# Motion Planning of Flexible One-dimensional Objects and Hyper-redundant Robots

A Thesis

Submitted for the Degree of

**Doctor of Philosophy**

in the **Faculty of Engineering**

by

**Midhun Sreekumar Menon**


Mechanical Engineering
Indian Institute of Science
Bangalore – 560 012 (INDIA)

MAY 2016

DEDICATED TO

*My family, friends, advisor and god*

*who have been with me all along and at all times*

# CERTIFICATE

I hereby certify that the content embodied in this thesis titled **Motion Planning of Flexible One-dimensional Objects and Hyper-redundant Robots** has been carried out by Mr. Midhun Sreekumar Menon at the Indian Institute of Science, Bangalore under my supervision and that it has not been submitted elsewhere for the award of any degree or diploma.

*Signature of the Thesis Supervisor*: .....

Professor Ashitava Ghosal
Dept. of Mechanical Engineering
Indian Institute of Science, Bangalore

# DECLARATION

I hereby declare that the content embodied in this thesis titled **Motion Planning of Flexible One-dimensional Objects and Hyper-redundant Robots** is the research carried out by me at the Department of Mechanical Engineering, Indian Institute of Science, Bangalore under the supervision Prof. Ashitava Ghosal, Department of Mechanical Engineering, IISc. In keeping with the general practice in reporting scientific observations, due acknowledgement has been made wherever the work described is based on the findings of other investigations.

*Signature of the Author*: .....

Midhun Sreekumar Menon
Dept. of Mechanical Engineering
Indian Institute of Science, Bangalore

# Acknowledgements

This thesis and the articles in this thesis are based on and have only a single author. But of course this does not mean that I could have written them without the help of others.

I would first like to express my deepest gratitude to the Department of Mechanical Engineering, Indian Institute of Science for all the help during this research work. I would like to acknowledge and thank my advisor Prof. Ashitava Ghosal for his guidance, help and encouragement. His technical expertise in the field of robotics has helped me through critical junctures throughout the period of my research work.

I also would like to thank Prof. G. K. Ananthasuresh and Prof. B. Gurumoorthy for providing valuable feedback and advice for portions of my thesis, without which I couldn't have completed my work in a timely manner. I would like to extend gratitude to Mr. V.C. Ravi for his valuable cooperation and help in fabrication the prototype at Center for Artificial Intelligence and Robotics (CAIR), DRDO, Bangalore. I would like to thank Robert Bosch Center for Cyber-Physical Systems (RBCCPS) for providing me with partial funding for developing the prototype hardware and supporting me partly for attending conferences to present my work to peers. I would like to thank my lab mates for their valuable support and company. I wish to thank my family who has been supporting me all along.

I would also like to thank Donald Knuth and Leslie Lamport for their roles in developing TEX and LATEX. Without these tools it would have been almost impossible to write this thesis. Finally let me thank you, the reader. Without you this thesis would be rather pointless.

# Abstract

This thesis deals with motion planning of flexible one-dimensional objects and hyper-redundant serial robots moving in a plane or in three dimensional space. The flexible one-dimensional object is modeled as a continuous curve and a point on the curve is given a prescribed displacement. The key problem studied in the thesis is to obtain the motion of all points on the curve for the prescribed displacement subject to the condition of the length of the curve being preserved. Such motions are motivated by the need to model, analyze and realistically render of motion of hair, ropes and, more recently, flexible endoscopes where the assumption of constant axial length is realistic and reasonable. In this thesis, the discretized form of the flexible one-dimensional object is related to hyper-redundant robots and motion planning for such robots are obtained when the robot moves in free space and in a cluttered environment, avoiding obstacles.

