The reference configuration ($\theta = 0$) is selected at the configuration when all the finger links are aligned with the positive y axis of Σ_s.

The initial configuration is selected as Fig. 2.4, and the initial values of the joint angles and the object position are expressed as

$$\theta_{f_i} = \left[\begin{array}{cc|cc} \frac{\pi}{2} & -\frac{\pi}{2} & \frac{\pi}{2} & \frac{\pi}{2} \end{array} \right]^{\mathrm{T}}$$
$$x_{o_i} = \left[\begin{array}{cc|ccc} 0 & 0 & 1 & 0 & 0 & 0 \end{array} \right]^{\mathrm{T}}.$$

From these parameters, we can calculate the finger and object dynamics and the constraint from both the approaches. No redundant motion exists in this case.

The control objective is selected so as to realize a translational movement of the object from the initial position x_{o_i} to the target position

$$x_{o_f} = \left[\begin{array}{ccc|ccc} 0 & 0.5 & 0.8 & 0 & 0 & 0 \end{array} \right]^{\mathrm{T}}, \tag{2.105}$$

while regulating the internal force $f_N = 0 \rightarrow 1$ [N]. The desired trajectories of the object motion and the internal force are generated by interpolating the initial and target variables by the sigmoid function.

The simulation is conducted on a commercial package, Matlab/Simulink (R2016b, MathWorks Inc., Natick, MA). The solver for the dynamical equations is set to auto with variable step size and the default options (relative error tolerance=10^{-3}, maximum step=auto, etc.).

2.9.1.2 Numerical results

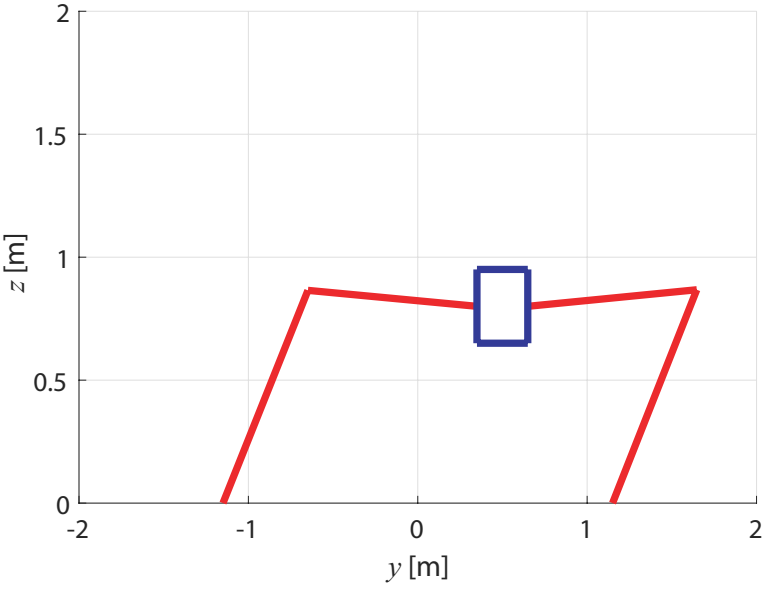


Figure 2.5: Final configuration of the two-link finger simulation

Figure 2.5 shows a screen shot of the final configuration for the proposed method. The time history of the control variables are shown in Fig. 2.6.

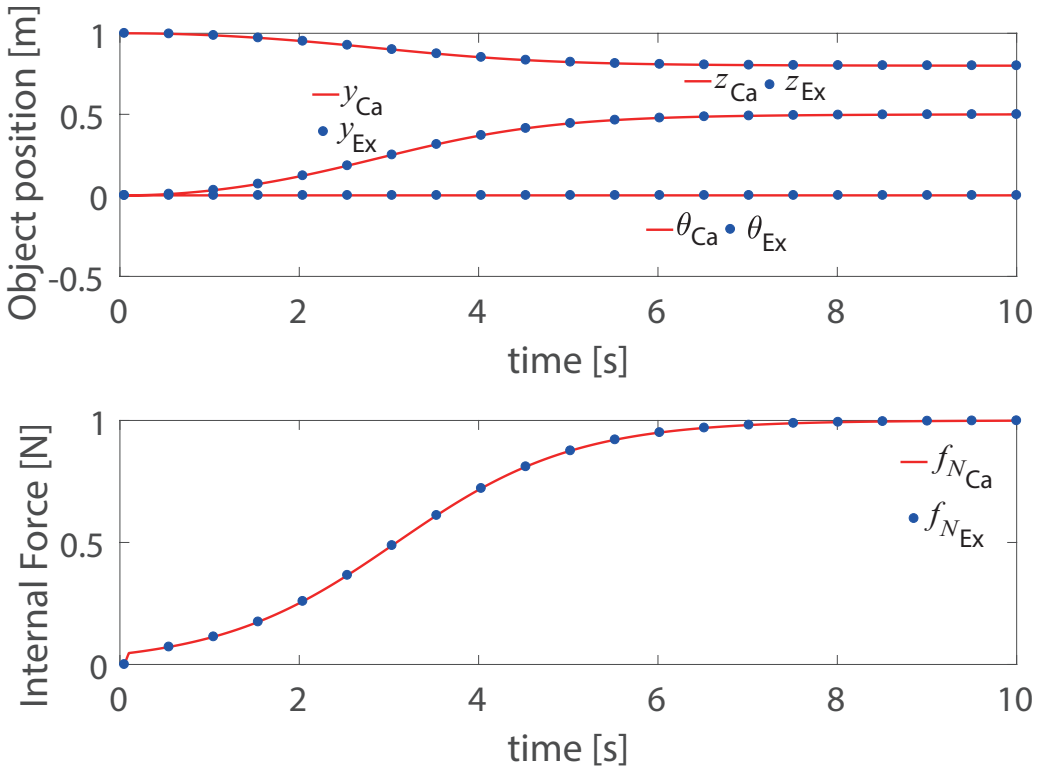


Figure 2.6: Object motion x_o and internal force f_N for the two-link finger

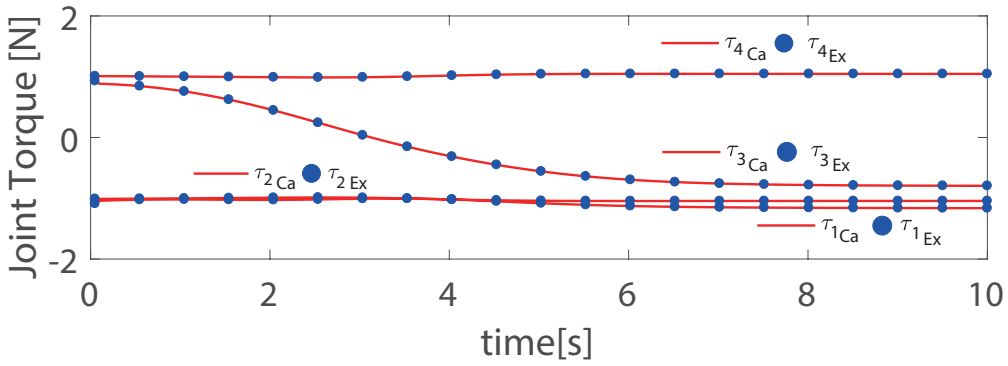


Figure 2.7: Control input τ for the two-link finger

The upper figure shows the object motion, and the lower figure shows the internal force. The response for the proposed approach is shown in the solid line and that for the conventional approach is shown in dots. Similarly, the time history of the control inputs is shown in Fig. 2.7. It is verified that both the results are in good agreement.

2.9.2 Modeling and control of the arm–hand system

2.9.2.1 Simulation settings

In this section, we illustrate the proposed algorithm more precisely by using a more complex system: the arm–hand system shown in Fig. 2.8.

The number of joints in the system is n=16 (Joints 1–3 and 5–7 of the arm part are ball-joints, composed of three rotational joints), and the number of chains is C=3. We define the system base frame Σ_s at the center of the first ball-joint (Joints 1–3) and the object frame Σ_o at the center of mass of the grasped object. For simplicity, the object is a cube of length l_o and mass m_o. All the links forming the arm part are of identical length l_a and mass m_a. The length of the palm is l_p and its mass m_p, and the length of the fingers are l_f and their mass m_f. These parameters are summarized in Table 2.2.

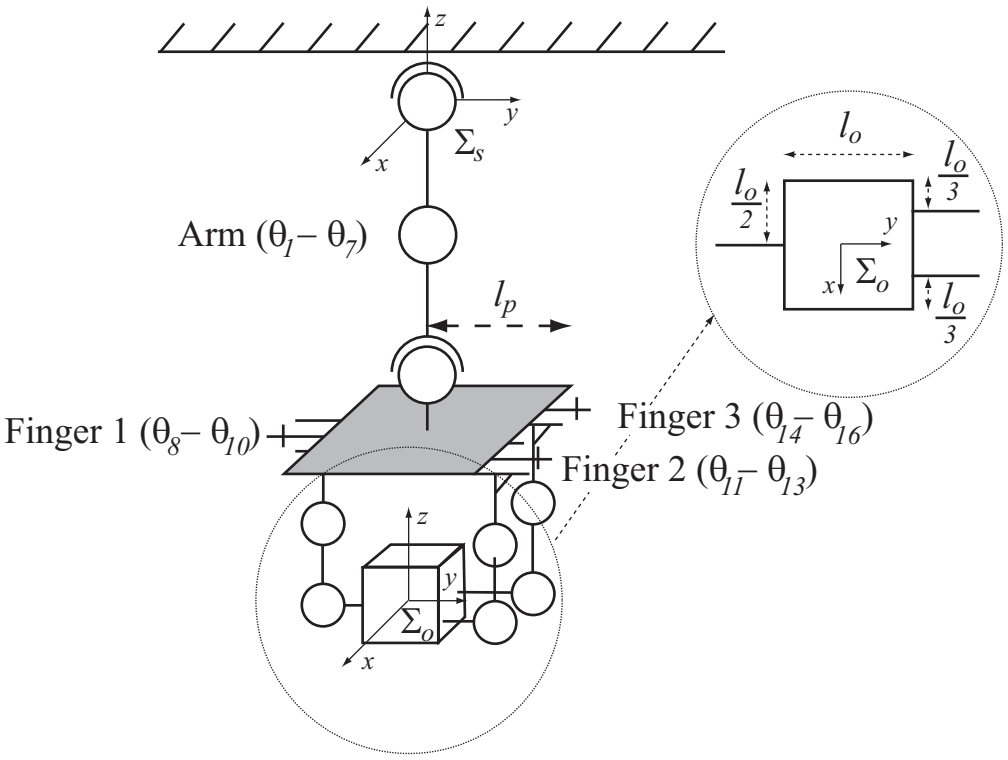


Figure 2.8: Arm–hand system

Table 2.2: Intrinsic parameters of the arm-hand simulation

Manipulator parameters	Arm links	Hand links
Length [m]	l_a=0.25	l_f=0.02
Mass [kg]	m_a=0.25	m_f=0.02
Length of the palm [m]	l_p=0.064	
Mass of the palm [kg]	m_p=0.064	
Object parameters		
Length [m]	l_o=0.06	
Mass [kg]	m_o=0.1	

Then, we define the reference configuration of the manipulator at $\theta = 0$ as shown in Fig. 2.8. From this reference configuration, we define all the constant exponential parameters (q_i,v_i,ω_i), the twists (ξ_i), and the configuration of the rigid links at the reference configuration ($g_{si^c}(0)$) using the definitions in section 1.2 and section 2.3.

Observing the structure of the manipulator in Fig. 2.8, we define the chain matrix and the simplified chain matrix as

$$C_h = \left[\begin{array}{ccccccc|ccccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right]$$

$$\widehat{C}_h = \left[\begin{array}{ccccccc|ccccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right].$$

The contact model is selected to be a point contact with friction for each contact, implying that each contact has $p = 3$ constraint motions, and the wrench basis in Eq. (2.54) is expressed by

$$B_{c_i}^{\mathrm{T}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

The contact points are selected as in the upper right figure in Fig. 2.8, and

the corresponding configurations are

$$g_{o_{c1}} = \left[\begin{array}{ccc|c} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -\frac{l_o}{2} \\ 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right], \quad g_{o_{c2}} = \left[\begin{array}{ccc|c} 1 & 0 & 0 & \frac{l_o}{6} \\ 0 & 0 & -1 & \frac{l_o}{2} \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right],$$

$$g_{o_{c3}} = \left[\begin{array}{ccc|c} 1 & 0 & 0 & -\frac{l_o}{6} \\ 0 & 0 & -1 & \frac{l_o}{2} \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right].$$

As described in the previous sections, all the coefficient matrices $\{M_f, C_f, N_f\}$, $\{M_o, C_o, N_o\}$ and $\{J_h, G\}$ present in the complete system equations Eq. (2.21), Eq. (2.22), and Eq. (2.23) can be computed from the above variables.

2.9.2.2 Simulation task

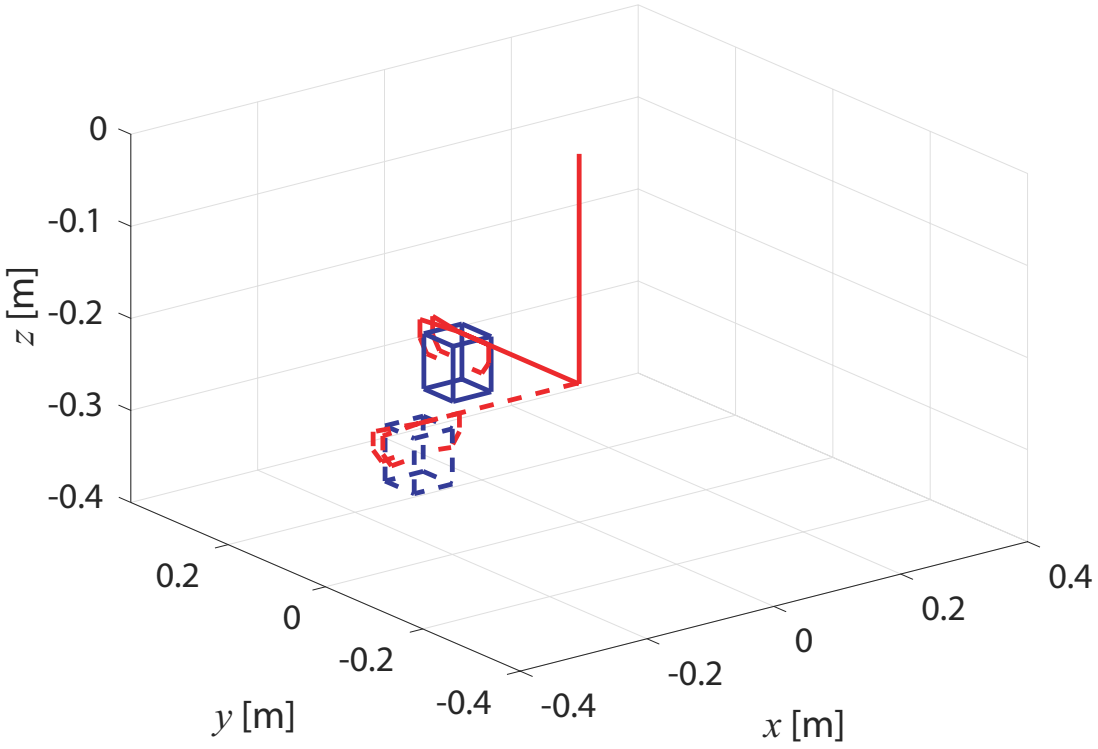


Figure 2.9: Initial configuration of the arm-hand system

We set the initial configuration as shown in Fig. 2.9 and consider the control task to crane the object 90 degrees counter-clockwise around the z axis to displace the object, while keeping the motion of the object relative to the palm fixed, as shown by the dashed line in Fig. 2.9.

In this system, we have seven ($= n - k = 16 - 3 \times 3$) degrees of redundant motion. The task can be achieved by rotating the arm 90 degrees around the arm root link while regulating the object motion x_o as observed from the palm coordinate to be constant. The crane motion can be accomplished by the arm joints. Thus, we select the arm joint velocities $\dot{\theta}_a = \begin{bmatrix}\dot{\theta}_1 & \cdots & \dot{\theta}_7\end{bmatrix}^{\mathrm{T}}$ as the redundant motion v_r, i.e.,

$$T_h = \left[\begin{array}{ccccccc|ccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right].$$

With the lengths in meters and the angles in radians, the initial values of the redundant motion $x_r = \theta_a$ and object motion x_o are

$$x_{r_i} = \left[\begin{array}{ccc|c|ccc} 0 & 0 & 0 & \frac{\pi}{2} & -\frac{\pi}{2} & 0 & 0 \end{array}\right]^{\mathrm{T}}$$
$$x_{o_i} = \left[\begin{array}{ccc|ccc} 0 & 0.25 & -0.28 & 0 & 0 & 0 \end{array}\right]^{\mathrm{T}},$$

and the final values are set to

$$x_{r_f} = \left[\begin{array}{ccc|c|ccc} 0 & 0 & \frac{\pi}{2} & \frac{\pi}{2} & -\frac{\pi}{2} & 0 & 0 \end{array}\right]^{\mathrm{T}}$$
$$x_{o_f} = \left[\begin{array}{ccc|ccc} -0.25 & 0 & -0.28 & 0 & 0 & \frac{\pi}{2} \end{array}\right]^{\mathrm{T}}.$$

The desired trajectory of the arm joints x_r are determined by interpolating its initial and final values by the sigmoid function, and the desired trajectories of the object motion x_o is generated from the desired velocity of x_r so that the object motion as observed from the palm is constant. For the internal force f_N in Newton, the initial and final values are set to

$$f_{N_i} = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^{\mathrm{T}}, \quad f_{N_f} = \begin{bmatrix}2 & 1 & 0.5\end{bmatrix}^{\mathrm{T}},$$

and the desired trajectory is generated by the sigmoid function. The control gains in the PD and PI controllers in Eq. (2.104) are set to $K_{d_o} = I_{13}$, $K_{p_o} = 1000 I_{13}$, $K_{i_f} = 0.01 I_3$.

2.9.2.3 Numerical results

The screen shot at the final configuration of the system is shown in Figs. 2.10 and 2.11. As is evident, the robot cranes the object 90 degrees

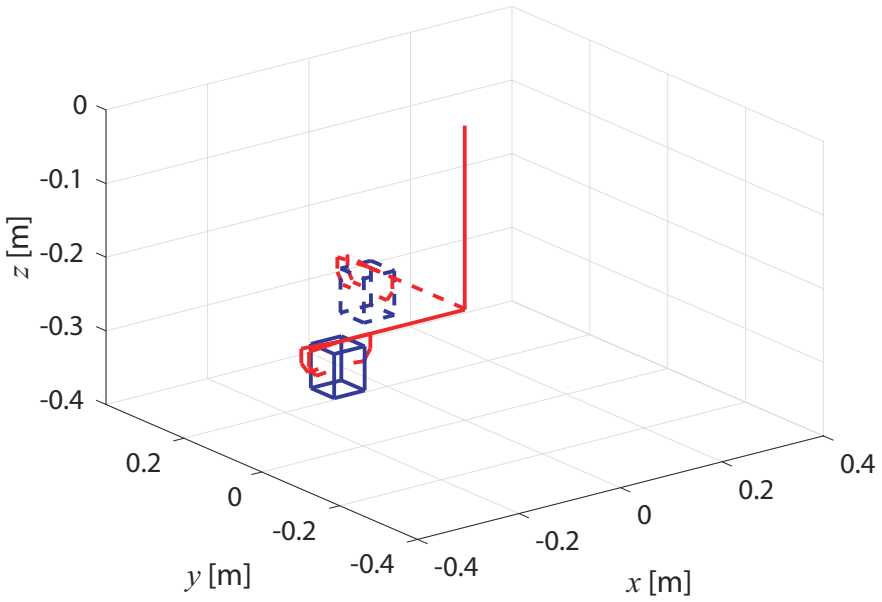


Figure 2.10: Final configuration of the arm-hand system

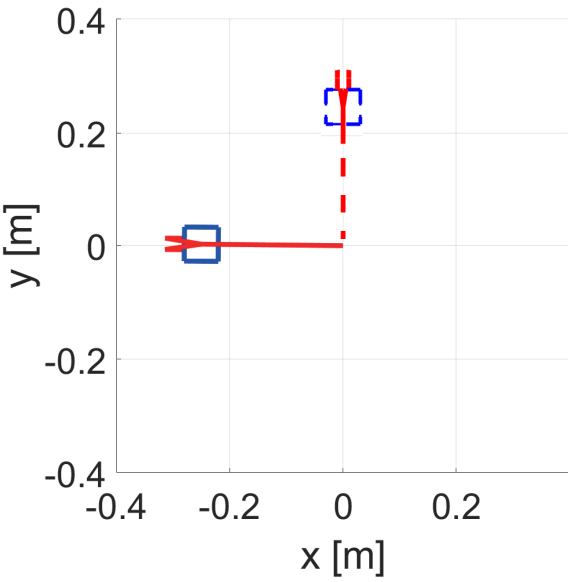


Figure 2.11: Final configuration of the arm-hand system (Top view)

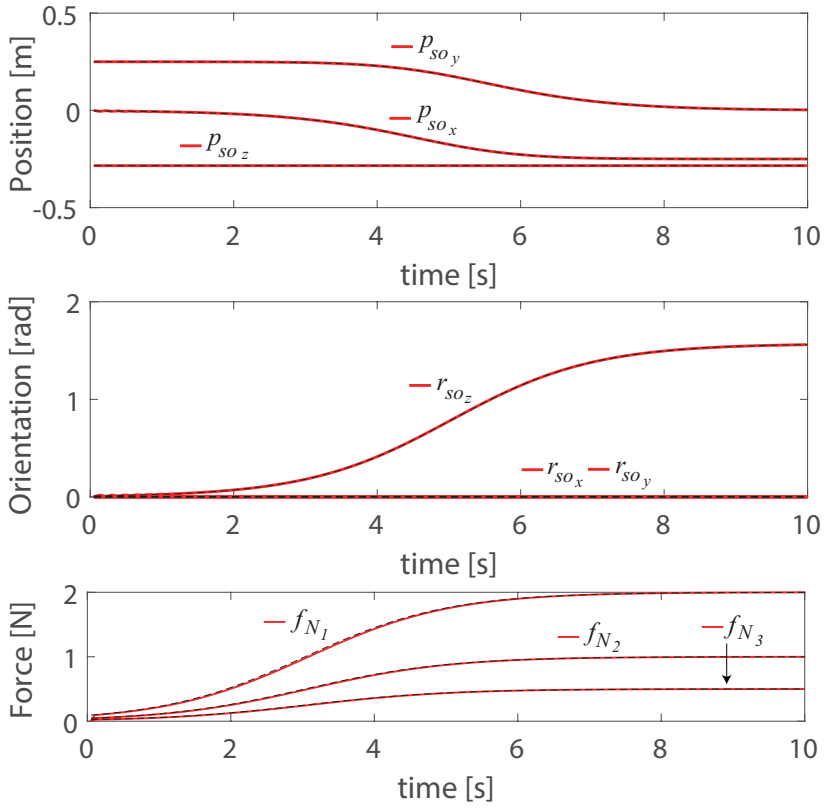


Figure 2.12: Object position p_{so}, rotation r_{so}, and internal force f_N for the arm-hand system

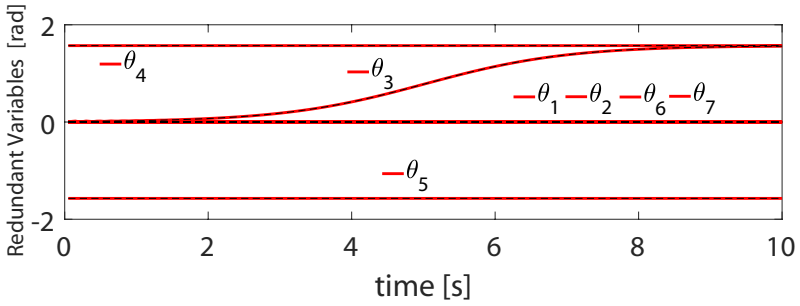


Figure 2.13: Redundant variable motion $x_r = \theta_a$ for the arm-hand system

about the z axis while keeping the object motion as observed from the palm to be fixed. The time history of the object position p_{so}, orientation r_{so}, and internal force f_N are shown by the solid line in Fig. 2.12 (dashed line represents the desired trajectory). The time history of the redundant variable $x_r = \theta_a$ is also shown in Fig 2.13. As shown in the figure, the control variables follow their desired trajectories, and the control objectives are successfully accomplished.

2.9.3 System design by optimization

2.9.3.1 Simulation settings

As described in section 2.8, the proposed dynamics and controller are defined from simple and intuitive parameters, and we can conveniently change the system configurations from one design to another. In this section, we conduct a simple optimization process for the system design to demonstrate this property.

As a system design problem, we consider the arm–hand system and the task in section 2.9.2, and suppose we want to change (or design) the direction of the elbow (θ_4) joint to minimize some cost. We can immediately formulate this problem by parametrizing the rotation parameter ω_4 in the spherical coordinates as

$$\omega_4 = \begin{bmatrix} \cos(\theta)\cos(\phi) \\ \cos(\theta)\sin(\phi) \\ \sin(\theta) \end{bmatrix}. \tag{2.106}$$

The optimal parameters can be determined by evaluating the cost over a specified range of (θ, ϕ). This situation is schematically illustrated in Fig. 2.14.

We can choose the cost depending on the manipulation requirements. In robotics, the manipulability and the consumption energy are often of interest, so we consider a cost composed of the sum of them. Let $J_h(\theta_f)$ be the manipulator Jacobian to the palm velocity $\dot{x}_p$, i.e.,

$$\dot{x}_p = J_h(\theta_f)\dot{\theta}_f. \tag{2.107}$$

The cost to minimize is defined as

$$J = \alpha J_1 + \beta J_2, \tag{2.108}$$

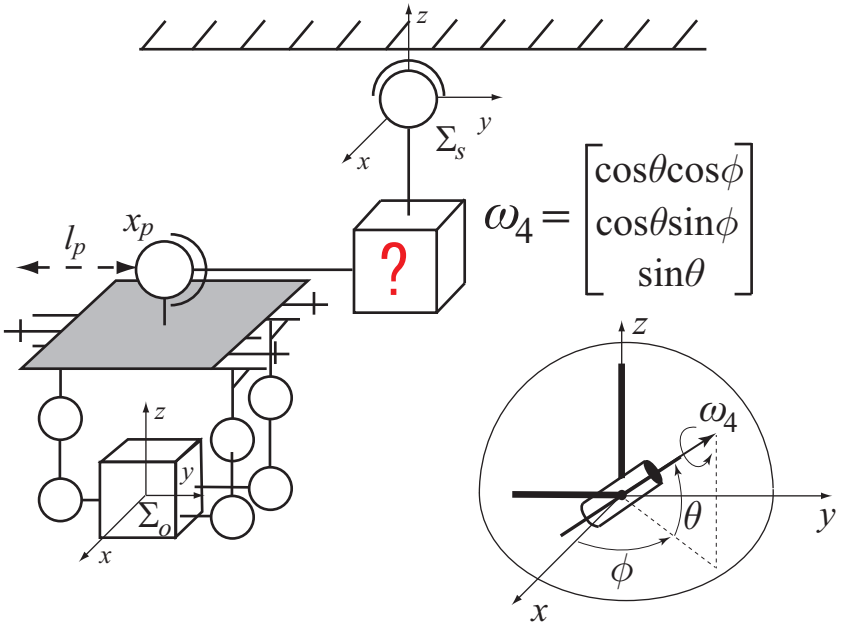


Figure 2.14: Arm-hand system with undetermined elbow direction

where

$$J_1 = \int_{t_0}^{t_f} \frac{1}{\sqrt{\det(J_h^{\mathrm{T}} J_h)}}\, dt \quad \text{(manipulability)}$$

$$J_2 = \int_{t_0}^{t_f} \tau^{\mathrm{T}} \dot{\theta}_f dt \qquad \text{(consumption energy)}$$

where $\alpha > 0$, $\beta > 0$.

The task given to the manipulator is the same crane movement in the previous section.To simplify the definition of the initial angles of the manipulator, we choose the reference configuration as illustrated in Fig. 2.14. Since the elbow joint is skewed and we can not determine the arm joint angles intuitively, we choose the internal velocity v_n as the redundant motion v_r ($T_h = K_h$) in this case. The desired trajectory of the redundant motion is simply set to $v_{n_d} = x_{n_d} \equiv 0$. Desired trajectory of the object motion x_o and the other simulation settings are kept unchanged from the previous section.

To find the optimal ω_4, we conduct the simulation and evaluate the cost J over the range $0 \leq \theta < \pi/2$, $0 \leq \phi < \pi/2$ by the increment of 0.17 [rad] ($\simeq$ 10 [deg]). The weights in the cost are set as $\alpha = 0.1$ and $\beta = 0.9$, to prioritize the energy needed for the object motion.

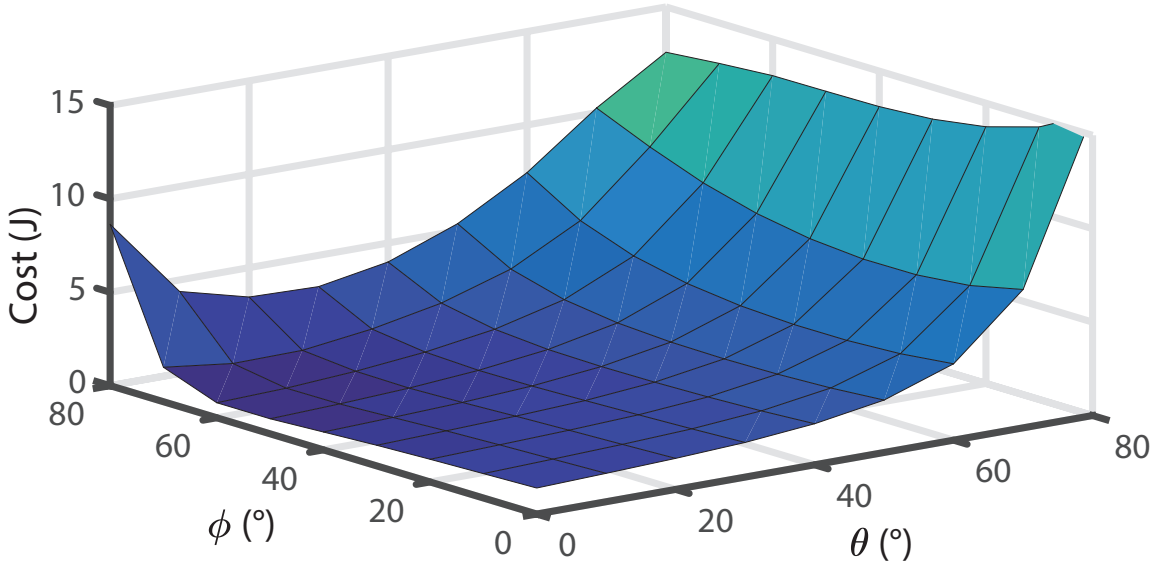


Figure 2.15: Value of the cost function J over (θ, ϕ) for the elbow direction

2.9.3.2 Numerical results

Figure 2.15 shows the value of the cost over the range of (θ, ϕ). In the figure, the origin $(\theta, \phi) = (0, 0)$ ($\omega_4 = [1\ 0\ 0]^{\mathrm{T}}$) corresponds to the configuration chosen in the previous section. From the figure, we first observe that the cost rapidly increases as $\theta \to \pi/2$ ($\omega_4 \to [0\ 0\ 1]^{\mathrm{T}}$). This is because the elbow axis aligns with the upper arm at the limit and the Jacobian becomes singular. The cost takes small values from the origin along ϕ axis. So for the crane movement it seems reasonable to set the elbow axis in the horizontal plane ($\theta = 0$) as in the previous section. It is interesting to see that the cost has a minimum of $J = 0.64$ at $(\theta, \phi) = (0, \pi/3)$ ($\omega_4 = [1/2\ \sqrt{3}/2\ 0]^{\mathrm{T}}$) which differs from the original configuration $\omega_4 = [1\ 0\ 0]^{\mathrm{T}}$ used in the previous section. The result suggests that we better choose a non-intuitive direction of the elbow axis at some intermediate angle between x and y axes in the horizontal plane for this specific task and cost.

The example is just a simple and conceptual one, but it illustrates the potential of the proposed formulas for future system and control designs. Note that we can also conveniently consider optimization problems to design the joint/link distribution (connection, number of joints/links) by employing the chain matrix as optimization variables.

2.10 Summary of tree-type systems kinematics, dynamics, and control

In this chapter, we have shown that the modeling and control process for a manipulator system grasping an object can be done in an automatized manner, based solely on the exponential parameters describing the joint information in the system q_i, v_i, ω_i and the chain matrices describing the system architecture C_h and $\widehat{C}_h$:

- In section 2.4, we proposed a new way of describing the connectivity between joints by introducing the chain matrix C_h in Eq. (2.17) and the simplified chain matrix $\widehat{C}_h$ in Eq. (2.19).
- In section 2.5, we modified the general equation of motions of a manipulator grasping an object given in Eqs. (2.21)-(2.23) by adapting the computation method for their main components such as the inertia, Coriolis matrix and potential vector to the tree-type systems architecture in a general fashion.
- In section 2.6, we improved the constraint equation to a more general case when the systems exhibits redundancy (which is often if not always the case for tree-type systems) in Eq. (2.71). Additionally, we proposed a general control strategy to apply for such systems.
- Finally in section 2.9, we studied an example of a tree-type system, the arm-hand system, where the simulation was successfully conducted.

As we observed in the end of the simulation example, with the proposed general formulation of the kinematics and dynamics, it is easy to change any basic parameters of the system. Additionally, a change in one of those parameters will not affect the other parameters, allowing for independent definition of the systems characteristics. Those features are primordial for complex mechanical system design, as it allows for global optimization method based on those versatile parameters. The adaptation of those parameters to global optimization scheme such as the genetic algorithms will be explored in depth in the next chapter.

Chapter 3

Global design optimization of tree-type robotic systems

3.1 Purpose of this chapter

In the previous chapter, we mainly focused on the description of the equations of motion governing a tree-type manipulator grasping an object:

$$M_f\ddot{\theta}_f + C_f\dot{\theta}_f + N_f = \tau - J_h^{\mathrm{T}}\lambda \tag{3.1}$$

$$M_o\dot{V}_{so}^b + C_oV_{so}^b + N_o = G\lambda \tag{3.2}$$

$$\underbrace{\begin{bmatrix} J_h \\ T_h \end{bmatrix}}_{\overline{J}_h}\dot{\theta}_f = \begin{bmatrix} G^{\mathrm{T}} & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} V_{so}^b \\ v_r \end{bmatrix}, \tag{3.3}$$

We also established that those equation can be automatized and based only on 4 parameters (the position of the joints given q, the orientation and type of the joints given by v and ω) and the connectivity between the joints given by C_h.

Here, we will focus on the implementation of those newly derived formulations in optimization schemes, to allow for a greater array of design possibilities for robotic systems by combining geometric and topological simultaneous optimization. As it can be seen from the nature of those parameters, the modeling possibilities for robotic systems are nearly endless. As such, we should aim to combine them with optimization schemes allowing for a wide array of data analysis. The best suited candidates of this condition are evolutionary algorithms, and more specifically genetic algorithms since they can scan a large array of data and thus analyse and compare a wide variety of robotic designs.

In this chapter, we will start by further improving the potency of our theories by extending their application to other complex systems whose architecture are closed to the tree-type systems, namely floating base systems and closed-chain systems. Then, since genetic algorithms are evolutionary algorithms working with parameters coded into binary strings, we will then propose methods to code and transform the afore mentioned parameters in binary expression, to implement them into the optimization scheme. This way, a set of strings of binary numbers will correspond to a type of configuration for the robot system. Additionally, we will propose evaluation methods using feedback control for those systems, which will be used to rate the robotic architecture for given tasks, rank them, and find the opti-

mized design for specific tasks. This optimization will be conducted at the end of the chapter, and we will discuss the obtained results.

3.2 Extension to floating base and closed-chain systems

When considering tree-type systems, it was assumed that the root joint is fixed in the environment and that no closed path exists in the tree. However, this is not a challenging limitation as we can treat the systems resulting from these assumptions in our framework with a slight modification. These systems are known as floating base systems and closed-chain systems. This section describes these systems and the necessary modification procedures.

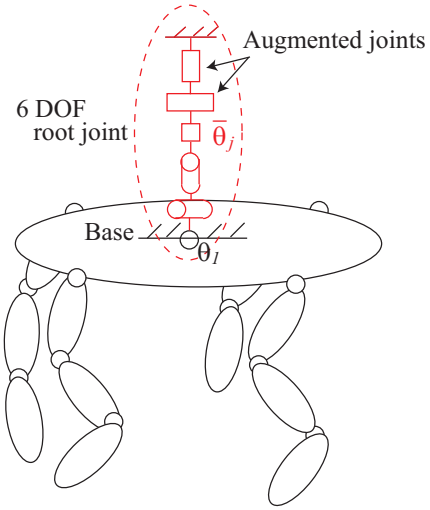
3.2.1 Floating base systems

Humanoids and mobile robots/vehicles are examples of floating base systems. Such structures can be described by adding artificial joints to ensure that the root joint has six DOFs (three translation and three rotation).

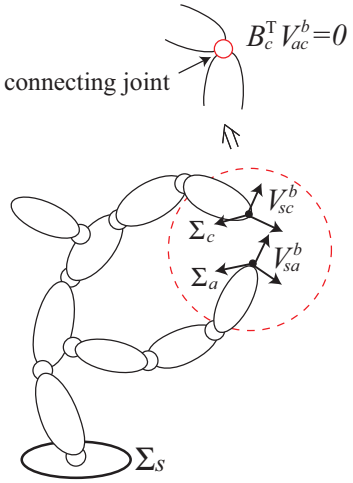
To illustrate this process, we consider the legged robot system shown in Fig. 3.1(a). The original tree-type system is shown in black; it comprises four chains with a revolute root joint fixed at the roof. To ensure that the base moves freely, we add 5-DOF joints (red joints) to the root joint. This modification can be performed by adding five columns of ones to the chain matrix from the left

$$\overline{C}_h = \left[\begin{array}{ccccc|ccccccc} 1 & 1 & 1 & 1 & 1 & 1 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & & & & & & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & 1 & & & C_h & & & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & & & & & & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & \cdots & \cdots & \cdots & \cdots & \cdots & 1 \end{array}\right], \tag{3.4}$$

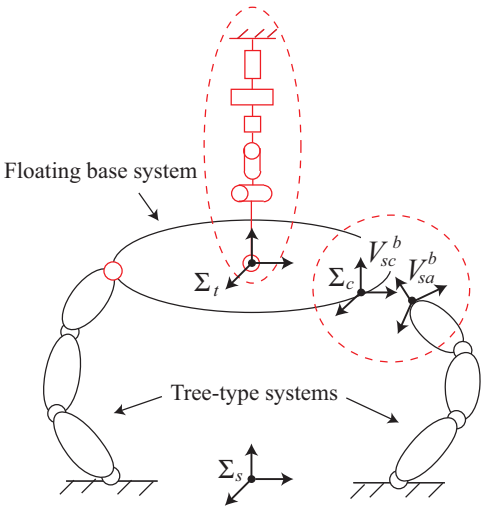
and by defining the exponential parameters of the augmented joints $\overline{\theta}_j$ ($j = 1, \cdots, 5$) according to the root joint property. If the root joint is rotational about the z-axis ($\omega_1 = [0, 0, 1]^{\mathrm{T}}$) located at q_1, the exponential



(a) Floating base system



(b) Closed-chain system



(c) Robotic platform

Figure 3.1: Extension to general tree-type structures

parameters are

$$\overline{q}_1 = \cdots = \overline{q}_5 = q_1, \text{ and, } \overline{v}_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}, \ \overline{v}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}, \ \overline{v}_3 = \begin{bmatrix}0\\0\\1\end{bmatrix}, \\ \overline{\omega}_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}, \ \overline{\omega}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}. \tag{3.5}$$

The mass $\overline{m}_{j^c}$ and inertia $\overline{I}_{j^c}$ of the links are set to zero.