The motion planning of flexible one-dimensional objects is posed as an optimization problem with constraints and calculus of variation is employed to derive general analytical results. The first analytical result is that, for a given motion of a point on the curve and subject to the preservation of the length of the curve, the infinitesimal motion of any other point on the curve is minimized when the velocity vector at that point of the curve is along the tangent to the curve at that point. This leads to the second key result that when one end of a straight line segment is moved along a straight line, the velocity of the distal (far) end is minimized when it is along the straight line segment and the curve traced by the distal end is the well-known tractrix curve whose closed-form analytic expressions can be obtained using hyperbolic functions. If the flexible one-dimensional object is discretized by several piece-wise straight line segments, the magnitude of the velocity vector of the distal end of the segments attenuates as one goes away from the end where the input displacement is provided and if the direction of the input displacement is not changed, all the line segments eventually line up along the direction of the input displacement. It is shown that the attenuating and eventual aligning features lead to realistic and a more natural motion of the discretized segments and results in the establishment of a $\mathcal{O}(n)$ algorithm for motion planning. It is shown that the developed algorithm can be

used for real-time simulation of the motion of discretized flexible one-dimensional objects and hyper-redundant serial robots.

For realistic simulation and rendering of the motion, the flexible object must be discretized into a large number of straight segments. In the second part of the thesis, the flexible one-dimensional object is represented by a spline and motion planning algorithm is applied to the segments of the underlying control polygon of the spline. Since the number of segments in a control polygon can be significantly less, a significant increase in efficiency in simulation and rendering of the motion is obtained. However, it is known that as the control polygon is moved, the length of the spline curve changes. To overcome this problem, an innovative adaptive algorithm, involving sub-division and merging of the segments of the control polygon, is presented and this restricts the variation in the length of the curve to within a user prescribed tolerance. New analytical results related to the length of the curve and the angle between the adjacent segments of the control polygon are derived for quadratic and cubic splines and, depending on the prescribed tolerance, threshold values of the angle are obtained and used in the algorithm for approximate length preservation.

The last part of the thesis deals with development of a planar hyper-redundant robot and implementation of motion planning algorithm on this robot. The hyper-redundant robot contains 12 links connected by actuated rotary joints which can change the angle between the links in a controlled manner. The links are on the wheels which provide support and allow it to move forward. The leading link also has a DC motor which can rotate the wheels so that it can move forward and pull the trailing links. Using the motion planning algorithm, for a prescribed motion of the leading link, the angle between two successive links are computed. These are given as input to the robot and the path traced by the 12 link robot is observed. It is seen that the motion of the hyper-redundant robot has the expected natural and realistic motion characteristics. It is furthermore demonstrated that the calculus of variation based approach for motion planning can be extended to include obstacle avoidance by adding additional constraints related to the location and size of the obstacles. It is shown that the entire robot optimally avoids the obstacles and moves in a more natural and realistic way.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## Motivation

Motion planning of flexible one-dimensional objects has been of continuing interest and an active area of research in the geometric modeling, computer aided design (CAD) and robotics community. The original motivation was the requirement of efficient simulation and realism in rendering and display of the motion of cables, ropes, chains, hair, snakes etc. in computer graphics, animation and gaming industry. In the recent past, there is an increased interest in building realistic surgical simulators for minimally invasive surgery where motion of the endoscope, catheters, sutures and other flexible one-dimensional (1D) objects is required. In typical physics based simulations of such flexible 1D objects, the flexible 1D object is modeled as rigid links connected by springs and dampers. These approaches suffer from the disadvantage of difficulty in choosing the spring, damping and inertia parameters and difficulty in numerical simulation of large number of ordinary differential equations which model the flexible 1D object. Furthermore the length of the flexible 1D object is typically not preserved and as a result the motion appears less realistic. In the robotics community, motion planning and simulation of snake, elephant trunk and other hyper-redundant robots with large number of links and actuated joints, in presence of obstacles, has been a continuing and active area of research. The key problem in these robots is how to resolve the redundancy and choose one of the infinite number of possible joint solutions for a given motion of the end or any part of the robot. This thesis deals with motion planning of flexible 1D objects and hyper-redundant robots. The aim of this work is to develop efficient algorithms for real-time and natural looking motions of flexible 1D objects and hyper-redundant robots and their implementation demonstrating efficient simulation, rendering in case of flexible 1D objects and validation with experiments in the case of hyper-redundant robots.