After these modifications, the kinematics and dynamics of the floating base system can be calculated from the standard derivation process of tree-type systems.

3.2.2 Closed-chain systems

Examples of closed-chain systems include parallel link manipulators and robotic platforms, such as the Stewart platform. Such structures can be described by connecting multiple chains or tree-type systems by imposing geometrical constraints.

To illustrate this process, consider the parallel link manipulator shown in Fig. 3.1(b). The original tree-type system comprises three chains, and the objective is to connect two chain tips to construct a parallel link mechanism. To describe the constraint, we define frames Σ_a and Σ_c on each tip and choose their initial configurations to coincide with each other. Let V^b_{sa} and V^b_{sc} be the body velocity of these frames, and let V^b_{ac} be their relative body velocity. From Eq. (2.16) and Eq. (2.11), by introducing an intermediate frame Σ_s (base frame), V^b_{ac} can be described by V^b_{sa} and V^b_{sc} as

$$V^b_{ac} = \mathrm{Ad}^{-1}_{g_{sc}} V^b_{as} + V^b_{sc} = -\mathrm{Ad}^{-1}_{g_{sc}} \mathrm{Ad}_{g_{sa}} V^b_{sa} + V^b_{sc} \tag{3.6}$$

$$= \underbrace{\left(-\mathrm{Ad}^{-1}_{g_{sc}} \mathrm{Ad}_{g_{sa}} J^b_{sa} + J^b_{sc}\right)}_{J^b_{ac}} \dot{\theta}. \tag{3.7}$$

The constraint can be represented by constraining the elements of $V^b_{ac} = [(v^b_{ac})^{\mathrm{T}} \ (\omega^b_{ac})^{\mathrm{T}}]^{\mathrm{T}}$ as

$$B_c^{\mathrm{T}} V^b_{ac} = B_c^{\mathrm{T}} J^b_{ac} \dot{\theta} = 0. \tag{3.8}$$

In Eq. (3.8), $B_c \in \mathbb{R}^{6\times n_c}$ is a constant matrix comprising unit vectors $e_i^{\mathrm{T}} \in \mathbb{R}^6$, where $e_1 = [1,0,0\,|\,0,0,0]^{\mathrm{T}}$, $e_2 = [0,1,0\,|\,0,0,0]^{\mathrm{T}}, \cdots$, $e_6 = [0,0,0\,|\,0,0,1]^{\mathrm{T}}$. In addition, the complement matrix $\overline{B}_c$ is defined such that the column vectors of B_c and $\overline{B}_c$ span $\mathbb{R}^6$. For example, if Σ_c can

rotate freely with respect to Σ_a, $B_c = [e_1, e_2, e_3]$ and $\overline{B}_c = [e_4, e_5, e_6]$. If Σ_c can rotate only about the z-axis, $B_c = [e_1, e_2, e_3, e_4, e_5]$ and $\overline{B}_c = e_6$. The freedom of these constraints can be regarded as those of additional joints (a ball joint in the first example, and a 1-DOF revolute joint in the second example). These joints are considered the connecting joints. The connecting joints can be active, and $\overline{B}_c$ represents the direction of their joint force/torque. If several closed paths exist, their constraints are stacked in the form of Eq. (3.8).

According to the constraint, the constraint force λ_c arises in the direction of B_c of V_{ac}^b. If the connecting joint is active, the joint force/torque also arises in the direction of $\overline{B}_c$ with the force $\overline{\lambda}_c$. Upon adding these constraints and forces to the original three-type system dynamics, the closed-chain system dynamics can be expressed as

$$\begin{cases} M\ddot{\theta} + C\dot{\theta} + N = \tau + (J_{ac}^b)^{\mathrm{T}} \left\{ B_c \lambda_c + \overline{B}_c \overline{\lambda}_c \right\} & \text{(closed-chain dynamics)} \\ B_c^{\mathrm{T}} J_{ac}^b \dot{\theta} = 0 & \text{(constraint)} \end{cases} \tag{3.9}$$

3.2.3 Platform systems

Upon combining the procedures for the floating base and the closed-chain system, robotic platforms can be described by considering the kinematics and dynamics of the tree-type system. This situation is illustrated in Fig. 3.1(c). The original system comprises two tree-type systems (legs) and a floating base system (table). The kinematics and dynamics in terms of the joint velocity $\dot{\theta}$ can be readily derived as in Eq. (3.9) by following the procedure described in the previous sections.

For the design and control of robotic platforms, it is sometimes convenient to describe the kinematics and dynamics in terms of the table position/orientation. Let Σ_t be the tool frame fixed on the table, and V_{st}^b be the body velocity of Σ_t with respect to Σ_s. Assuming that the constraint holds, V_{st}^b can be described by the joint angle as $V_{st}^b = J_{st}^b \dot{\theta}$ from Eq. (2.11) as well as by the position and rotation vectors $x_t = [p_{st}^{\mathrm{T}}\ r_{st}^{\mathrm{T}}]^{\mathrm{T}}$ of Σ_t as $V_{st}^b = T_t \dot{x}_t$, similar to Eq. (2.47). Upon equating both equations, the relationship between $\dot{x}_t$ and $\dot{\theta}$ can be given by

$$V_{st}^b = T_t \dot{x}_t = J_{st}^b \dot{\theta}. \tag{3.10}$$

It is also possible to describe the constraint Eq. (3.8) in terms of $\dot{x}_t$.

From Eq. (2.11), Eq. (2.16), and Eq. (3.10),

$$V_{sa}^b = J_{sa}^b \dot{\theta} \tag{3.11}$$

$$V_{sc}^b = \mathrm{Ad}_{g_{tc}}^{-1} V_{st}^b + V_{tc}^b = \mathrm{Ad}_{g_{tc}}^{-1} T_t \dot{x}_t, \tag{3.12}$$

and we use $V_{tc}^b = 0$ because both frames are fixed on the same link. Upon substituting Eq. (3.11) and Eq. (3.12) into Eq. (3.6) and applying the resultant equation to $B_c^{\mathrm{T}} V_{ac}^b = 0$, the constraint can be represented by

$$B_c^{\mathrm{T}} \mathrm{Ad}_{g_{tc}}^{-1} T_t \dot{x}_t = B_c^{\mathrm{T}} \mathrm{Ad}_{g_{sc}}^{-1} \mathrm{Ad}_{g_{sa}} J_{sa}^b \dot{\theta}. \tag{3.13}$$

Equations (3.10) and (3.13) form a kinematic relation between $\dot{\theta}$ and $\dot{x}_t$, and it is given by

$$\underbrace{\begin{bmatrix} T_t \\ B_c^{\mathrm{T}} \mathrm{Ad}_{g_{tc}}^{-1} T_t \end{bmatrix}}_{\overline{T}_t} \dot{x}_t = \underbrace{\begin{bmatrix} J_{st}^b \\ B_c^{\mathrm{T}} \mathrm{Ad}_{g_{sc}}^{-1} \mathrm{Ad}_{g_{sa}} J_{sa}^b \end{bmatrix}}_{\overline{J}_{st}} \dot{\theta}. \tag{3.14}$$

This kinematic equation is used for the cost evaluation described in section 3.6.1.

3.3 Optimization using GA

As noted in the previous section, the kinematics and dynamics of tree-type systems can be described by the parameters defined from a single base frame, and we can conveniently use these aspects for system design. Because optimization problems are large scale and highly nonlinear with mixed continuous and discrete variables, a GA [38, 39, 40] is employed for the optimization. This section describes the necessary procedures to formulate the design optimization problem by using the GA (see also section 1.3 for an introduction on the GA).

3.3.1 GAs

The basic concept of a GA involves observing a population of N samples called a *structure*, which is composed of *strings*. Each string is allocated a fitness that represents its performance and the probability of its survival through time. The population of structures undergoes three adaptation phases to produce the next generation by mimicking the process of nature evolution, as described below:

The first phase is called *reproduction*, in which a probability process is used to select the strings to give birth to their offspring. The probability

of survival in the next generation is weighted by the fitness of the strings. Some of the best population, called the *elite*, are selected automatically and preserved in the next generation without performing the adaptation process described below.

The second phase is called *crossover*, in which the selected individuals are merged. In this process, at a specified probability, each string is cut into two parts, and the parts from different strings are then combined to produce new individuals. To avoid being stuck in the local minima, the crossover can be conducted for different levels of strings [78, 79], such as for strings, higher-level strings (groups of strings), and structures (whole set of groups). For example, if s_{j_k} and s_{j_l} are the string elements of a higher-level string s_j and are the members of a group, we also conduct the crossover process for each string s_{j_k} and s_{j_l}, the string group $s_{j_{kl}} = [s_{j_k}\ s_{j_l}]$, as well as the whole string s_j.

The final phase is called the *mutation* phase, in which one value in a string is changed to another random value with a specified probability. The mutation can also be conducted for different levels of strings. One cycle of these three phases produces a new set of strings called the *next generation*. The cycle is repeated until the best fitness becomes constant or the target value is attained.

To conduct the adaptation processes, the GA strings are usually represented by binaries. By adopting the binary strings, we can easily implement the adaptation process by exchanging or manipulating their bits. Therefore, the main challenge in design optimization is to enable the binary coding of the system parameters for the GA process. In the exponential formulation, the parameters that we wish to code into the strings are the exponential parameters $\{q_i, v_i, \omega_i\}$ and the chain matrix $\{C_h\}$. Note that the length of the links l_{i^c} is determined from the joint position q_i, and it is not a design parameter. The link mass m_{i^c} and inertia I_{i^c} can also be functions of l_{i^c}.

3.3.2 Binary string and rate decoding

Binary strings can be naturally associated with natural numbers. Consequently, we identify the string and the natural number and denote both by s; that is, $s \in \mathbb{B}^{n_s}$ or $s \in \mathbb{N}$, where $\mathbb{B}$ is the set $\{0, 1\}$ and $\mathbb{N}$ is the set of natural numbers. A binary string $s \in \mathbb{B}^{n_s}$ can be associated with a real

number (rate or percentage) between $0 \leq p_s \leq 1$ by

$$p_s = \frac{s}{2^{n_s} - 1}. \tag{3.15}$$

Rate decoding p_s can be used in several ways depending on the problem. A real number x in a range $x_0 \leq x \leq x_1$ can be represented as

$$x = x_0 + (x_1 - x_0)p_s. \quad \text{(interval)} \tag{3.16}$$

A total number x can be distributed to x_i's by using multiple rate decoding strings p_{s_i} sequentially; for example,

$$x_1 = xp_{s_1},\ x_2 = (x - x_1)p_{s_2}, \cdots . \quad \text{(distribution)} \tag{3.17}$$

We can align the position x_i accordingly between a range $x_0 \leq x \leq x_1$ by

$$x_1 = x_0 + xp_{s_1},\ x_2 = x_1 + (x - x_1)p_{s_2}, \cdots . \quad \text{(sequential)} \tag{3.18}$$

These values can be rounded if natural numbers are required.

The exponential parameters are immediately described by the rate strings with this decoding. Note that the following coding will be much more complicated if the joint parameters are defined relative to its parent joint (not from the base frame) like the DH parameters.

3.3.3 GA coding for exponential parameters $\{q_i, v_i, \omega_i\}$

The joint position q_i can be determined from the triplet position binary strings $s_{q_{i_x}} \in \mathbb{B}^{n_{q_{i_x}}}$, $s_{q_{i_y}} \in \mathbb{B}^{n_{q_{i_y}}}$, $s_{q_{i_z}} \in \mathbb{B}^{n_{q_{i_z}}}$. The position can be chosen within a region by using Eq. (3.16) or can be ordered in a region by using Eq. (3.18).

The joint movement direction $\{v_i, \omega_i\}$ can be determined from the joint type (rotation or translation) and the unit vectors v_i or ω_i. These vectors can be described by two real numbers ϕ_i and γ_i in the polar coordinates as

$$v_i = \begin{bmatrix} \cos(\phi_i)\cos(\gamma_i) \\ \cos(\phi_i)\sin(\gamma_i) \\ \sin(\gamma_i) \end{bmatrix}, \quad \omega_i = \begin{bmatrix} \cos(\phi_i)\cos(\gamma_i) \\ \cos(\phi_i)\sin(\gamma_i) \\ \sin(\gamma_i) \end{bmatrix}, \tag{3.19}$$

where $0 \leq \phi_i \leq 2\pi$, $0 \leq \gamma_i \leq \frac{\pi}{2}$. The joint type can be specified by the one-bit binary string $s_{t_i} \in \mathbb{B}$. ϕ_i and γ_i can be described by the binary strings $s_{\phi_i} \in \mathbb{B}^{n_{\phi_i}}$ and $s_{\gamma_i} \in \mathbb{B}^{n_{\gamma_i}}$ with the interval decoding described in Eq. (3.16).

The next section describes the coding procedure of the chain matrix C_h.

3.4 GA coding for chain matrix

If the number of joints n and number of chains C are given, the chain matrix C_h is a $(0,1)$-valued $C \times n$ matrix. We can determine a $(0,1)$ matrix by the binary string for each element; however, most of them are infeasible from a structural viewpoint. In particular, the chain matrix can be coded simply by using $2C-1$ binary strings with range decoding, as described below.

3.4.1 Tree-type architectures

To examine the chain matrix characteristics from the tree-type system architecture, the example shown in the upper part in Fig. 3.2 is considered. The system comprises seventeen joints and five chains, and thus, the matrix is $C_h \in \mathbb{R}^{5\times 17}$. Recall that a row and a column of the chain matrix correspond to a chain and a joint, respectively; then, we can derive the chain matrix of the system as

$$C_h = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}. \tag{3.20}$$

It can be validated that the matrix elements are not random and have several series $(0,1)$ patterns arising from the tree-type system architecture. We utilize this aspect for the GA coding and determine the chain matrix from a smaller number of strings containing the key information. Toward this end, we note that the system comprises chains branching at the links driven by a joint, known as the branching joint. Subsequently, the system can be regarded as an assembly of the trunk and branches, as shown in the right-hand side of Fig. 3.2. Note that the branching joint is not included in the branch, and the branch starts from a link. When the start links are connected to a branching joint in the assembly, they are merged into the parent links. Therefore, the one-to-one relation of the joint and the link is retained, as shown in the lower part of Fig. 3.2.

In the following analysis, to avoid the multiplicity of the structure from different chain matrices, we assume that the joint numbers are assigned from the root to the tip from the first chain to the last and that each branch starts (sprouts) from the trunk tip or the elder (smaller number)

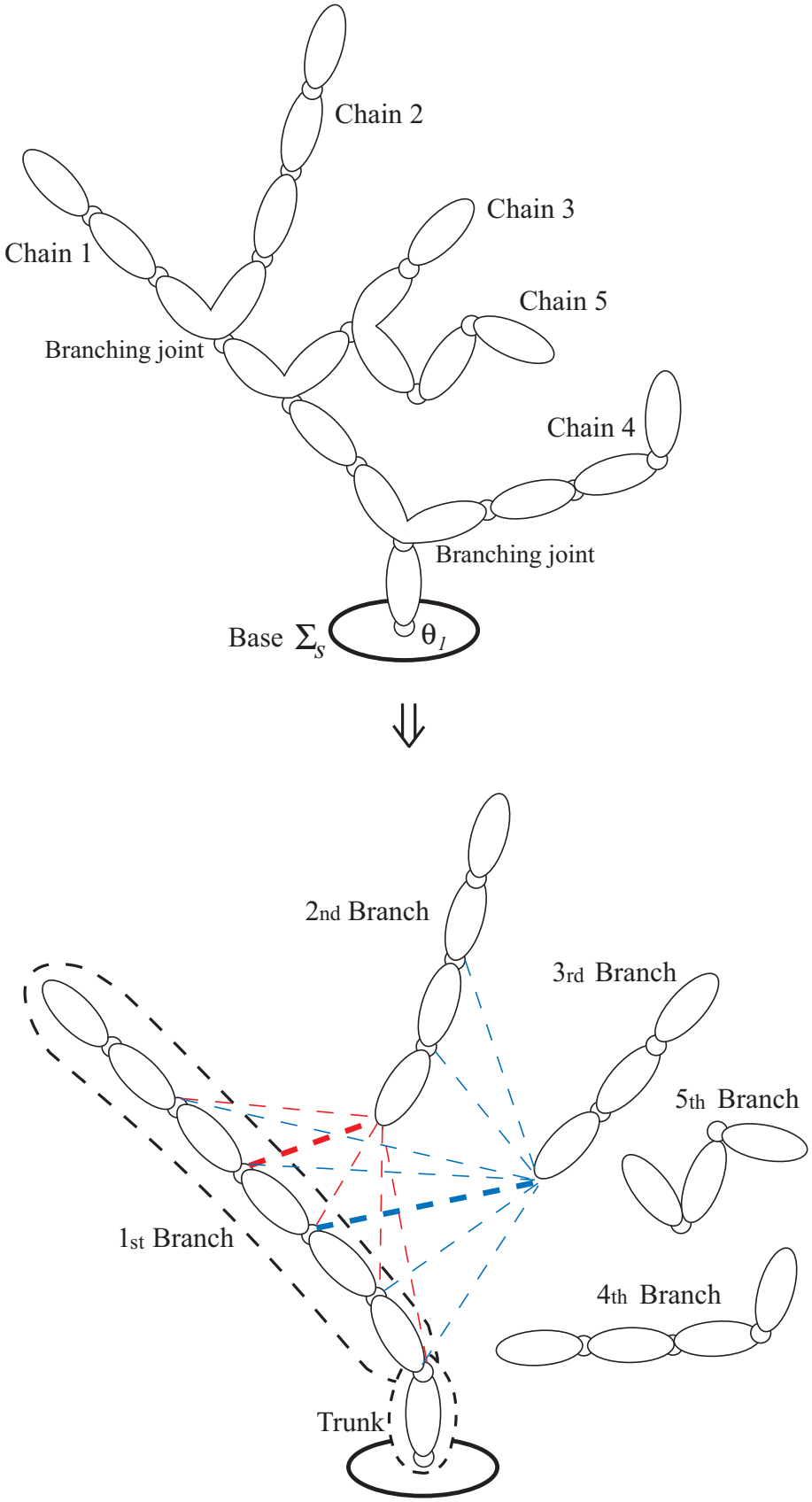


Figure 3.2: Tree-type system and system components

branches, as indicated by the red and blue dotted lines on the lower part of Fig. 3.2. Note that the first branch always starts from the trunk tip.

3.4.2 Construction of chain matrix from components

As mentioned previously, the system is characterized by the components (trunk and branches) and their connections to the branching joint. This section describes how the chain matrix is determined from this information.