There exists extensive literature on modeling, simulation and motion planning of flexible one-dimensional objects and hyper-redundant robots. The approach in this thesis is based on optimization of appropriate functionals using the powerful tool of calculus of variations. The flexible 1D object is modeled as a finite length curve in three-dimensional space, a motion is prescribed at one end or any one point on the curve and depending on a chosen objective function, the motion of the entire curve is obtained. We present new analytical results and extensive numerical simulation results to illustrate the developed algorithms. When the flexible 1D object is discretized with rigid links connected by joints, it is shown that the developed algorithms can be used for motion planning of hyper-redundant robots. It is shown that a more natural looking motion can be realized efficiently and, furthermore, such natural motions can be obtained with obstacle avoidance.

Since this work deals with flexible 1D objects and hyper-redundant robots, in the rest of the chapter, we present existing relevant literature in these two areas separately. We end the chapter with contributions of this work and the organization of the thesis.

## Modeling & simulation of flexible one-dimensional objects

Flexible geometries are modeled as deformable curves (1D), surfaces (2D) and solids (3D). Hence, as far as this thesis is concerned, the question of representation of flexible bodies boils down to representation of curves and surfaces. We restrict our discussion to curves as deformable surfaces and solids are not considered in this work.

Two of the most common methods used for representing curves are implicit equations and parametric functions[1]. An implicit representation for a curve $\mathcal{C} \in \mathbb{R}^n,\ n = 2, 3$ describes the set of points $\boldsymbol{P}$ lying on the curve in terms of relations between the coordinates of points as

$$\mathcal{C} = \{\boldsymbol{P} \in \mathbb{R}^n | f(\boldsymbol{P}) = K\},\ \text{where } K \text{ is a constant.} \tag{1.1}$$

A simple example is the equation of circle in $\mathbb{R}^2$ given by $x^2 + y^2 = R^2$. In parametric form, a curve is represented by expressing the coordinates of the points as functions of an independent parameter $t$. For example, a helix in $\mathbb{R}^3$ has a parametric representation $\boldsymbol{P} = \{\cos t, \sin t, t\},\ t \in (-\infty, \infty)$. It may be noted that the parametric representation for a curve is not unique and also the curve may have both an implicit and parametric representation. Both representations have their own advantages and disadvantages. The main advantages of a parametric representation are ease of representing bounded curves in 3D space, possessing a natural direction of traversal and extensive amount of available tools. In this work we use parametric representations of curves. Two of the commonly used parametric representations are the Bézier and B-spline and

these are discussed briefly next.

- **Bézier Curve**: An $n^{th}$ degree Bézier curve is defined by

$$\mathfrak{C}(u) = \sum_{i=0}^{n} B_{i,n}(u)\boldsymbol{P}_i,\ 0 \leq u \leq 1 \tag{1.2}$$

The basis(blending) functions $\{B_{i,n}(u)\}$, are the well known $n^{th}$ degree Bernstein polynomials[2] given by

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!}u^i(1-u)^{n-i} \tag{1.3}$$

The geometric coefficients of this form, $\{\boldsymbol{P}_i\}$ are called control points and the polygon formed by $\{\boldsymbol{P}_0, \boldsymbol{P}_1, \boldsymbol{P}_2, \ldots, \boldsymbol{P}_n\}$, called the control polygon, approximates the shape of the curve rather nicely. Bézier curves are more suitable for interactive curve design because the curve passes through $\boldsymbol{P}_0$ and $\boldsymbol{P}_2$ and the tangent directions to the curve at its endpoints are parallel to $\boldsymbol{P}_1 - \boldsymbol{P}_0$ and $\boldsymbol{P}_n - \boldsymbol{P}_{n-1}$. Moreover, the curve is restricted within the convex hull of the $n$ defining control points, where $n$ is the degree of the curve.

Many important geometric entities, such as circles, cannot be precisely represented using polynomials. However, as it is also well know that all conic curves, including circles, can be represented using rational functions (ratio of two polynomials) and an $n^{th}$-degree rational Bézier curve is defined as

$$\boldsymbol{C}(u) = \sum_{i=0}^{n} R_{i,n}(u)\boldsymbol{P}_i, \quad 0 \leq u \leq 1 \tag{1.4}$$

$$\text{where}\ \ R_{i,n} = \frac{B_{i,n}(u)w_i}{\sum_{j=0}^{n} B_{j,n}(u)w_j} \tag{1.5}$$

The symbols $\boldsymbol{P}_i = (x_i, y_i, z_i)$ and $B_{i,n}(u)$ are as before and the $w_i$ are positive scalars called the weights, $R_{i,n}(u)$ are the rational basis functions for this curve form and $\boldsymbol{P}_i's$ are the control points.. It is clear from Figure 1.1 that the general shape of the curve follows intuitively from the control polygon which is the polygon formed from the control points.

- **B-Splines:** Even though the Bézier curves have many attractive properties making it suitable for digitized representation of curves/surfaces, it has some shortcomings namely,
  - a high degree is required to satisfy large number of constraints. For e.g.,$(n-1)^{th}$

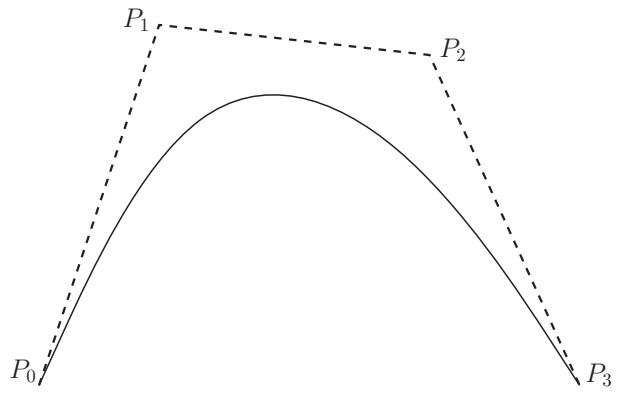



Figure 1.1: Cubic Bézier curve

degree curve/basis functions are needed to pass a polynomial Bézier curve through $n$ data points, and it is well known that higher the degree, more inefficient it is to process them and they are numerically unstable;

- a high degree curve is required to fit complex shapes;
- although Bézier curves can be shaped via the control polygon (and weights), the control is not sufficiently local.

In order to circumvent these issues, piece wise polynomials, or piece wise rational functions are used, mapping different polynomial segments across specified parameter intervals defined via breakpoints. The piece wise segments are constructed so that they join with some level of continuity (not necessarily same at all every break point). Hence, we have the B-spline definition for a $p-$degree interpolation function as follows.

$$\boldsymbol{C}(u) = \sum_{i=0}^{n} f_{i,p}(u)\boldsymbol{P}_i \tag{1.6}$$

where the $\boldsymbol{P}_i$ are the control points and the $\{f_i(u),\ i = 0, \ldots, n\}$ are piece wise polynomial functions forming a basis for the vector space of all piece wise polynomial function of the desired degree and continuity (for a fixed break point sequence, $U = \{u_i\}, 0 \leq i \leq m$). Continuity is determined by the basis functions and $\{f_i\}$ is expected to have all the *usual* nice analytic properties mentioned earlier in this section.

The B-spline basis functions are defined by various methods including divided differences of truncated power functions[3, 4], blossoming[5], and by a recurrence formula[6, 7, 8]. In this thesis, we will use the recurrence formula method for all computations.

Given the breakpoints as $U = \{u_0, \ldots, u_m\}$, a non-decreasing sequence of real numbers

($u_i$ are called knots and $U$, the knot vector), the $i^{th}$ B-spline basis functions of degree $p$ (order $p+1$), denoted by $N_{i,p}(u)$, is defined as follows.

$$\boldsymbol{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\boldsymbol{P}_i, \quad u_0 \leq u \leq u_m \tag{1.7}$$

where, (1.8)

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \leq u < u_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{1.9}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \tag{1.10}$$

It is to be noted that breakpoints correspond to the set of distinct knot values, and the knot spans of nonzero length define the individual polynomial segments. It may be noted here that multiplicity of knots and control points can all reduce the continuity of the curve locally. For example, a knot $u_i \in U$ with multiplicity $k \leq p$ (degree of curve) will have only $C^{p-k}$ continuity at the point on curve corresponding to $u_i$. This is due to discontinuities induced in the interpolation functions by such multiplicities. Similar effects can also be produced by duplication of control points.