The trunk and branches are characterized by the number of joints distributed to them. From these numbers, some of the elements of the chain matrix can be determined as

$$C_h = \left[\begin{array}{cc|ccccc|ccc|cc|ccc|cc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & & & & & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & & & & & 0 & & & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & & & & & 0 & & & 0 & & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & & & & & 0 & & & 0 & & 0 & & & 0 & 1 & 1 \end{array}\right]. \tag{3.21}$$

trunk 1st branch $\cdots$

In Eq. (3.21), the partition of each block is first determined from the given numbers of joints. The columns of the first block correspond to the trunk joints, and thus, we fill one in the first block, as they are contained in all the chains. The columns of the remaining blocks correspond to the branch joints, and thus, we fill ones in the row corresponding to the branch number. The upper elements of the branch joints ones are zero according to the joint numbering assumption. The elements below one in the last column of each branch block are also zero because the tip joint cannot belong to other chains.

The system connection is determined by selecting the branching joint for each branch from the second branch to the last one. As shown in Fig. 3.2, the branching joint for the second branch is the fifth joint; therefore, we fill one in the fifth column in the second row. This process is continued until the last row (chain), which leads to the underlined ones in blue, as follows:

$$C_h = \left[\begin{array}{cc|ccccc|ccc|cc|ccc|cc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & \underline{1} & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & \underline{1} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & \underline{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \underline{1} & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right]. \tag{3.22}$$

trunk 1st branch $\cdots$

From the position of the underlined ones (branching joints), the other elements in the same row can be determined. For the elements in the branch block involving the underlined ones (first branch block for the second row), we fill ones on the left and zeros on the right, as the chain branches off at the branching joint. The elements in the elder (or right-hand-side) branch blocks (third to fifth blocks for the second row) are set as zero from the numbering assumption. If younger branch blocks exist on the left-hand side of the underlined ones block (first and second branch blocks for the fifth row), we copy the (0,1) pattern from the row to which the branching joint belongs. For the fifth row, the branching joint belongs to the third row, and thus, we copy the elements of the first and second branch blocks in the third row. This process completes the chain matrix.

3.4.3 Strings for chain matrix

From the above observation, for a given n and C, the chain matrix can be constructed from

- n_t: number of joints in trunk
- n_{b_i} $(i = 1, 2, \cdots, C-1)$: number of joints in branch i (for the last branch, $n_{b_C} = n - (n_t + n_{b_1} + \cdots + n_{b_{C-1}})$)
- n_{c_i} $(i = 2, \cdots, C)$: branching joint number for branch i (for the first branch, $n_{c_1} = n_t$)

These numbers ((2C-1) in total) can be coded using binary strings with the following rate decoding:

- For the numbers of joints n_t and n_{b_i}, we use strings $s_{n_t} \in \mathbb{B}^{n_{n_t}}$, $s_{n_{b_i}} \in \mathbb{B}^{n_{n_{b_i}}}$ and apply the distribution decoding in Eq. (3.17) with rounding.
- For the branching joint number n_{c_i}, we use a string $s_{n_{c_i}} \in \mathbb{B}^{n_{nc_i}}$ and apply the interval decoding in Eq. (3.16) with rounding. The range of n_{c_i} is given by $n_{c_i} \in [0, (n_{b_1} - 1) + \cdots + (n_{b_{i-1}} - 1)]$ as the tip joint cannot be the branching joint. If $n_{c_i} = 0$, the trunk tip joint is the branching joint.

In the above example in Eq. (3.22), the joint distributions are $n_t = 2$, $n_{b_1} = 5$, $n_{b_2} = 3$, $n_{b_3} = 2$, and $n_{b_4} = 3$. The number and possible ranges of the branching joint location are $n_{c_2} = 3 \in [0, 4]$, $n_{c_3} = 2 \in [0, 6]$, $n_{c_4} = 0 \in [0, 7]$, $n_{c_5} = 7 \in [0, 9]$.

3.5 Overall GA procedure

It has been demonstrated that if the numbers of the joints n and chains C are given, the optimization parameters are the joint position q_i, joint direction $\{v_i, \omega_i\}$, and chain matrix C_h. Therefore, to minimize (or maximize) a cost J, the optimization problem can be stated as

Optimization problem

- given: n and C
- find: q_i, v_i, ω_i, and C_h
- minimize (or maximize): J

The optimization parameters are coded into the GA strings as

GA strings

- q_i: $s_{q_{i_x}} \in \mathbb{B}^{n_{q_{i_x}}}$, $s_{q_{i_y}} \in \mathbb{B}^{n_{q_{i_y}}}$, $s_{q_{i_z}} \in \mathbb{B}^{n_{q_{i_z}}}$ (interval/sequential decoding) $(i = 1, \cdots, n)$
- $\{v_i,\ \omega_i\}$: $s_{t_i} \in \mathbb{B}$ (binary decoding), $s_{\phi_i} \in \mathbb{B}^{n_{\phi_i}}$, $s_{\gamma_i} \in \mathbb{B}^{n_{\gamma_i}}$ (interval decoding) $(i = 1, \cdots, n)$
- n_t, n_{b_i}: $s_{n_t} \in \mathbb{B}^{n_{n_t}}$, $s_{n_{b_i}} \in \mathbb{B}^{n_{n_{b_i}}}$ (distribution decoding) $(i = 1, \cdots, C-1)$
- n_{c_i}: $s_{n_{c_i}} \in \mathbb{B}^{n_{nc_i}}$ (interval decoding) $(i = 2, \cdots, C)$

For the GA optimization, we assign the above strings for each individual. For the crossover and mutation operation for different levels, from the lower-level strings above, we define the higher-level string groups as follows:

String groups

- intermediate level: $s_{q_i} = [s_{q_{i_x}}, s_{q_{i_y}}, s_{q_{i_z}}]$, $s_{v_i} = [s_{t_i}, s_{\phi_i}, s_{\gamma_i}]$
- higher level: $s_q = [s_{q_1}, \cdots, s_{q_n}]$, $s_v = [s_{v_1}, \cdots, s_{v_n}]$,
  $s_{C_h} = [s_{n_t}; s_{n_{b_1}}, \cdots, s_{n_{b_{n-1}}}; s_{n_{c_2}}, \cdots, s_{n_{c_n}}]$
- total system: $s = [s_q, s_v, s_{C_h}]$

For the individual j, we denote the total system string (or structure) as s_j.

Fig. 3.3 shows a typical optimization process. From the initial population of N individuals (or total system string) s_j, we obtain the optimization parameters q_i, $\{v_i, \omega_i\}$, and C_h from the decoding rule of the strings. From these parameters, the physical parameters of the link mass, length, and inertia (m_{i^c}, l_{i^c}, and I_{i^c}, respectively); twist ξ_i; and simplified chain matrix $\widehat{C}_h$ can be obtained. The kinematics/dynamics and a controller can be determined from these values (see section 3.6 for details). By using the system and controller, a simulation can be performed to evaluate the fitness (or cost) J_j for each individual. If the best fitness attains the target

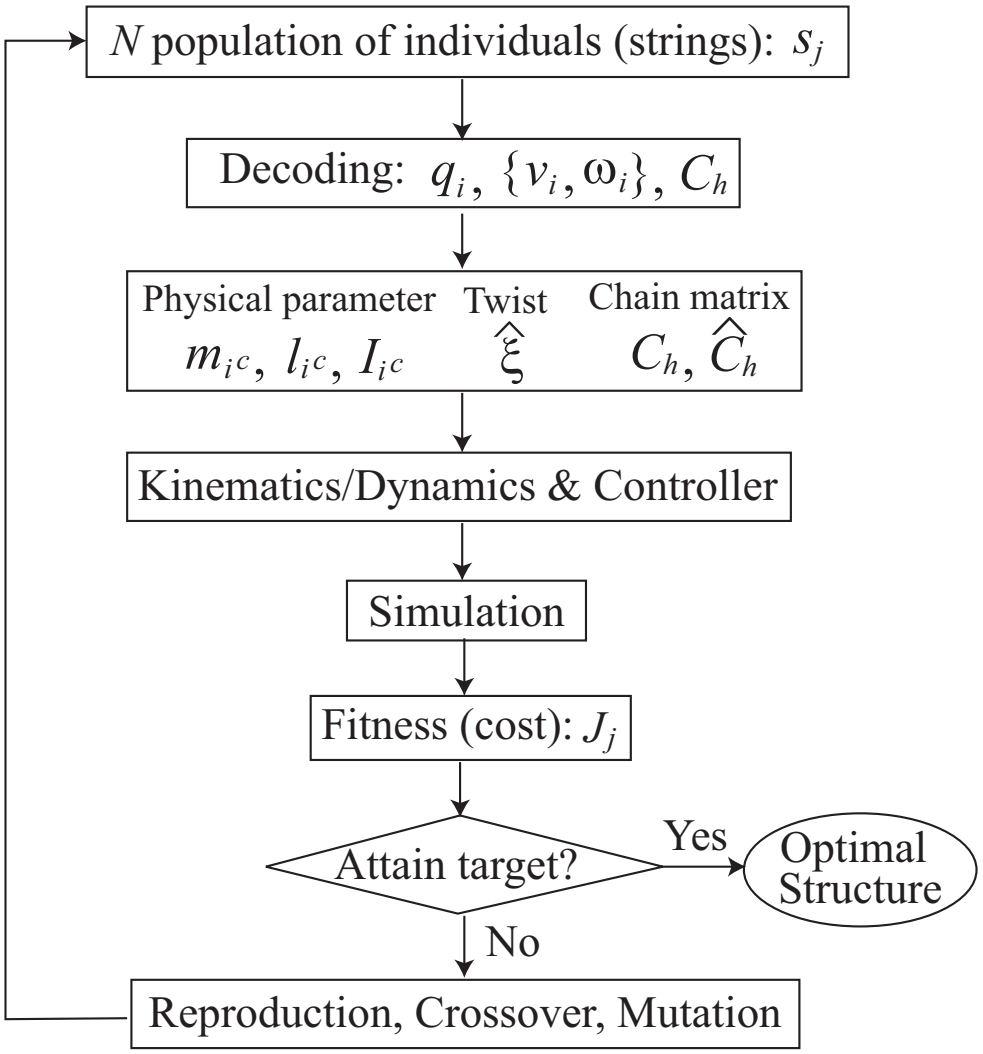


Figure 3.3: Process flow for genetic algorithm based optimization

value or becomes constant through the iteration, the optimal structure can be obtained. If not, the adaptation process is performed to yield the new generation, and the entire process is repeated.

3.6 Cost evaluation using feedback control

For the cost evaluation, we conduct a numerical simulation for each individual obtained from the strings. Because the general closed-form formulas for the kinematics and dynamics of tree-type systems are available, a controller of any type can be constructed from them. This allows us to perform numerical simulations to evaluate a variety of costs during the optimization process, for which analytical calculation formulas are unavailable. This section illustrates this evaluation process.

3.6.1 Cost evaluation by kinematics control

One of the most important measures that can be evaluated by the kinematics is the region that the manipulators can access. This region is referred to as the workspace, and it is especially important in designing closed-link manipulators. Solving the inverse kinematics is the most straightforward approach, and it is efficient for some simple manipulators [80, 81]. However, this approach is intractable in general as it requires solving several multivariable nonlinear equations [8]. This difficulty arises from tackling the position-level kinematics directly and can be alleviated by conducting a numerical simulation for the velocity-level kinematics with feedback control. This section illustrates this process by taking an example of the robotic platform shown in Fig. 3.4(a).

3.6.1.1 Basic idea

As is described in section 3.2.3, the kinematics of the robotic platform between the joint angle and the tool flame is given by Eq. (3.14). This equation can be viewed as a linear equation of $\dot{x}_t \in \mathbb{R}^6$ characterized by the joint velocity $\dot{\theta} \in \mathbb{R}^n$ and the coefficient matrices $\overline{T}_t \in \mathbb{R}^{(6+n_c)\times 6}$ and $\overline{J}_{st} \in \mathbb{R}^{(6+n_c)\times n}$ determined from the current joint angle θ. From linear algebra, a necessary and sufficient condition for the solution existence can be expressed as

$$\operatorname{rank} \overline{T}_t = \operatorname{rank} \left[\overline{T}_t \;\; \overline{J}_{st}\dot{\theta} \right]. \tag{3.23}$$

If Eq. (3.23) holds, one of the solutions (minimum norm solution) for Eq. (3.14) can be given as

$$\dot{x}_t = \overline{T}_t^+ \overline{J}_{st} \dot{\theta}, \tag{3.24}$$

where $\overline{T}_t^+$ is the pseudo-inverse of $\overline{T}_t$. If $\text{rank}\overline{T}_t = 6$, Eq. (3.24) is the unique solution.

Equation (3.23) can be used to evaluate the workspace. Suppose that the tool frame Σ_t is within the workspace at the current configuration θ and that we wish to move Σ_t toward the direction specified by $\dot{\theta}$. If Eq. (3.23) holds and has a solution $\dot{x}_t$, Σ_t can move along the direction $\dot{x}_t$. Consequently, for a small Δt, we can expect that $x_t + \dot{x}_t \Delta t$ is still within the workspace. If Eq. (3.23) does not hold, the specified direction is directed out of the workspace, and thus, Σ_t lies on the boundary of the workspace. Because the workspace is usually connected and has a smooth boundary, the workspace can be estimated by controlling the tool frame x_t along a path sweeping over a region of interest from the center until the condition in Eq. (3.23) fails. The numerical simulation for this purpose can be conducted by using Eq. (3.14) as the system whose control input is $\dot{\theta}$.

In the following sections, we define some typical path alternatives and a controller for the workspace evaluation.

3.6.1.2 Trajectory for workspace evaluation

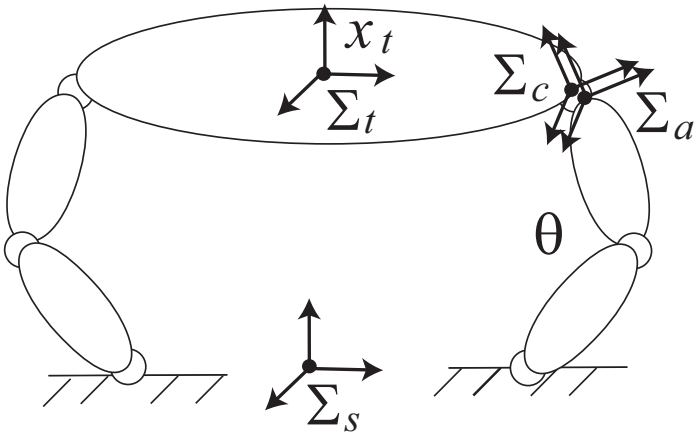
As is well known, a path $p(s) \in \mathbb{R}^3$ can be represented using a parameter between an interval $s = [0, 1]$. We can describe a segment of any interval using s:

$$l_k = ks \quad \text{(line)}, \qquad \phi_k = 2\pi ks \quad \text{(angle)}, \tag{3.25}$$

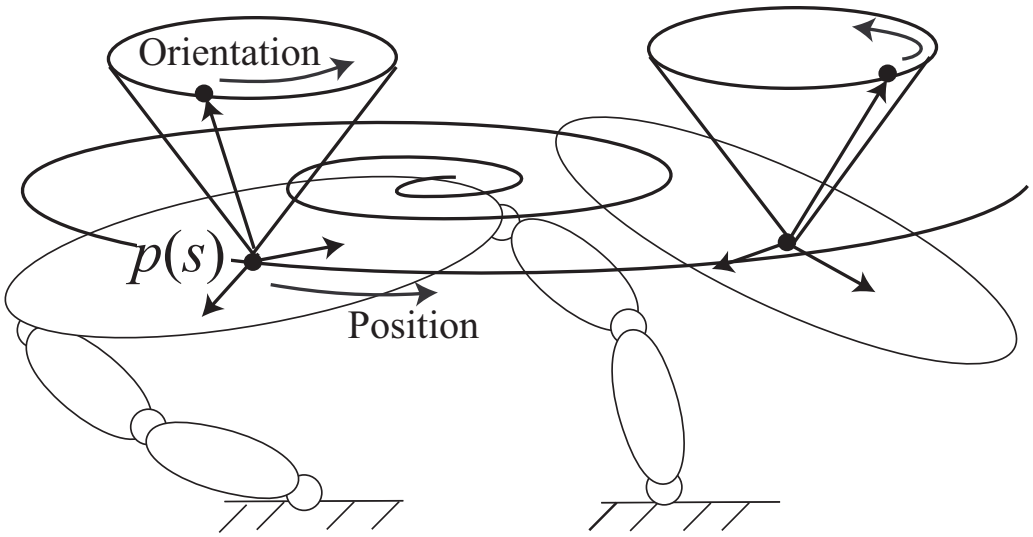
where $k \in \mathbb{R}$. By using these segments, we can describe a variety of paths; for example,

$$p(k_1) = \begin{bmatrix} r\cos(\phi_{k_1}) \\ r\sin(\phi_{k_1}) \\ h \end{bmatrix} \quad \text{(circle)}, \tag{3.26}$$

$$p(k_1, k_2) = \begin{bmatrix} l_{k_2}\cos(\phi_{k_1}) \\ l_{k_2}\sin(\phi_{k_1}) \\ h \end{bmatrix} \quad \text{(spiral)}, \tag{3.27}$$



(a) Robotic platform



(b) Trajectories for workspace evaluation

Figure 3.4: Trajectory for workspace evaluation

$$p(k_1,k_2)=\begin{bmatrix} r\cos(\phi_{k_1}) \\ r\sin(\phi_{k_1}) \\ l_{k_2} \end{bmatrix} \text{ (helix)}, \quad p(k_1,k_2)=\begin{bmatrix} r(1-l_{k_2})\cos(\phi_{k_1}) \\ r(1-l_{k_2})\sin(\phi_{s_1}) \\ l_{k_2} \end{bmatrix} \text{ (cone)}, \tag{3.28}$$

$$p(k_1,k_2)=\begin{bmatrix} r\cos(\phi_{k_2})\cos(\phi_{k_1}) \\ r\cos(\phi_{k_2})\sin(\phi_{k_1}) \\ r\sin(\phi_{k_2}) \end{bmatrix} \text{ (sphere)}. \tag{3.29}$$

Fig. 3.4(b) shows the typical trajectories to evaluate the workspace for a robotic platform. The possible translational region of the table center x_t can be evaluated by a spiral trajectory expanding from the center position. At each translational position, the table is required to tilt by a certain angle, which can be evaluated by a circle trajectory for the table normal to rotate. The spiral for the translation can be produced by the second equation of Eq. (3.26) with $\phi_{k_1} = 2\pi k_1 s$ and $l_{k_2} = k_2 s$, where $k_1 \gg k_2$ (the translational rotation should be faster than the radius expansion). The circle trajectory for the tilting is given by the first equation of Eq. (3.26) with $\phi_{k_3} = k_3 s$, where $k_3 \gg k_1$ (the tilting rotation should be faster than the translational rotation).

3.6.1.3 Controller and controlled system kinematics

For the workspace evaluation, we design a tracking controller for the system in Eq. (3.14). Because Eq. (3.24) is applicable if Eq. (3.23) holds, a simple choice for the controller is

$$\dot{\theta}_d = \left(\overline{T}_t^+ \overline{J}_{st}\right)^+ \left\{\dot{x}_{t_d} - K_p(x_t - x_{t_d})\right\}, \tag{3.30}$$

where $K_p > 0$ is some constant matrix, and $\left(\overline{T}_t^+ \overline{J}_{st}\right)^+$ is the pseudo-inverse matrix. If $\left(\overline{T}_t^+ \overline{J}_{st}\right) \in \mathbb{R}^{6\times n}$ is row full rank, $\left(\overline{T}_t^+ \overline{J}_{st}\right)\left(\overline{T}_t^+ \overline{J}_{st}\right)^+ = I$ holds. Then, by substituting Eq. (3.30) into Eq. (3.24), the closed-loop system becomes

$$(\dot{x}_t - \dot{x}_{t_d}) + K_p(x_t - x_{t_d}) = 0, \tag{3.31}$$

which yields $x_t \to x_{t_d}$.

The system response can be calculated by combining the controlled system ($\overline{T}_t \dot{x}_t = \overline{J}_{st}\dot{\theta}_d$) and the system kinematics in Eq. (3.14). The system kinematics with feedback control can be described as

$$\begin{bmatrix} \overline{T}_t & 0 \\ \overline{T}_t & -\overline{J}_{st} \end{bmatrix} \begin{bmatrix} \dot{x}_t \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \overline{J}_{st} \\ 0 \end{bmatrix} \dot{\theta}_d. \tag{3.32}$$

3.6.1.4 Cost evaluation process

Fig. 3.5 shows the workspace evaluation procedure. We first choose a desired trajectory $x_{t_d}(s)$ to cover a region of interest, as described in section 3.6.1.2. To adjust the speed of the desired trajectory, we introduce the adjusting parameter α and set $s = \alpha t$. The parameter α should be

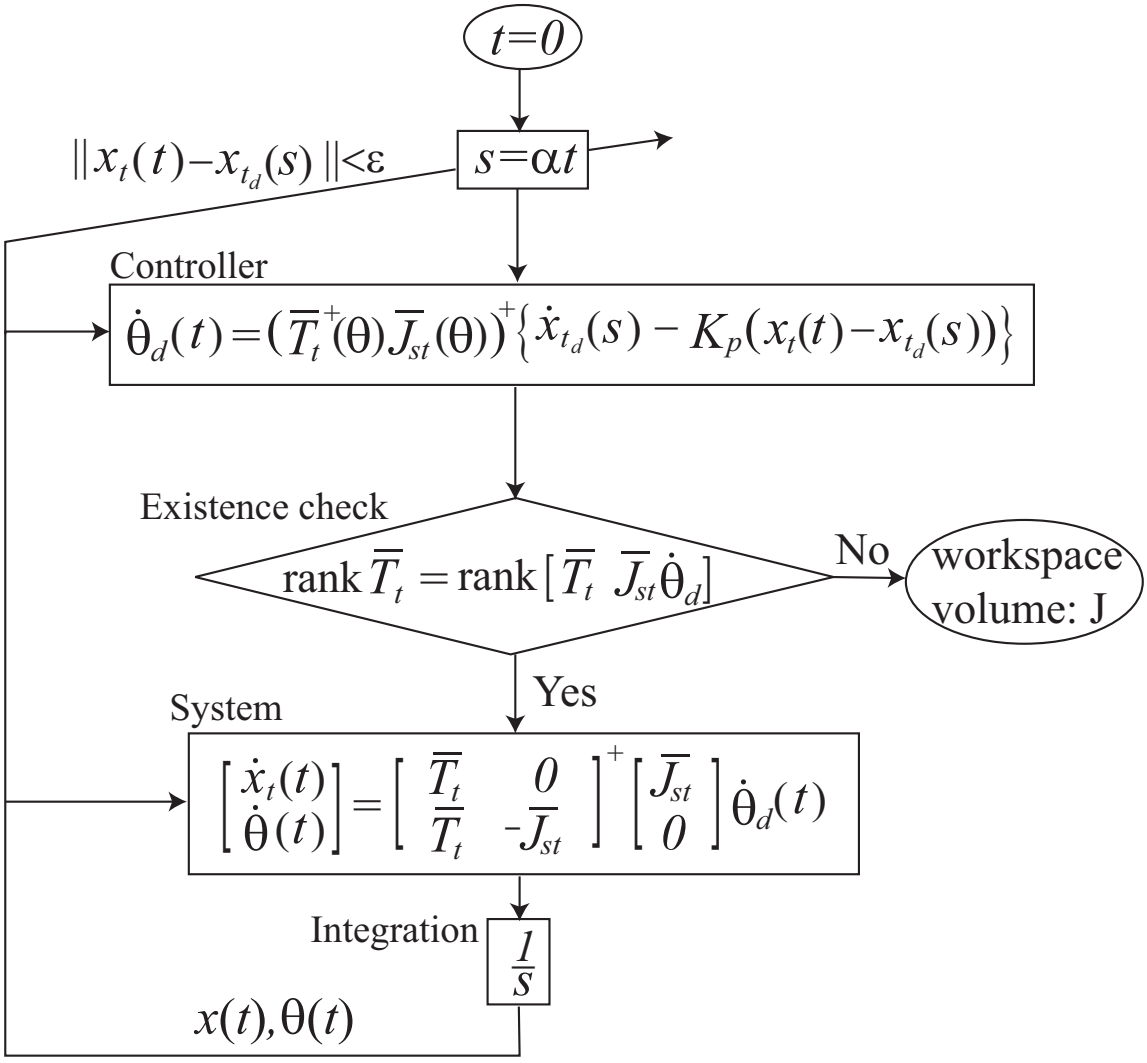


Figure 3.5: Cost evaluation by kinematics control (robotic platform)

sufficiently small for the controlled system to follow $x_{t_d}(s)$. The control input $\dot{\theta}_d$ is calculated using Eq. (3.30), and the solution existence condition in Eq. (3.23) is validated. If this condition holds, a numerical integration for Eq. (3.32) is performed and the time evolution is continued. If the condition does not hold, the tool frame reaches the workspace boundary, the simulation is terminated, and the workspace volume J is calculated from the area covered by $x_{t_d}(s)$ until this point. The trajectory speed parameter α can be adjusted by observing the tracking error $\|x_t - x_{t_d}\| < \varepsilon$ if necessary. Note that the position-level kinematics for θ do not need to be solved, as it is calculated using the numerical integration of Eq. (3.32).

We can choose any costs (manipulability, joint movement range, etc.) other than the workspace volume while guaranteeing the existence condition in Eq. (3.23) by evaluating the cost during the simulation.

3.6.2 Cost evaluation by dynamics control

In some cases, we want to evaluate the costs involving the dynamic response including the input torque and the forces interacting with the environment. These costs can be evaluated by conducting the dynamical simulation with feedback control instead of kinematics control. This section illustrates this process by taking an example of the grasping/manipulation system.

3.6.2.1 Basic idea

As described in section 2.5, the dynamics of the grasping/manipulation system is given by Eq. (2.21)–(2.23). For grasping and manipulation control, the following are assumed:

(A1) $\overline{J}_h \in \mathbb{R}^{n\times n}$ in Eq. (2.71) is nonsingular.

(A2) For any F_o, there exists f_N to make F_f satisfy Eq. (2.90).

The constraint equation in Eq. (2.71) is a linear equation of $\dot{\theta}$. Therefore, the condition (A1) ensures that there exists a joint motion $\dot{\theta}$ satisfying the constraint for any object motion $\dot{\overline{x}}_t$. By contrast, the condition (A2) ensures that there exists an internal force f_N to regulate the fingertip force F_f inside the friction cone for any manipulating force F_o. These assumptions correspond to the manipulable and force closure conditions in the field [8].

We can evaluate the dynamic response of the grasping and manipulation system by controlling the augmented object motion $\overline{x}_o$ and the internal force f_N. The condition (A1) can be used to evaluate the workspace for the grasping/manipulation system.

3.6.2.2 Controller and controlled system dynamics

The system expression for grasping and manipulation is given by Eq. (2.97). As described in section 2.1.2, a variety of tracking controllers can be designed, and a simple choice is a linearizing compensator with PID control [33], that is,

$$\tau = \tau_d = \overline{M}_o u_o + \overline{C}_o \dot{\overline{x}}_o + \overline{N}_o + J_h^{\mathrm{T}} K_G u_{f_N}, \tag{3.33}$$

where

$$u_o = \ddot{\overline{x}}_{o_d} - K_{d_o}(\dot{\overline{x}}_o - \dot{\overline{x}}_{o_d}) - K_{p_o}(\overline{x}_o - \overline{x}_{o_d}) \tag{3.34}$$

$$u_{f_N} = f_{N_d} - K_{I_f} \int (f_N - f_{N_d}) dt. \tag{3.35}$$

$K_{d_o}, K_{p_o}, K_{I_f} > 0$ are the control gains, and $\overline{x}_{o_d}$ and f_{N_d} denote the desired trajectories of $\overline{x}_o$ and f_N. Note that f_N in Eq. (3.35) can be calculated from F_f by using Eq. (2.96).

Upon substituting Eq. (3.33) with Eq. (3.34) and Eq. (3.35) into Eq. (2.97), the closed loop system becomes

$$(\ddot{\overline{x}}_o - \ddot{\overline{x}}_{o_d}) + K_{d_o}(\dot{\overline{x}}_o - \dot{\overline{x}}_{o_d}) + K_{p_o}(\overline{x}_o - \overline{x}_{o_d}) = 0 \tag{3.36}$$

$$(f_{f_N} - f_{N_d}) + K_{i_f} \int (f_N - f_{N_d}) dt = 0, \tag{3.37}$$

which yields $\overline{x}_o \to \overline{x}_{o_d}$ and $f_N \to f_{N_d}$.

The system response can be calculated by combining Eq. (2.21), Eq. (2.22) with Eq. (2.92), the time derivative of Eq. (2.71), and Eq. (3.33). The system dynamics with feedback control can be described as

$$\underbrace{\begin{bmatrix} M_f & 0 & J_h^{\mathrm{T}} \\ 0 & M_o \widehat{T}_o & -G \\ \overline{J}_h & -\overline{T}_o & 0 \end{bmatrix}}_{\overline{M}} \begin{bmatrix} \ddot{\theta} \\ \ddot{\overline{x}}_o \\ F_f \end{bmatrix} = \underbrace{\begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix}}_{\overline{B}} \tau_d - \underbrace{\begin{bmatrix} C_f \dot{\theta} + N_f \\ C_o \widehat{T}_o \dot{x}_o + N_o + M_o \dot{\widehat{T}}_o \dot{x}_o \\ \dot{\overline{J}}_h \dot{\theta} - \dot{\overline{T}}_o \dot{x}_o \end{bmatrix}}_{\overline{N}} \tag{3.38}$$

3.6.2.3 Cost evaluation process

Like Fig. 3.5 for the kinematics control, Fig. 3.6 shows the evaluation process by using dynamics control. From the region of interest, we first choose the desired trajectories $\overline{x}_{o_d}(s)$ and $f_{N_d}(s)$ according to section 3.6.1.2. The desired trajectory for the internal force magnitude $f_{N_d}(s)$ can be a constant estimated a priori from the object weight. If the manipulable condition (A1) holds, the control input τ_d is calculated by Eq. (3.33) with Eq. (3.34) and Eq. (3.35). If not, the system reaches the boundary of the workspace. By applying the control input, the system response is calculated according to Eq. (3.38), and a numerical integration for the solutions $\ddot{\theta}$ and $\ddot{\overline{x}}_o$ is performed. For the fingertip force solution F_f, the friction cone condition in Eq. (2.90) should be checked. If it holds, the time evolution is continued.

We can choose any cost (work, system mass, etc.) other than the workspace volume while guaranteeing the manipulability and friction cone conditions over the region.

In the next sections, we illustrate the optimization process through two numerical examples. In the first example, we optimize the joint position of a robotic platform to maximize the workspace by using the kinematics con-

trol. In the second example, we optimize the grasping/manipulation system structure (geometry and topology) to minimize the link mass by using the dynamics control. In the following simulation, the Matlab/Simulink commercial package (R2018b, MathWorks Inc., Natick, MA) is used. The parameters for the numerical integration are set to auto with variable step size and the default options (relative error tolerance = 10^{-3}, maximum step = auto, etc.).

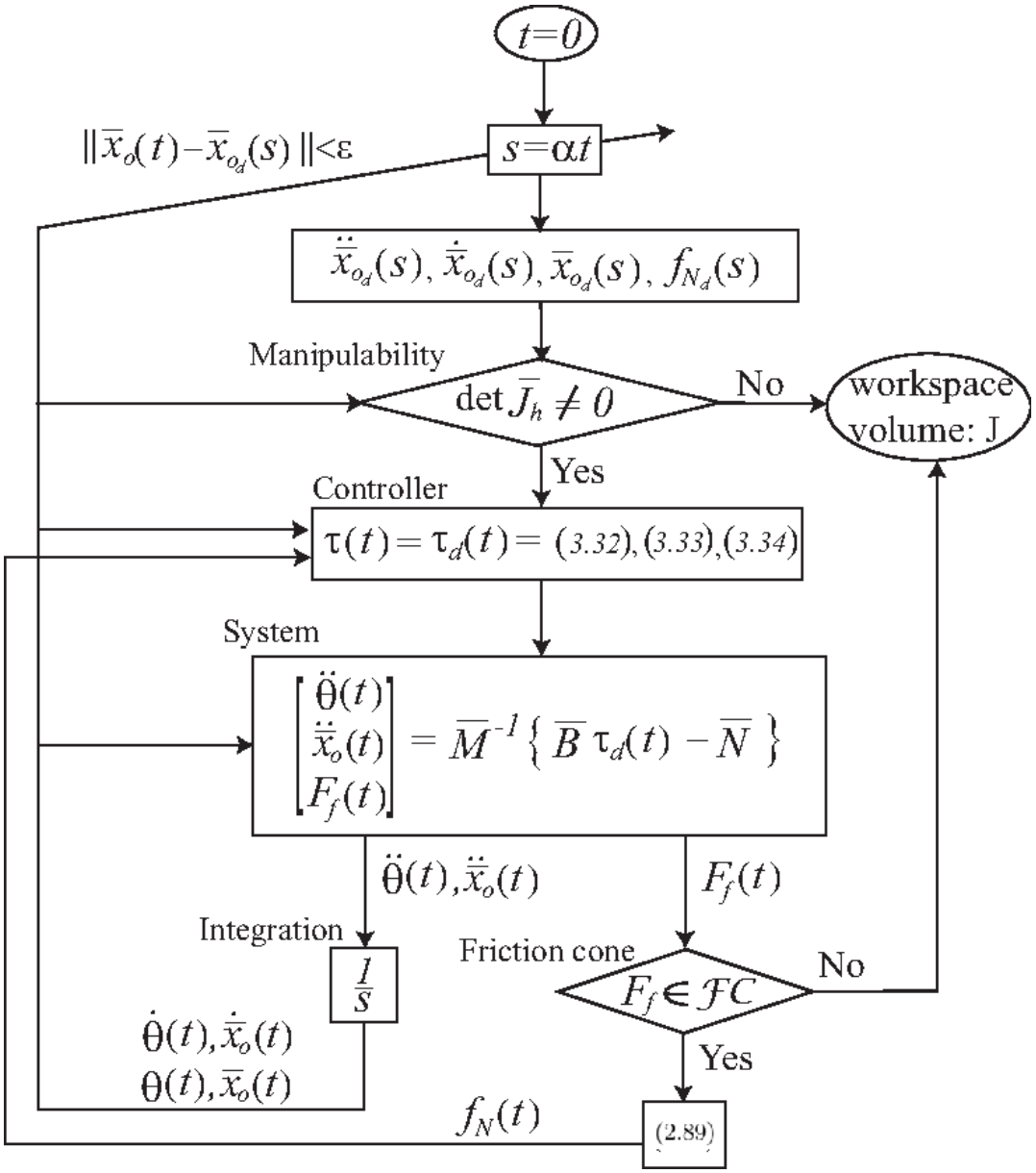


Figure 3.6: Cost evaluation by dynamics control (grasping/manipulation)

3.7 Optimization of robotic platform

In the first example, we consider a simple optimization problem of a robotic platform by using the kinematics control described in section 3.6.1. In the design, the joint position of the base is optimized to maximize the workspace.

3.7.1 Problem settings

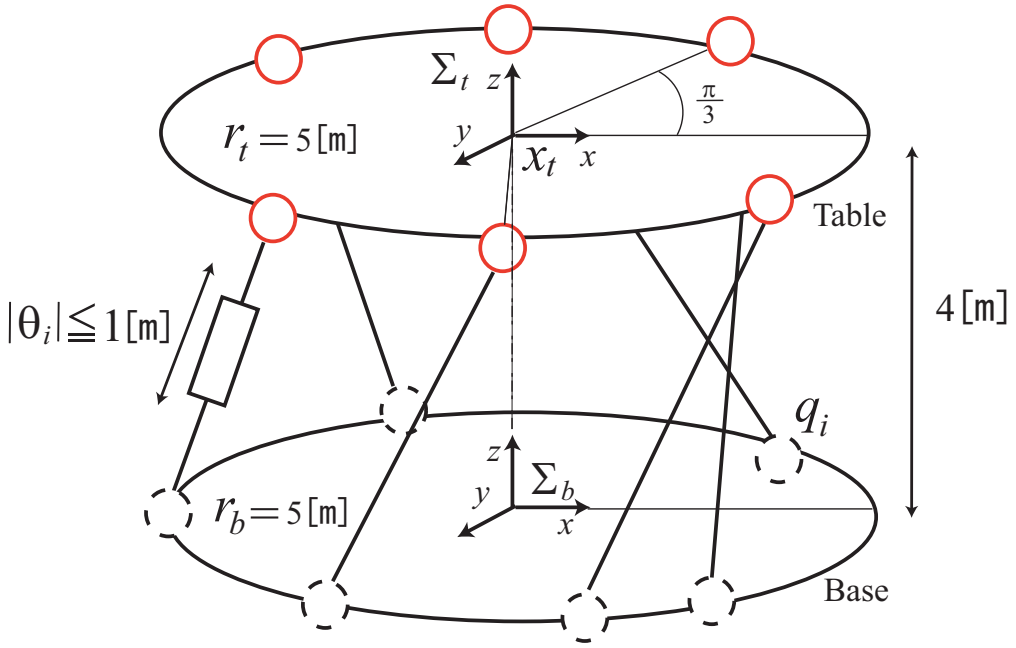


Figure 3.7: Robotic platform system configuration for optimization

For the first design, we consider the robotic platform shown in Fig. 3.7. The system comprises the base, table, and six connecting legs. Each leg has two ball joints on each end and one translational joint at the middle. All the ball joints are passive, and the translational joints are active to be controlled. The table joint position is fixed and is distributed evenly on the rim (interval of $\pi/3$). The base joint position is variable along the rim and is the design parameter. The translational joints have a maximum elongation of $|\theta_i| \leq 1$ [m]. The radius of the base and table is $r_b = r_t = 5$ [m]. The initial position of the table is $x_t = [p_{t_x}, p_{t_y}, p_{t_z} \,|\, \theta_{t_x}, \theta_{t_y}, \theta_{t_z}]^{\mathrm{T}} = [0, 0, 4 \,|\, 0, 0, 0]^{\mathrm{T}}$ as shown in Fig. 3.8(a).

In the GA optimization, we design the base joint position

$$q_i = [r_b \cos\psi_i,\ r_b \sin\psi_i,\ 0]^{\mathrm{T}}, \tag{3.39}$$

that is parametrized by $0 \leq \psi_i \leq 2\pi$ $(i = 1, \cdots, 6)$. Therefore, we in-

troduce the string $s_{q_{i_\psi}} \in \mathbb{B}^{n_{q_{i_\psi}}}$ and set its length (bits) as $n_{q_{i_\psi}} = 6$. The number of the population is set as $N = 50$, and the initial population is produced from random numbers. The adaptation process is conducted for the local-level string $s_{q_{i_\psi}}$ as well as the higher-level sting $s_q = [s_{q_1}, \cdots, s_{q_6}]$. Table 3.1 summarizes the probabilities of the adaptation process. For the reproduction, we use the *roulette wheel selection* [38] strategy, in which we distribute all the fitness-weighted strings onto a wheel and then generate a random number within that wheel to select the next offspring.