Bézier curve is a special case of B-spline of degree $p$ with knot vector of the form

$$U = \{ \underbrace{0, \ldots, 0}_{(p+1) \text{ times}}, \underbrace{1, \ldots, 1}_{(p+1) \text{ times}} \}. \tag{1.11}$$

In other words, the cubic Bézier curve in Figure 1.1 is a cubic B-spline with knot vector $U = \{0, 0, 0, 0, 1, 1, 1, 1\}$. Further, Figure 1.2 shows the effect which degree of interpolation has on the curve. Clearly, as the degree of interpolation functions increase, the curve get more and more distant from the control polygon and starts getting flatter due to larger region of influence/interpolation.

In this thesis, we use B-spline framework for modeling the curve because it is highly flexible, in the sense that it spans both continuous and piece wise continuous curves. Hence, it can easily represent a wide variety of curves. One has full and independent control over the degree and the knot vector to be used to approximate the curve. Moreover, there exist many numerically robust spline algorithms for refining and coarsening the defining control polygon of a spline, which is one of the key ideas employed in this thesis. Last but not the least, extension of this idea to NURBS (Non-Uniform Rational B-Splines), which is a generalization of B-splines, is

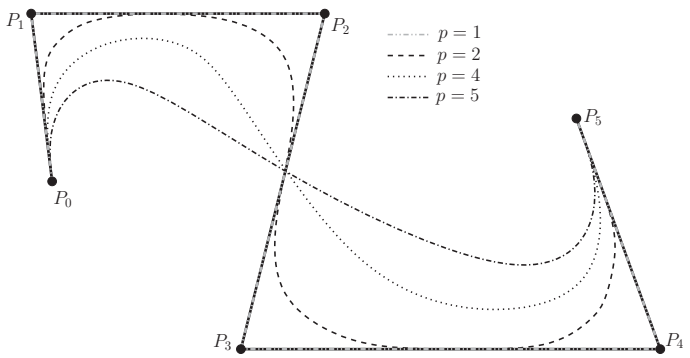



Figure 1.2: B-spline with clamped ends and varying degrees

easy and straightforward.

One of the key idea in this work is length-preserving transformation. In differential geometry, the length of a curve is defined as the limiting length of a polygonal line inscribed in the curve (i.e., with vertices lying on the curve) as the maximum length of the chords forming the polygonal line goes to zero. If this limit exists and is finite, the curve is said to be rectifiable[9]. For a given generic 2D rectifiable curve, with a readily available parametric representation as in Equation 1.6 (B-spline, Bézier, NURBS etc.), the elemental curve (also called as elemental arc) length can be calculated as

$$ds = \lim_{\delta x, \delta y \to 0} \sqrt{\delta x^2 + \delta y^2} = \sqrt{x'^2 + y'^2} du, \text{ where } x' = \frac{dx}{du}, y' = \frac{dy}{du} \tag{1.12}$$

The length can be obtained by integration as

$$s = \int_{u_0}^{u_m} ds = \int_{u_0}^{u_m} \sqrt{x'^2 + y'^2} du \tag{1.13}$$

The original motivation for simulation of one dimensional flexible objects was the requirement of realism in simulation and display of the motion of cables, ropes, chains, hair, snakes, cloth etc. in computer graphics and animation industry and there exists a large amount of literature on hair/cloth simulation[10]. In recent past, there is renewed interest in real-time motion planning of flexible one-dimensional objects driven by the need to build simulators for laparoscopy, endoscopy and in the general area of training of medical practitioners where motion of blood vessels, tendons etc., motion inside the gastro-intestinal tract or intestine and actions such as tying of knots and suturing needs to be simulated with a high degree of realism [11, 12]. Majority of the existing approaches may be categorized into two groups, algorithms based purely on physically deformable models and algorithms based on geometry. We first discuss

algorithms based on physically deformable models.