For each individual, we conduct a numerical simulation for the kinematics using Eq. (3.32). The controller is given by Eq. (3.30), and the control gain is set as $K_p = 10$. The desired trajectory for the table position is a spiral and that for the table normal is a circle to tilt, as described in the example in section 3.6.1.2. The trajectory parameters are set as $k_1 = 3$, $k_2 = 0.24$, and $h = 4$ for spiral translation and $k_3 = 10$ for tilting rotation. The simulation is stopped if any of the following three conditions fails: solution existence condition in Eq. (3.23); maximum translational joint elongation $|\theta_i| \leq 1$; or no collision between the legs. The fitness (cost) is the simulation duration time T because it is directly correlated with the spiral expansion and thus the workspace.

Table 3.1: GA adaptation probability (platform optimization)

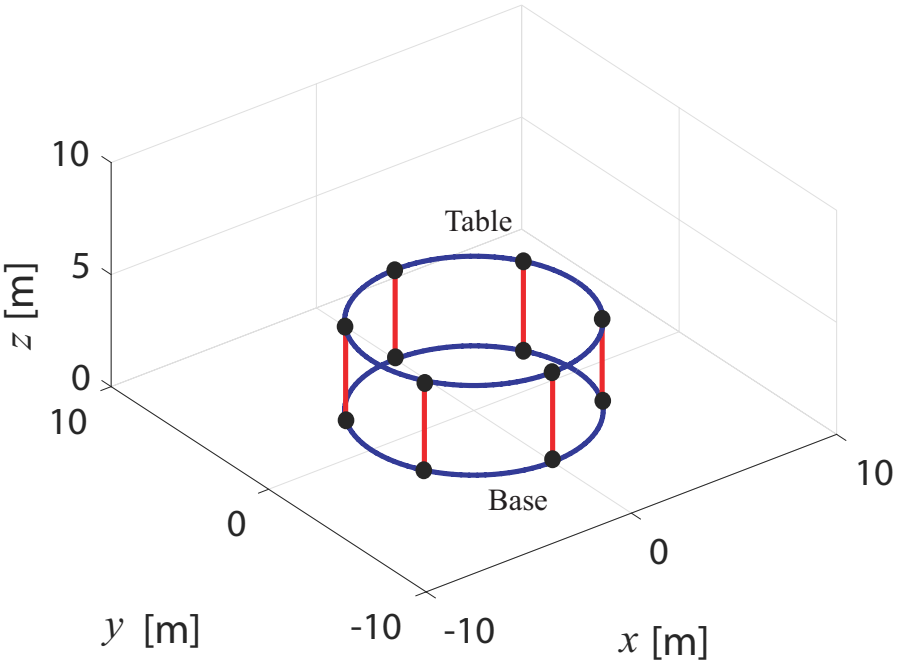
Adaptation	String level	Values
Reproduction	-	Roulette Wheel Selection
	Number of elites	$N_l = 1$
Crossover	Higher level	$p_{c_H} = 0.300$
	Lower level	$p_{c_L} = 0.550$
Mutation	Higher level	$p_{m_H} = 0.0125$
	Lower level	$p_{m_L} = 0.0300$

3.7.2 Optimization Results

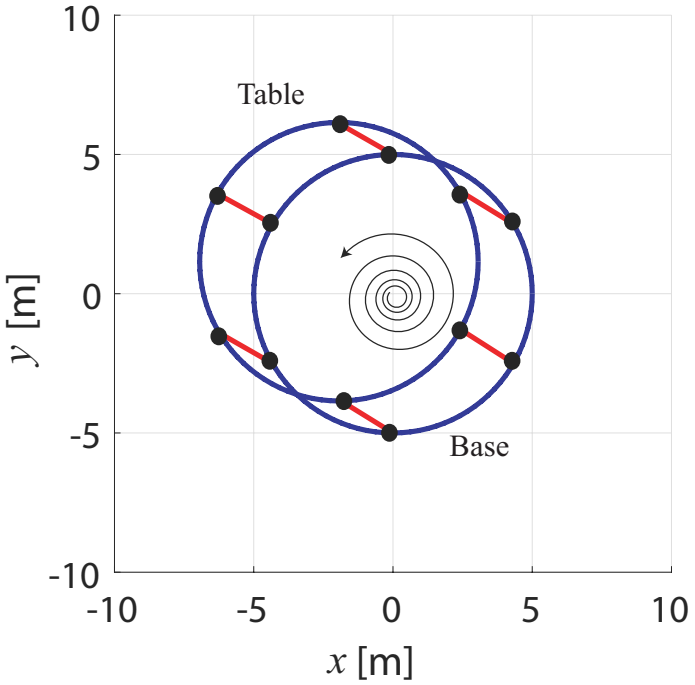
Fig. 3.9 shows the evolution of the best individual fitness. The best time duration increases as the generation proceeds, and the optimal design achieves $T = 10$ [s]. Fig. 3.8(b) shows the spiral trajectory during the time duration (showing the workspace).

Fig. 3.8(a) and (b) respectively show the initial and final configuration of the best individual. The optimal parameters (angle interval) are

$$\psi_1 = \frac{\pi}{6},\ \psi_2 = \frac{3\pi}{6},\ \psi_3 = \frac{5\pi}{6},\ \psi_4 = \frac{7\pi}{6},\ \psi_5 = \frac{9\pi}{6},\ \psi_6 = \frac{11\pi}{6}. \tag{3.40}$$



(a) Initial Position (3D view)



(b) Final Position (upper view)

Figure 3.8: Initial and final configuration (platform optimization)

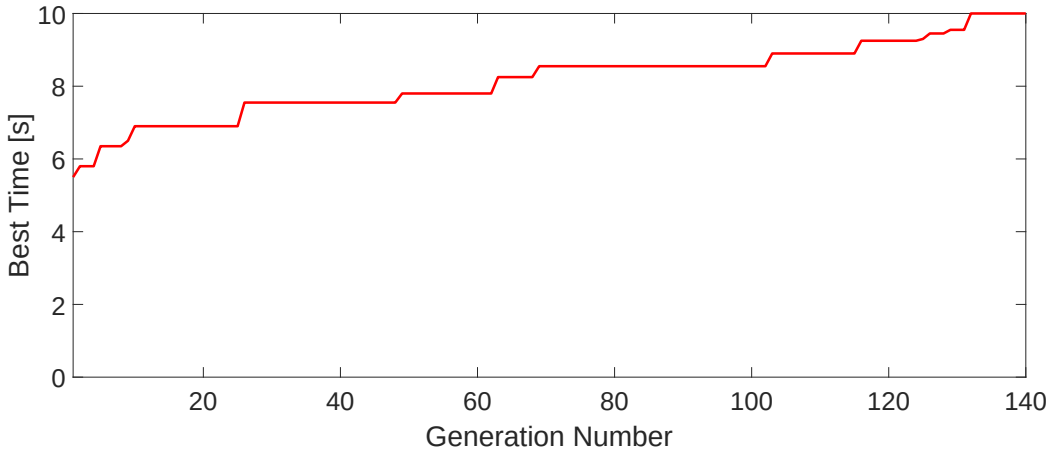


Figure 3.9: Evolution of the best individual (platform optimization)

The results show that the design distributes the base joint evenly along the rim. The joints are located just below the table joints; therefore, the legs go straight up to the table from the base center. This result seems reasonable because we evaluate the workspace by sweeping the spiral region expanding from the center. The parallel vertical-leg configuration is beneficial to utilize the translational joint elongation effectively for the symmetrical circular region.

3.8 Optimization of grasping/manipulation system

In the second example, we consider an optimization problem of the grasping/manipulation system by using the dynamics control described in section 3.6.2. In the design, both the joint position/direction (geometry) and the joint distribution/connection (topology) are optimized simultaneously.

3.8.1 Classical design

Before we proceed to the design optimization, we first conduct a grasping and manipulation simulation using the classical robot hand shown in Fig. 3.10. The physical parameters are shown in the figure.

In this design, the palm is connected on the roof thorough a rotational joint, and three 3-DOF fingers are installed on the edges of the palm ($n = 10$ and $C = 3$). The grasped object is a cube, and the contact points are chosen as in the right-hand-side figure. The initial object position

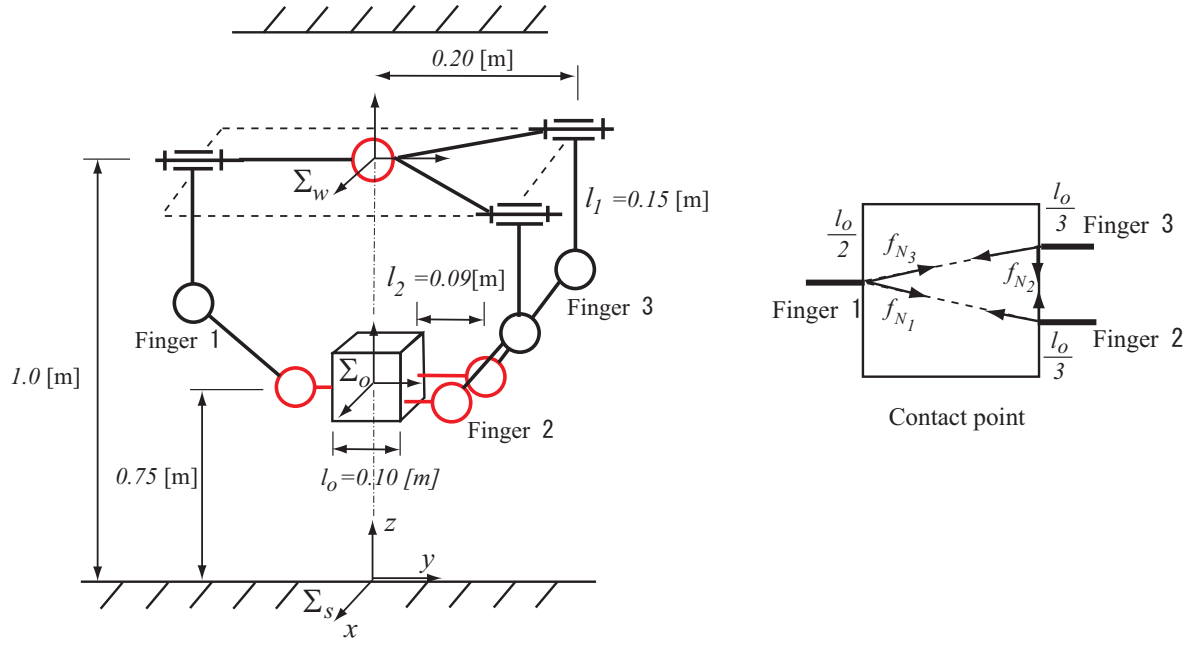
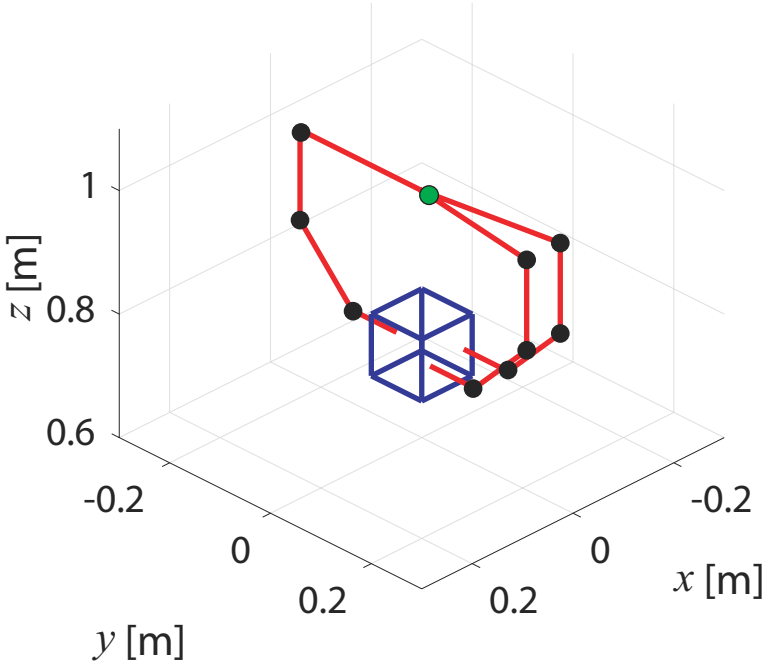


Figure 3.10: Classical robot hand system

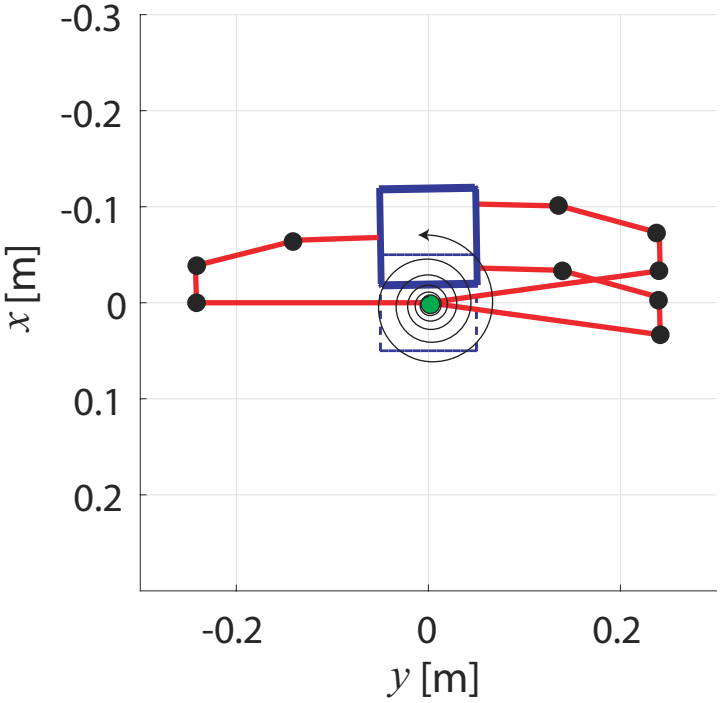
is $x_o = [p_{o_x}, p_{o_y}, p_{o_z} \,|\, \theta_{o_x}, \theta_{o_y}, \theta_{o_z}]^{\mathrm{T}} = [0, 0, 0.8 \,|\, 0, 0, 0]^{\mathrm{T}}$. The system has $n - k = 10 - 9 = 1$ redundancy, and we choose the internal velocity $\dot{x}_n$ for simplicity. We assume that the links have a cylindrical shape with radius of $r = 0.01$ [m] and are made of steel with a mass density of $\rho = 4000$ [kg/m^3]. The total mass $M = \sum \rho \pi r^2 l_{i^c}$ of the links is $M_{ref} = 2.3516$ [kg].

As in the first design example in section 3.7, the desired trajectory for the object position is a spiral and that for the object normal is a circle to tilt. The trajectory parameters are set as $k_1 = 4.7$, $k_2 = 0.006$, and $h = 0.75$ for spiral translation and $k_3 = 10$ for tilting rotation. Fig. 3.11(a) and (b) respectively show the initial and final configurations. Fig. 3.11(b) also shows the translational spiral trajectory. The desired trajectory of the internal motion is set as $x_n \equiv 0$ and that of the internal force magnitude is set as $f_{N_d} = [f_{N_{d_1}}, f_{N_{d_2}}, f_{N_{d_3}}]^{\mathrm{T}} \equiv [10, 3, 10]^{\mathrm{T}}$ (see Fig. 3.10). The controller is given by Eq. (3.33), and the control gains are set as $K_{d_o} = 10 I_9$, $K_{p_o} = 300 I_9$, and $K_{I_f} = 0.01 I_3$. The dynamics of the system is calculated using Eq. (3.38), and during the task, the manipulability and friction cone conditions as well as whether the links collide with the object are checked.

According to the simulation, the classical hand can successfully conduct the task without violating the manipulability and friction cone conditions while avoiding collisions with the object. Fig. 3.12 shows the time history of the object motion x_o. The object motion (blue) is controlled properly to follow the desired trajectory (red).



(a) Initial position (3D view)



(b) Final position (upper view)

Figure 3.11: Initial and final configurations (classical hand)

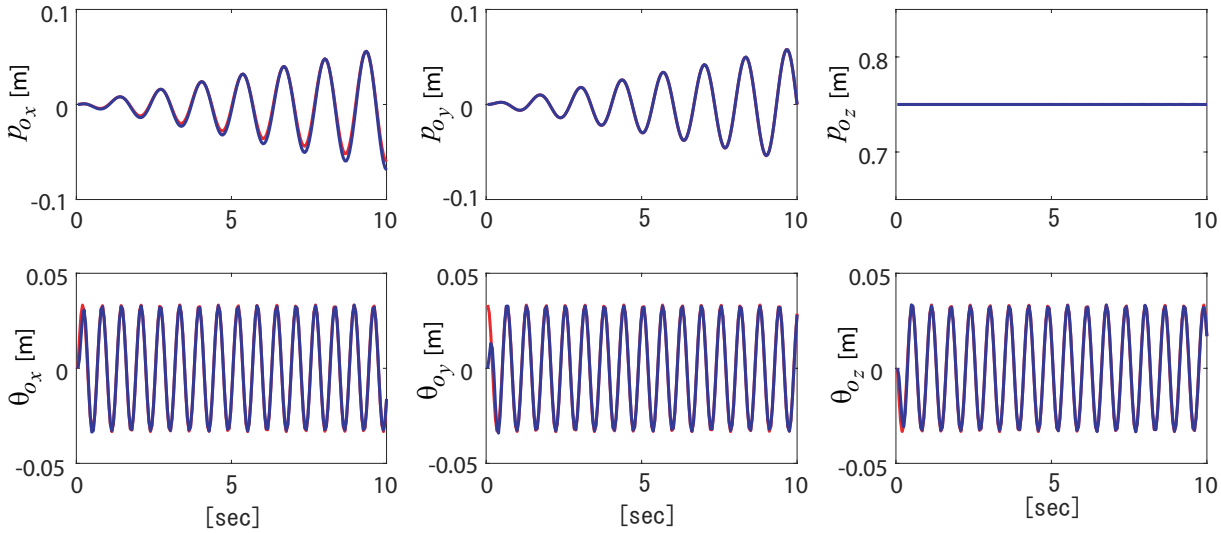


Figure 3.12: Time history of object motion x_o (blue: actual, red: desired)

3.8.2 Optimization settings

Although the classical hand can conduct the task, a more efficient hand can possibly be designed for the given task. Below, we investigate the design of a manipulator that minimizes the total link mass while using the same number of joints $n = 10$ and fingers $C = 3$.

We design the joint position q_i, type (revolutional/translational), and movement direction (v_i or ω_i) as well as their tree-type architecture C_h. The position and orientation of the first joint (wrist) are fixed at their original values. The contact points and positions of the tip joints are also fixed for object grasping. This leaves six free joint positions and nine free joint orientations. Fig. 3.13 shows the possible joint region for the optimization.

The mass is calculated from the link length l_{i^c} as before, that is, $M = \sum \rho\pi r^2 l_{i^c}$. For a revolutional joint, the link length is the distance of the joint positions q_i. As for a translational joint, the length between the joints varies during the task; therefore, we assign a fixed length $l_t = 0.3$ [m] to represent its maximum elongation.

For the GA optimization, we use the strings described in section 3.5. The length (bits) of the strings is set as $n_{q_{ix}} = n_{q_{iy}} = n_{q_{iz}} = 4$; $n_{\phi_i} = n_{\gamma_i} = 3$; and $n_{n_t} = n_{b_1} = n_{b_2} = n_{C_2} = n_{C_3} = 3$. The total size of the string s_j for each individual is 150. The adaptation process is conducted by the different string group levels as described in section 3.5. Table 3.2 summarizes the probability of the adaptation process.

The number of the population is set as $N = 50$. Candidates of the initial

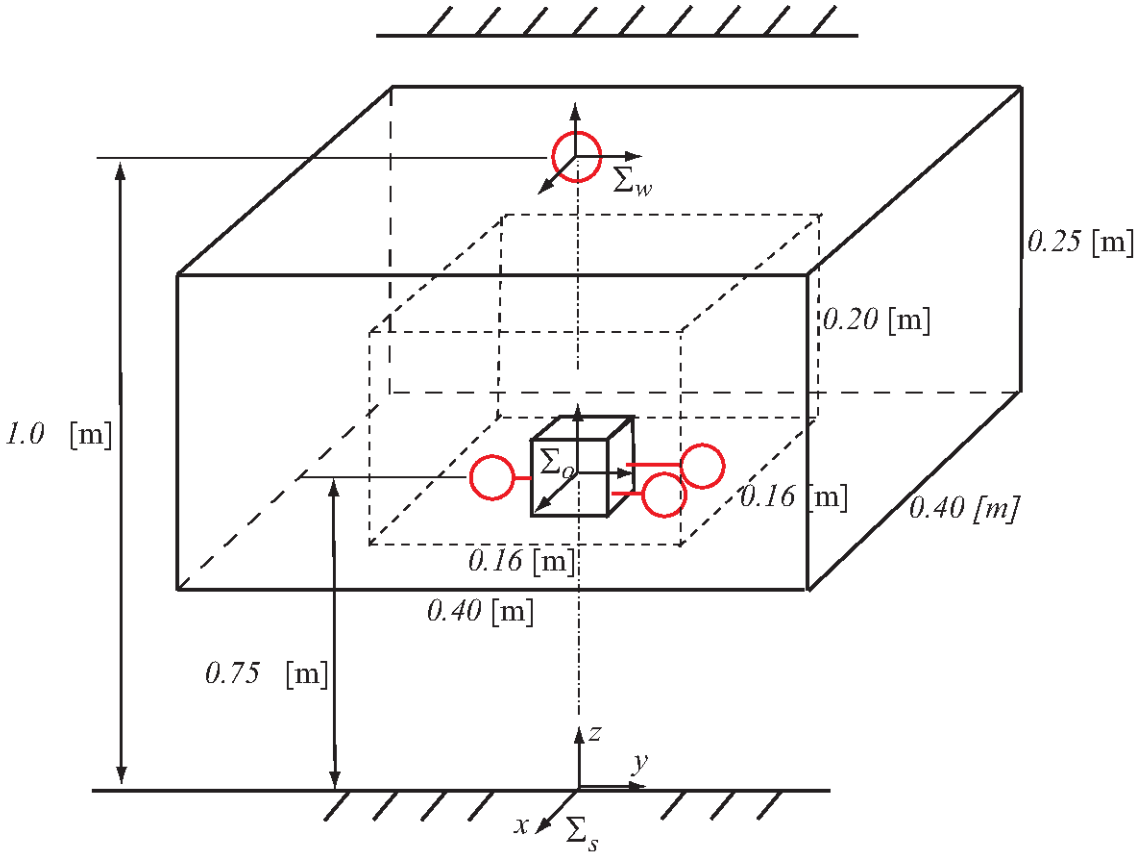


Figure 3.13: Possible joint region for geometric optimization

Table 3.2: GA adaptation probability (grasping/manipulation optimization)

Adaptation	String level	Values
Reproduction	-	Roulette Wheel Selection
	Number of elites	$N_l = 1$
Crossover	Higher level	$p_{c_H} = 0$
	Intermediate level	$p_{c_I} = 0.18$
	Lower level	$p_{c_L} = 0.72$
Mutation	Higher level	$p_{m_H} = 0.0033$
	Intermediate level	$p_{m_I} = 0.0033$
	Lower level	$p_{m_L} = 0.0165$

population are produced from random numbers, and we select the first $N = 50$ individuals that conduct the task successfully without violating the conditions or colliding. In this optimization, to save the computation time and avoid being stuck in local minima, we use a multistep optimization strategy. For a rough search, we first conduct four sets of optimizations up to $G = 50$ generations. Then, for a more precise search, we conduct the final optimization up to $G = 100$ generations by using the ten best individuals from the first optimization along with ten random individuals as the initial population ($N = 50$ in total).

3.8.3 Optimization results

Fig. 3.14 shows the evolution of the best individual in the first optimization. The solid blue line indicates the mass of the classical design, and we have three best individuals below the classical design. Fig. 3.15 shows the best configuration for each optimization, and the corresponding chain matrix is

$$C_{h_1} = \left[\begin{array}{ccc|c|ccccccc|c} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \end{array}\right], \tag{3.41}$$

$$C_{h_2} = \left[\begin{array}{cccc|ccccc|cc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right], \tag{3.42}$$

$$C_{h_3} = \left[\begin{array}{cccc|cc|ccc|c} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right], \tag{3.43}$$

$$C_{h_4} = \left[\begin{array}{ccc|c|ccc|ccc|c} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \end{array}\right]. \tag{3.44}$$

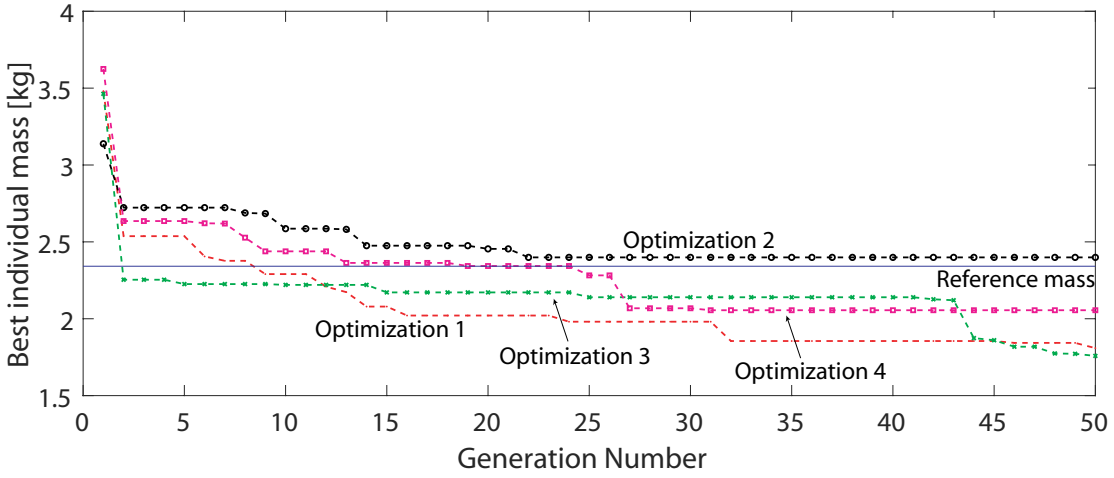
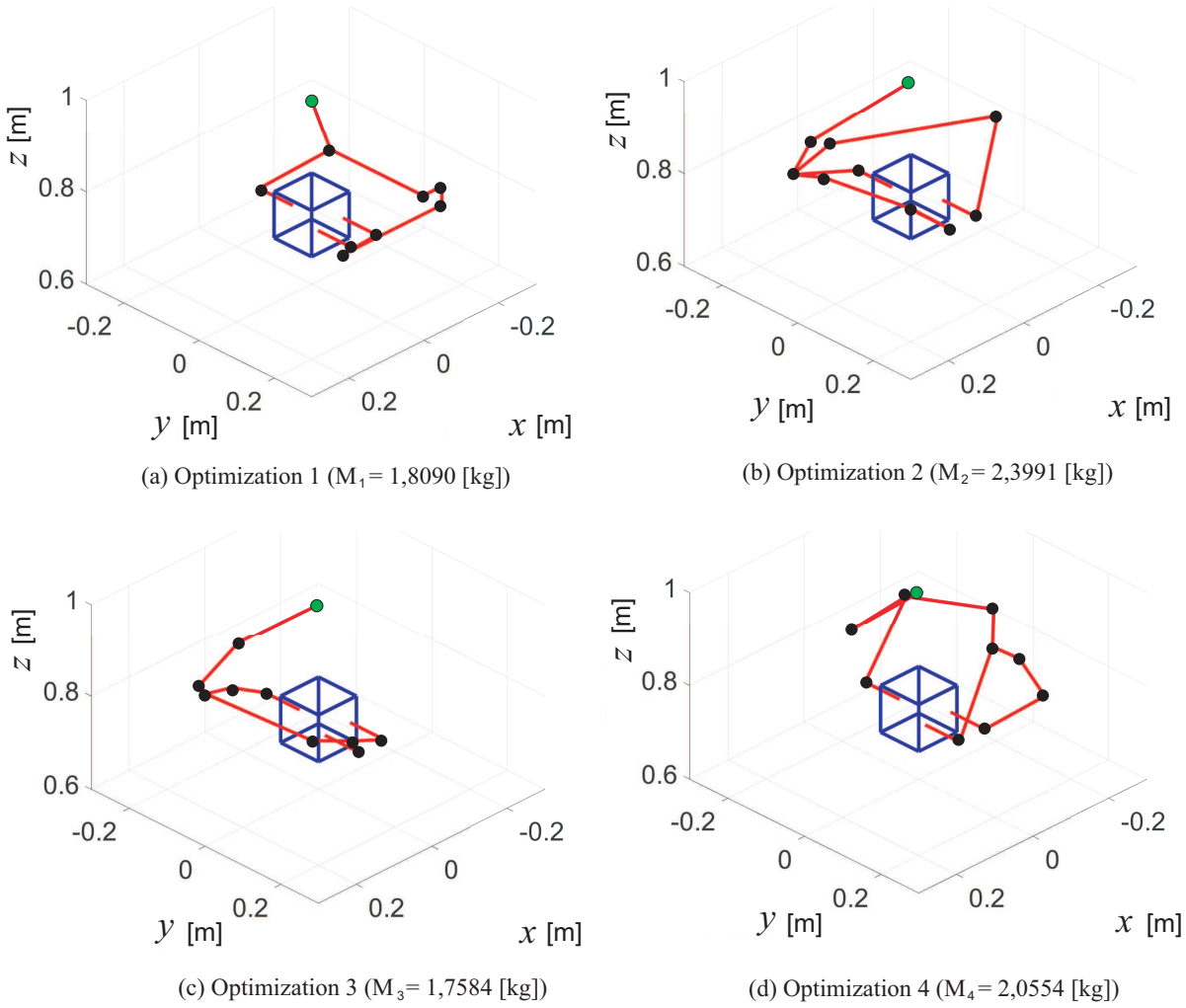


Figure 3.14: Evolution of best individual (first optimization for grasping/manipulation)

By using the best individuals described above, we conduct the final optimization. Fig. 3.16 shows the evolution of the best individual and Fig. 3.17, the best configuration after $G = 100$ generations. The optimal total mass is $M_{opt} = 1.6082$ [kg], which represents a 32% reduction compared with the classical design. Table 3.3 summarizes the exponential parameters



(a) Optimization 1 (M_1= 1,8090 [kg])

(b) Optimization 2 (M_2= 2,3991 [kg])

(c) Optimization 3 (M_3= 1,7584 [kg])

(d) Optimization 4 (M_4= 2,0554 [kg])

Figure 3.15: Configuration of best individual (first optimization for grasping/manipulation)

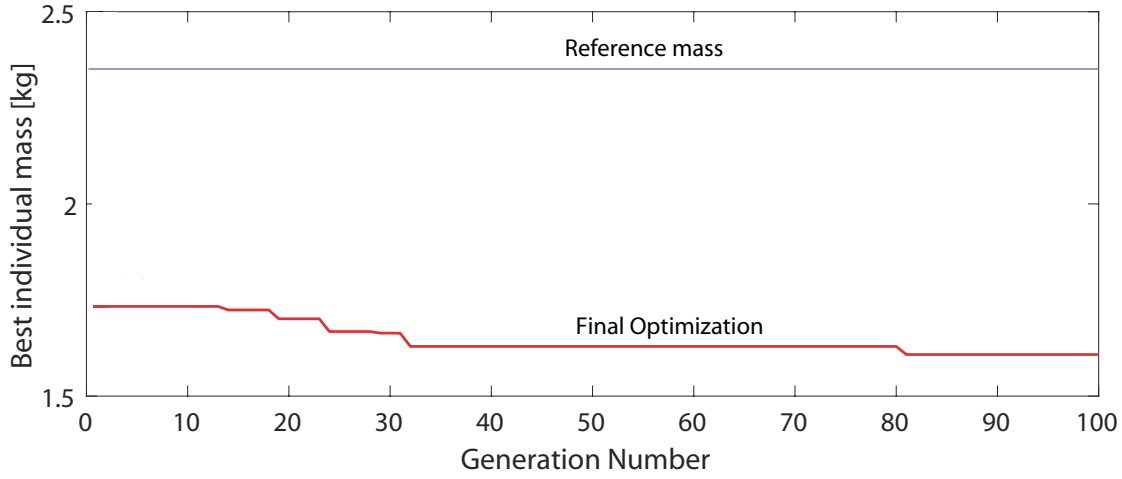
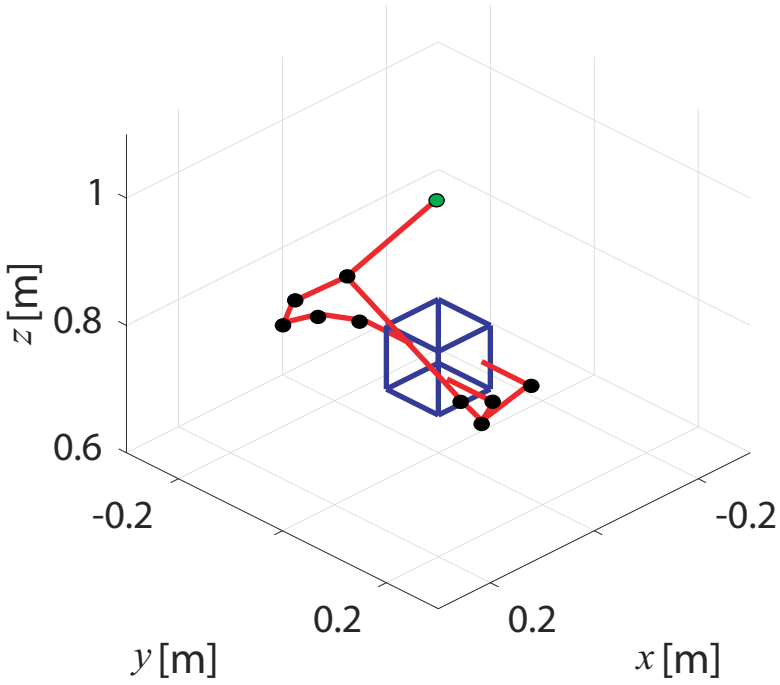
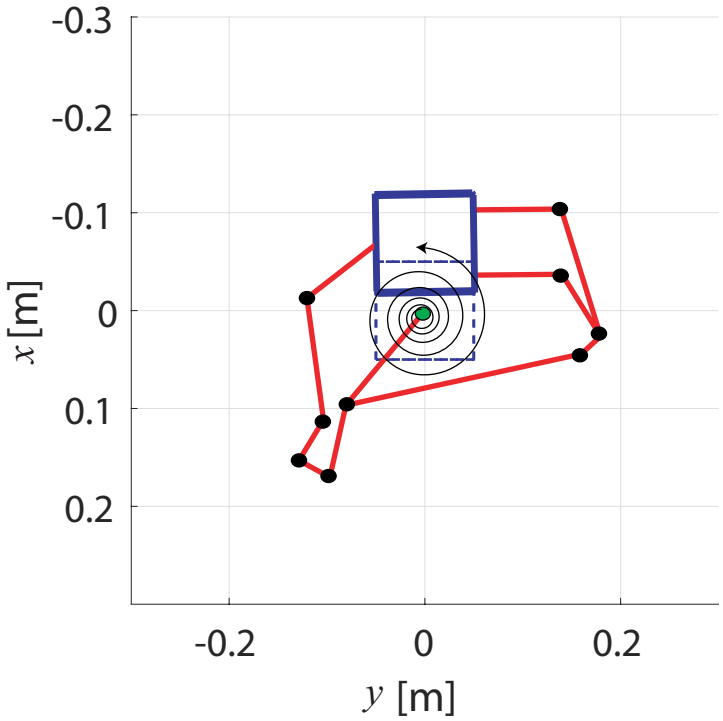


Figure 3.16: Evolution of best individual (final optimization for grasping/manipulation)

(joints 6, 9, and 10 are the fingertip joints and their positions are not the



(a) Initial position (3D view)



(b) Final position (upper view)

Figure 3.17: Configuration of best individual (final optimization for grasping/manipulation)

design parameters), and the chain matrix is

$$C_{h_{opt}} = \left[\begin{array}{cc|cccc|ccc|c} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]. \tag{3.45}$$

Table 3.3: Exponential parameters of arm-hand optimal design

	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Joint position q_i	0.10 −0.08 0.88	0.16 −0.12 0.86	0.15 −0.15 0.80	0.11 −0.12 0.82	(0.00) (−0.14) (0.75)
Joint type t_i (0:rev., 1:trans.)	0	0	0	0	0
Joint direction $\{v_i, \omega_i\}$	0.09 0.06 0.99	0.64 0.77 0.00	1.00 0.00 0.00	−0.02 −0.17 0.98	−0.02 0.11 0.99
	Joint 7	Joint 8	Joint 9	Joint 10	
Joint position q_i	0.11 0.16 0.79	0.10 0.18 0.76	(0.03) (0.14) (0.75)	(−0.03) (0.14) (0.75)	
Joint type t_i (0:rev., 1:trans.)	0	0	0	0	
Joint direction $\{v_i, \omega_i\}$	−0.45 0.00 0.89	0.92 0.27 0.29	−0.29 0.57 0.77	0.40 0.92 0.00	

We note the following about the optimization result:

(1) For joint type selection, all joints are revolutional and no translational joint exists. This is reasonable because the introduction of translational joints automatically increases the mass by the fixed length of $l_t = 0.3$ irrespective of the joint position selection.

(2) For mass reduction, the total length of the link should be shortened while avoiding collisions. For this purpose, the design introduces an elbow (first joint) and locates the wrist or a branching joint (second joint) closer to the object in the lower left part of its original position. The two right-hand-side fingers branch at the end of the tree for the same reason.

(3) For manipulation purposes, the design distributes the joints equally to the left (thumb) and right fingers (index and middle) to accommodate the spiral motion. As a result, the thumb has more DOFs than the other fingers.

3.9 Summary of design optimization using genetic algorithms and exponential coordinates

In this chapter, we proposed an adaptation of the exponential coordinates driven kinematics and dynamics method to the optimization scheme of the genetic algorithms:

- In section 3.2, we derived an extension of the tree-type system motion theories and proposed method to adapt those theories to floating-base systems, closed-chain systems and platform systems.
- In sections 3.3 and 3.4, we developed tools to transform the parameters describing a robotic architecture (q,v,ω and C_h) into binary strings to implement those parameters into a genetic algorithm driven optimization scheme.
- In section 3.6, we proposed cost evaluation procedures using feedback control to judge the fitness of different robotic configuration given a specific task.
- In section 3.7 and 3.8, we used the proposed theories to conduct an optimization of a robotic platform, as well as a variation of the arm-hand system studied in section 2.9, and discussed the obtained results.

As a whole, owing to the new proposed methods, we were able to successfully improve the design of the arm-hand to fit a specific goal by lowering the overall mass of the manipulator system while maintaining the completion of the task at a satisfactory level.

Chapter 4

Conclusion

4.1 Research results

In this thesis, we proposed a new modeling, control, and optimization method for robotic systems design. This method is based on a combination of exponential coordinates and GAs to allow for a very flexible modeling and automatized optimization process of robotic structures.