The review paper by Nealen et al. [13] discusses the existing physically deformable models in computer graphics in detail and gives a very good overview. In one of the earliest works, Barzel [14] uses mode shapes of a constrained string model and key-framing techniques to fake the dynamics of a string. Hergenröether and Däehne [15] discretize the flexible 1D object into a large number of small (linear) rigid objects, each endowed with mass and connected by different kinds of springs and dampers. With appropriate choice of parameter values of mass, spring and damping constant, a physics-based realistic simulation was obtained. Taskıran et al.[16] have also employed spring-mass system for simulating hair dynamics. More recent interest in surgery training, where flexible suturing thread is modeled as spring-mass-damper systems [11, 12], is related and similar to this problem. In these works, the equations of motion are solved to predict the motion of parts of the rope. In an extension to these works, Güdükbay et al.[17] proposes spring mass systems for simulation of cloth motion. However, a major difficulty in this approach is that there is no systematic way of choosing the spring, damping and other parameters. The main issue in these techniques is to choose or obtain appropriate spring and damping constants for which Natsupakpong et al.[18] have devised an algorithm for estimating these based on error minimization between FEM and lumped element models. However, the algorithm is not feasible for real-time implementation. In another approach, Grégoire et al. [19], Spillman et al. [20] and Dinesh et al. [21] uses Cosserat model for rod-like solids to model bending and torsion for real-time realistic simulation of flexible parts. Moll and Kavaraki [22] present path planning for flexible 1D objects using minimal energy curves and probabilistic root maps. Lenoir et al.[23] uses Lagrangian formulation posing lumped masses on the control polygon vertices and springs along the curve (resisting bending and stretching) combined with spline refinement techniques of sub-division/merging to simulate the motion of flexible objects. In an extension to this work, Theetten et al. [24] generates geometrically exact expressions for deformations of flexible 1D objects by concurrently using beams and spline theory. Goldenthal et al. [25] use a constrained Lagrangian mechanics based approach to handle in-extensible cloth simulation – the length constraint (in-extensibility) is explicitly enforced on the cloth mesh thereby increasing computational effort especially when the resolution in the cloth model is increased. To overcome the stiff nature of the differential equation in cloth simulation (due to the high compliance along normal motion compared to almost zero compliance in the in-plane extension), Baraff and Witkin [26] use an implicit solver to simulate cloth motion. This is however an iterative procedure and convergence is an issue. Wang et.al. [27] use strain limiting algorithms to overcome the stiffness issue. In another work, Mikchevitch et. al. [28] use free-form surfaces and flexible beams to model a real time simulator for assembly-dis-assembly

operations. In all the above mentioned works, dynamics is incorporated and a large amount of effort is towards speeding up the computation or improving the accuracy by adjusting the algorithms. However, in general, all of these methods suffer from one or more issues like stability, convergence, stiff systems, dependence on many arbitrary parameters, phantom forces from high residuals, excessive damping/numerical losses or lack of feasibility for real-time implementation.

In contrast to the above mentioned approaches, several authors have focused on viewing simulation of flexible one-dimensional objects as a kinematic problem so that the issues of stability, convergence and choice of parameters do not arise. Brown et al. [29] have presented tying of knots in a rope with a geometric approach where the flexible 1D object is discretized into linear segments connected by joints and the motion of a trailing segment uses a *follow the leader* based strategy. Su et al. [30] use inverse kinematics and energy minimization to approximately preserve length of deformed poly-line and a 4-point subdivision scheme is used to obtain smooth $\mathcal{C}^1$ curve from the deformed poly-line. In another work, Sreenivasan et al. [31] use the closed-form equations of the classical *tractrix* curve[1] to iteratively compute the motion of all trailing linear segments. In a subsequent work, Menon et al. [33] have shown that the tractrix based solution can be derived from a constrained optimization problem involving minimizing the velocity of points on a curve subject to preservation of the length of the curve. In the above works, only the kinematics of the 1D object is used to impart realism in the simulation and rendering and since the flexible 1D object is discretized into linear rigid segments, the length is explicitly and always preserved. Ideally, the algorithm should be purely kinematic, numerically stable and easily scalable to large degree of freedom (DOF) systems and the preservation of the length. In this work we have used the powerful tools from calculus of variations to solve an optimization problem which results in a purely kinematics algorithm. The topic of calculus of variations is described in brief next.

Calculus of variations is a field of mathematical analysis that deals with maximizing or minimizing functionals, which are mappings from a set of functions to the real numbers. Functionals are often expressed as definite integrals involving functions and their derivatives, as

$$J[y] = \int_{x_1}^{x_2} F\left(x, y, \frac{dy}{dx}, \dots\right), \; y(x_1) = y_1, \; y(x_2) = y_2 \tag{1.14}$$

where $J[y]$ is the functional and $y(x_1) = y_1$ and $y(x_2) = y_2$ are the end/boundary conditions.