In Chapter 2, we extended the exponential coordinates method to tree-type systems using the chain matrix. This extension included the derivation of the equation of motion of the manipulator for tree-type systems, the object motion, as well as the constraint equation linking the object and the manipulator through the contact points. We also derived the expression of system redundancy through the equations. Using the exponential parameters and the chain matrix, we derived an automated process to design such systems using only a few parameters: the exponential parameters describing the joint characteristics in the system (joint position, orientation, and type) and the chain matrix describing the system architecture (the connection between joints). Finally, we proposed control strategies to comply with the manipulation requirement by allowing for the simultaneous control of the grasped object position and orientation and the redundancy present in the system, which can be used freely (for example, to control some of the joint rotation without object motion changes). These theories were applied to a well-known example of the industry, the combination of a robotic arm and a robotic hand: the arm-hand system, and the results were found to be conclusive.

In Chapter 3, we extended the exponential coordinates even further, and we proposed schemes to derive equations of motion in the case of floating-base systems, closed-loop systems, and platform systems. Then, we proposed a conversion scheme of the exponential parameters and the chain matrix into binary strings to allow for a simultaneous geometry/topology optimization scheme based on GAs. Within this optimization, we proposed a cost evaluation of the simulated design by kinematics and dynamical control for specific tasks. Finally, we tested the optimization scheme on the previously designed tree-type system and managed to improve its design by reducing the mass of the system for possible applications into embedded systems.

4.2 Comments on results

While developing the proposed new tools for robotic design, we observed several advantages and drawbacks of the proposed method. The advantages of the proposed method are listed below.

- Exponential parameters are very versatile, allowing for near infinite design possibilities for robotic systems. In addition, they are independent, which means that replacing parameters one at a time is an easy task and does not affect the system as a whole. They also allow for a closed form of the kinematics and dynamics, combined with the chain matrix.
- The chain matrix allows for a new look on the description of tree-type systems, by describing the dividing kinematic paths within the system. Combined with the exponential parameters, it allows for simultaneous or global geometrical and topological optimization.
- With the proposed method of binary coding of exponential parameters, it is possible to assign different precisions for each parameters for reducing computational effort, thereby allowing a flexible optimization adapted to the user's need.
- GAs allow for an immense research space that can allow for very interesting design solutions that were difficult to achieve until now.

The drawbacks of the proposed method are listed below.

- Kinematic and dynamic equations of motion using the exponential coordinates are obtained by matrix computation for control design purpose, however, it is a rather computation-heavy method. This means that the on-line control of such systems with this method can be challenging.
- GAs are known for their random search method, so while they are allowing for a tremendous size of the research space. The speed at which they converge to the best solution is also random, thereby it may result in long optimization time.
- GAs are known to suffer from the local minima problem. While we employed methods to overcome these phenomena (the division of optimization parameters in several levels), the risk still exists and it can take time for the algorithms to drop out of those minimas.

In conclusion, the proposed method is very flexible and can cover an extremely high number of possible designs; however, the method is time con-

suming; therefore it may be more efficiently used in optimization schemes rather than on-line computation.

4.3 Further prospects and improvement avenues

The designs and optimizations shown in this thesis have been showing the potential of design and optimization schemes based on exponential parameters. Nevertheless, some improvements can be made both on the modeling and control part and the optimization part:

- Most of robotic systems nowadays are implementing more and more springs and dampers, which are not included in our automated method. While it is possible to implement them directly to the equations, a general implementation method could further the design, and thus optimization possibilities.
- The manipulation aspect in our study has been kept simple for design purposes (only three contact points, single point contact for each fingertip, ...), however it is possible to include more general manipulation condition such as several contact points on each chain or multiple contacts.
- We studied the extension of those modeling and control theories for closed-chains, but a chain matrix describing these connections has not been derived. It could be interesting to implement, as it could be linked with the proposed optimization method with only small changes.
- For floating-base systems as well, time-varying constraints with the environment were not considered in this thesis. As most humanoid robots are in contact with the ground, a generalized expression of those constraint varying with time is the natural next step to generalized humanoid design. Additionally, this method could allow for an overall control scheme managing simultaneously the humanoid movements as well as the grasping of a potentially hold object, which will bring humanoid closer to human behaviour.
- In order to improve the overall computation time, changing the matrix computation into vector computation might allow for on-line control schemes.
- It could be interesting to combine exponential parameters with other evolutionary algorithms schemes such as particle swarm optimization

or adaptive firefly algorithms, to compare the results and aim for a faster overall convergence to the solutions.

Overall, the priority will be given to the addition of new parameters involved, since it would allow for a bigger improvement of the results obtained in the optimization part, which is the main appeal of the proposed method.

Acknowledgments

This thesis has been has been realized in Wakayama University through a PhD. in robotic design.

First, I would like to thank the Wakayama University for accepting me and providing the environment necessary to complete this thesis, as well as helping me on my daily life as a foreign student in Japan.

I would like to thank my PhD. advisor, Prof. Kenji Nagase, for helping me with the realization of this thesis, for guiding me trough my research experiences and to help me improve as a researcher as well as a person.

I would also like to thank Lecturer Hiroki Dobashi for helping me and providing me the tools to overcome several challenges I faced trough my thesis.

Additionally, I would like to thank the different students of the system control laboratory for always being friendly towards me and guiding me trough the lifestyle in Japan.

Finally, I would like to thank my family, especially my mother Ms. Sonia Meunier for supporting my project going abroad, as well as my friends, especially Mr. Daisuke Fujita for always being there in times of need.

Bibliography

[1] K. H. Petersen, N. Napp, R. Stuart-Smith, D. Rus, and M. Kovac. A review of collective robotic construction. *Science Robotics*, 4(28):eaau8479, 2019.

[2] D. Stewart. A platform with six degrees of freedom. *Proceedings of the Institution of Mechanical Engineers*, 180(1):371–386, 1965.

[3] H. Masato and O. Kenichi. Honda humanoid robots development. *Philosophical Transactions of the Royal Society A*, 365:11–19, 2006.

[4] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcarde, B. Marnier, J. Serre, and B. Maisonnier. Mechatronic design of nao humanoid. *2009 IEEE International Conference on Robotics and Automation*, pages 769–774, 2009.

[5] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson. Optimization based full body control for the atlas robot. *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 120–127, 2014.

[6] R. Sparrow. Kicking a robot dog. *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, page 229, 2016.

[7] J. J. Craig. *Introduction to Robotics: Mechanics and Control, Third Edition*. Pearson Prentice Hall, 2005.

[8] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[9] M. W. Spong, S. Hutchinson, and M. Vidiasagar. *Robot Modeling and Control*. John Wiley and Sun, 2006.

[10] B. K. Rout and R. K. Mittal. Tolerance design of robot parameters using taguchi method. *Mechanical Systems and Signal Processing*, 20(8):1832–1852, 2006.

[11] W. Li, Z. Li, Y. Liu, L. Ding, J. Wang, H. Gao, and Z. Deng. Semi-autonomous bilateral teleoperation of six-wheeled robot on soft terrains. *Mechanical Systems and Signal Processing*, 133:106234, 2019.

[12] R. J. Alattas, S. Patel, and T. M. Sobh. Evolutionary modular robotics: Survey and analysis. *Journal of Intelligent & Robotic Systems*, 95:815–828, 2019.

[13] V. Patidar and R. Tiwari. Survey of robotic arm and parameters. In Coimbatore, editor, *International conference on computer communication and informatics (ICCCI)*, pages 1–6, 2016.

[14] S. Saeedvand, M. Jafari, H. S. Aghdasi, and J. Baltes. A comprehensive

survey on humanoid robot development. *The knowledge engineering review*, 34:E20, 2019.

[15] T. Fujikawa, K. Hirakawa, S. Okuma, T. Udagawa, S. Nakano, and K. Kikuchi. Development of a small flapping robot: Motion analysis during takeoff by numerical simulation and experiment. *Mechanical Systems and Signal Processing*, 22(6):1304–1315, 2008.

[16] K. J. Kaliński and M. Mazur. Optimal control of 2-wheeled mobile robot at energy performance index. *Mechanical Systems and Signal Processing*, 70-71:373–386, 2016.

[17] D. J. Montana. The kinematics of contact and grasp. *The International Journal of Robotics Research*, 7(3):17–31, 1988.

[18] Z. Li, P. Hsu, and S. Sastry. Grasping and coordinated manipulation by a multifingered robot hand. *The international Journal of Robotics Research*, 8(4):33–50, 1989.

[19] N. Brook, M. Shoham, and J. Dayan. Controllability of grasps and manipulations in multi-fingered hands. *IEEE Transactions on Robotics and Automation*, 14(1):185–190, 1998.

[20] M. Zribi, J. Chen, and M. S. Mahmoud. Coordination and control of multi-fingered robot hands with rolling and sliding contact. *Journal of Intelligent and Robotic Systems*, 24:125–149, 1999.

[21] S. Arimoto, Z. Doulgeri, P. T. A. Nguyen, and J. Fasoulas. Stable pinching by a pair of robot fingers with soft tips under the effect of gravity. *Robotica*, 20:241–249, 2002.

[22] T. Yoshikawa. Manipulating and grasping forces in manipulation by multifingered robotic hand. *IEEE Transaction on Robotics and Automation*, 7(1):67–77, 1991.

[23] T. Yoshikawa. Multifingered robot hands: Control for grasping and manipulation. *Annual Reviews in Control*, 34(2):199–208, 2010.

[24] T. Wimbock, C. Ott, A. Albu-Schaffer, and G. Hirzinger. Comparison of object-level grasp controllers for dynamic dexterous manipulation. *The International Journal of Robotics Research*, 31(1):3–23, 2011.

[25] T. Watanabe, K. Yamazaki, and Y. Yokokohiji. Survey of robotic manipulation studies intending practial applications in real environment-object recognition, soft robot hand, challenge program and benchmarking. *Advanced Robotics*, 31(19-20):1114–1132, 2017.

[26] K. Nagai and T. Yoshikawa. Impedance control of redundant macro micro manipulators. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 94)*, 2:1439–1445, 1994.

[27] K. Nagai, S. Iwasa, Y. Nakagawa, and K. Ohno. Development of a redundant macro-micro manipulator for compliant motion. pages 707–712, 1997.

[28] S. V. Shah, S. K. Saha, and J. K. Dutt. *Dynamics of Tree-Type Robotics Systems*, volume 62 of *Intelligent Systems, Control and Automation: Science and Engineering*. Springer, 2013.

[29] R. Motro. *Tensegrity: Structural Systems for the Future*. Kogan Page Science, 2003.

[30] R. E. Skelton and M. C. de Oliveira. *Tensegrity Systems*. Springer, 2009.

[31] J. Y. Zhang and M. Ohsaki. *Tensegrity Structures, Form, Stability, and Symmetry*, volume 6 of *Mathematics for Industry*. Springer, 2015.

[32] K. Nagase and R. E. Skelton. Double-Helix tensegrity structures. *AIAA Journal*, 53(4):847–862, 2015.

[33] J. Amar and K. Nagase. A unified framework for dynamics and control of tree-type systems using exponential coordinates. *Mechanical Systems and Signal Processing*, 131:446–468, 2019.

[34] K. Kikuchi, K. Sakaguchi, T. Sudo, N. Bushida, Y. Chiba, and Y. Asai. A study on a wheel-based stair-climbing robot with a hopping mechanism. *Mechanical Systems and Signal Processing*, 22(6):1316–1326, 2008.

[35] A. Odry, R. Fullér, I. J. Rudas, and P. Odry. Kalman filter for mobile-robot attitude estimation: Novel optimized and adaptive solutions. *Mechanical Systems and Signal Processing*, 110:569–589, 2018.

[36] K. Cai, Y. Tian, X. Liu, S. Fatikow, F. Wang, L. Cui, D. Zhang, and B. Shirinzadeh. Modeling and controller design of a 6-DOF precision positioning system. *Mechanical Systems and Signal Processing*, 104:536–555, 2018.

[37] J. Deng, W. Chen, Y. Wang, S. Zhang, and Y. Liu. Modeling and experimental evaluations of a four-legged stepper rotary precision piezoelectric stage. *Mechanical Systems and Signal Processing*, 132:153–167, 2019.

[38] D. E. Goldberg. *Genetic Algorithms in search, Optimization and Machine Learning*. Addison Wesley, 1990.

[39] C. H. Liu and C. H. Chiu. Optimal design of a soft robotic gripper with high mechanical advantage for grasping irregular objects. pages 2846–2851, 2017.

[40] E. Bjørlykhaug and O. Egeland. Mechanical design optimization of

a 6DOF serial manipulator using genetic algorithm. *IEEE Acces*, 6:59087–59095, 2018.

[41] B. Paden and S. Sastry. Optimal kinematic design of 6R manipulators. *The International Journal of Robotics Research*, 7(2):43–61, 1988.

[42] F. Pierrot, V. Nabat, O. Company, S. Krut, and P. Poignet. Optimal design of a 4-DOF parallel manipulator: From academia to industry. *IEEE Transactions on Robotics*, 25(2):213–224, 2009.

[43] D. Chablat, S. Venkateswaran, and F. Boyer. Mechanical design optimization of a piping inspection robot. *Procedia CIRP*, 70:307–312, 2018.

[44] S. Rajappa, M. Ryll, H. H. Bülthoff, and A. Franchi. Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4006–4013, 2015.

[45] S. Hwang, H. Kim, Y. Choi, K. Shin, and C. Han. Design optimization method for 7 DOF robot manipulator using performance indices. *International Journal of Precision Engineering and Manufacturing*, 18:293–299, 2017.

[46] P. I. Corke. A simple and systematic approach to assigning denavit hartenberg parameters. *IEEE Transactions on Robotics*, 23(3):590–594, 2007.

[47] G. Gao, G. Sun, J. Na, Y. Guo, and X. Wu. Structural parameter identification for 6 DOF industrial robots. *Mechanical Systems and Signal Processing*, 113:145–155, 2018.

[48] S. Singh and E. Singla. Realization of task-based designs involving DH parameters: a modular approach. *Intelligent Service Robotics*, 9:289–296, 2016.

[49] G. Zeinoun, R. Sedaghati, and F. Aghili. Optimal design parameters of reconfigurable robots with lockable joints. *Transactions of the Canadian Society for Mechanical Engineering*, 41(1):23–38, 2017.

[50] A. Csiszar. A combinatorial optimization approach to the Denavit-Hartenberg parameter assignment. pages 1451–1456, 2015.

[51] S. F. P. Saramago and V. Jr Steffen. Optimization of the trajectory planning of robot manipulators taking into account the dynamics of the system. *Mechanism and Machine Theory*, 33(7):883–894, 1998.

[52] S. Ha, S. Coros, A. Alspach, J. Kim, and K. Yamane. Computational co-optimization of design parameters and motion trajectories for robotic systems. *The International Journal of Robotics Research*,

37(13-14):1521–1536, 2018.

[53] M. Smith. *Applications of Dual Quaternions in Three Dimensional Transformation and Interpolation*. PhD thesis, University of Canterbury, 2013.

[54] X. Yang, H. Wu, Y. Li, and B. Chen. A dual quaternion solution to the forward kinematics of a class of six-dof parallel robots with full or reductant actuation. *Mechanism and Machine Theory*, 107:27–36, 2017.

[55] A. Valverde and P. Tsiotras. Spacecraft robot kinematics using dual quaternions. *Robotics*, 7(4):64, 2018.

[56] D. R. Isenberg and Y. P. Kakad. Quaternion based computed-torque and feed-forward tracking controllers for a space robot. pages 232–236, 2010.

[57] Y. Li, X. Yang, H. Wu, and B. Chen. Optimal design of a six-axis vibration isolator via stewart platform by using homogeneous jacobian matrix formulation based on dual quaternions. *Journal of Mechanical Science and Technology*, 32:11–19, 2018.

[58] B. Yin, Z. Liang, X. Dai, J. Mo, and S. Wang. Task-oriented configuration optimization of a lattice distortable reconfigurable robot. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 230(9):1532–1543, 2016.

[59] F. C. Park. Optimal robot design and differential geometry. *Journal of Mechanical Design*, 117(B):87–92, 1995.

[60] E. J. Van Henten, D. A. Van't Slot, C. W. J. Hol, and G. L. Van Willigenburg. Optimal manipulator design for a cucumber harvesting robot. *Computers and Electronics in Agriculture*, 65(2):247–257, 2009.

[61] C. Zheng, Y. Zhang, J. Li, J. Bai, X. Qin, and B. Eynard. Survey on design approaches for robotic manufacturing systems in SMEs. *Procedia CIRP*, 84:16–21, 2019.

[62] L. Zhou, Y. Li, and S. Bai. A human-centered design optimization approach for robotic exoskeletons through biomechanical simulation. *Robotics and Autonomous Systems*, 91:337–347, 2017.

[63] J. Fu and F. Gao. Optimal design of a 3-leg 6-DOF parallel manipulator for a specific workspace. *Chinese Journal of Mechanical Engineering*, 29:659–668, 2016.

[64] R. Datta, S. Pradhan, and B. Bhattacharya. Analysis and design optimization of a robotic gripper using multiobjective genetic algorithm. *IEEE Transactions on Systems, Man and Cybernetics: Sys-*

tems, 46(1):16–26, 2016.

[65] K. F. Man, K. S. Tang, and S. Kwong. Genetic algorithms: Concepts and applications. *IEEE Transactions on Industrial Electronics*, 43(5):519–534, 1996.

[66] D. J. Schaffer, D. Whitley, and L. J. Eshelman. Combinations of genetic algorithms and neural networks: A survey of the state of the art. *International Workshop on Combinations of Genetic Algorithms and Neural Networks*, pages 1–37, 1992.

[67] K. Saitou C.D. Chapman and M. J. Jakiela. Genetic algorithms as an approach to configuration and topology design. *Journal of Mechanical Design*, 116(4):1005–1012, 1994.

[68] A. Baykasoglu and F. B. Ozsoydan. Adaptive firefly algorithm with chaos for mechanical design optimization problems. *Applied Soft Computing*, 36:152–164, 2015.

[69] J. Angeles. *Rational kinematics*. Springer, 1988.

[70] H. Asada and J. J. Slotine. *Robot analysis and control*. John Wiley & Sons, 1986.

[71] A. R. Yildiz, H. Adberazek, and S. Mirjalili. A comparative study of recent non-traditional methods for mechanical design optimization. *Archives of Computational Methods in Engineering*, 27:1031–1048, 2020.

[72] G. Corriveau, R. Guilbault, and A. Tahan. Genetic algorithms and finite element coupling for mehanical optimization. *Advances in Engineering Software*, 41(3):422–426, 2010.

[73] I. C. Trelea. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Information Processing Letters*, 85(6):317–325, 2003.

[74] B. Akay and D. Karaboga. A modified artificial bee colony algorithm for real-parameter optimization. *Information Sciences*, 192:120–42, 2012.

[75] D. Karaboga and B. Akay. A comparative study of artificial bee colony algorithm. *Applied Mathematics and Computation*, 214(1):108–132, 2009.

[76] N. J. Cheung, X. M. Ding, and H. B. Shen. Adaptive firefly algorithm: Parameter analysis and its application. *PLoS ONE*, 9(11):e112634, 2014.

[77] J. Kim, S. H. Lee, and F. C. Park. Kinematic and dynamic modeling of spherical joints using exponential coordinates. *Proceedings of the*

Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 228(10):1777–1785, 2014.

[78] D. Corus and P. S. Oliveto. Standard steady state genetic algorithms can hillclimb faster than mutation-only evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 22(5):720–732, 2018.

[79] S. H. Chung and H. K. Chan. A two-level genetic algorithm to determine production frequencies for economic lot scheduling problem. *IEEE Transactions on Industrial Electronics*, 59(1):611–619, 2012.

[80] B. Dasgupta and T. S. Mruthyunjaya. A Newton-Euler formulation for the inverse dynamics of the Stewart platform manipulator. *Mechanisms and Machine Theory*, 33(8):1135–1152, 1998.

[81] O. Masory and J. Wang. Workspace evaluation of Stewart platforms. *Advanced Robotics*, 9(4):443–461, 1994.