[1]According to Steinhaus [32], the tractrix is *the path traced by an object starting off with a vertical offset when it is dragged along by a string of constant length being pulled along a straight horizontal line.* The tractrix curve was studied by the great mathematician Leibniz, who obtained the differential equation and analytical solution of the curve.

The Euler-Lagrange equation for the local extremization of the functional as in Equation (1.14) is given by

$$\frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) = 0, \quad \text{where } y' = \frac{dy}{dx} \tag{1.15}$$

Often there are problems where the extremal must satisfy a constraint (also known as subsidiary conditions or side conditions) in two forms-integral equation or algebraic equation. These are addressed using the method of Lagrangian multipliers[34] to generate an augmented functional which is subsequently solved by the Euler-Lagrange equation. We discuss briefly, the two types of constraints.

- **Global Constraint:** In this case the subsidiary condition is in the form of an integral equality and the functional and the constraint is given as

$$\begin{aligned} J[y] &= \int_{x_1}^{x_2} F(x, y, y', \dots), \ y(x_1) = y_1, \ y(x_2) = y_2 \\ \text{such that } K[y] &= \int_{x_1}^{x_2} G(x, y, y', y'') = l \end{aligned} \tag{1.16}$$

  Solution to this problem, if the extremas of $K[y]$ and $J[y]$ do not coincide, is the extremal of the modified functional

$$\int_{x_1}^{x_2} (F + \Lambda G) dx \tag{1.17}$$

  where $\Lambda \in \mathbb{R}$ is a Lagrange multiplier.

- **Local Constraint:** Here, the subsidiary condition is in the form of an algebraic equality to be obeyed over the problem domain ie.

$$\begin{aligned} J[y, z] &= \int_{x_1}^{x_2} F(x, y, z, y', z', \dots), \\ & y(x_1) = y_1, z(x_1) = z_1, \ y(x_2) = y_2, \ z(x_2) = z_2 \\ \text{such that } & g(x, y, z) = 0 \end{aligned} \tag{1.18}$$

  Provided the partial derivatives $g_x$ and $g_y$ do not vanish simultaneously at any point on the surface defined by the constraint, the solution to this problem is given by the extremal

of the modified functional

$$\int_{x_1}^{x_2} (F + \lambda(x)g)dx \tag{1.19}$$

In this work, we wish to minimize a norm (or a *distance measure*) between two configurations of a deformable curve subject to length-preservation constraint[2]. There have been extensive studies in the field of kinematics to develop a norm for rigid body pose so that questions such as those related to rigid body guidance can be answered using variational calculus. However, this has been largely unsuccessful till date[36]. In this thesis, we use the temporal area traced by a curve (of the form of an $\boldsymbol{L}^2$ norm) to measure the closeness between two curves. Though not a norm/distance function in the classical sense (it does not satisfy conditions in pp.125,[37]), for incremental motion of curves, this norm is a measure of velocity/incremental motion of the flexible one dimensional object. The concept is illustrated in the Figure 1.3 and the functional takes up the following form mathematically.

$$J[y] = \int_0^L \sqrt{\left(\frac{\partial}{\partial t}x(s,t)\right)^2 + \left(\frac{\partial}{\partial t}y(s,t)\right)^2}\, ds \tag{1.20}$$

where $s, t$ denote arc length and time, respectively.

We minimize the root mean square velocity norm of the incremental motion of a constant length flexible one dimensional object between two time instants $t$ and $t + \Delta t$ for a known perturbation of a point on the flexible object. From a robotics perspective, this minimization leads to resolution of redundancy and we get a solution involving least overall motion of the object, which in turn can be related to minimization of kinetic energy and expenditure incurred from the actuators used to move the manipulator.

## Motion planning of hyper-redundant serial robots

In a serial robot, if the number of actuated joints is more than six for motion in 3D space and more than three for motion in a plane, then there exist infinite joint angle sets which will achieve the same position and orientation of the end-effector of the robot. As an example, consider a planar robot with three revolute (R) joints with rigid links of lengths $l_i,\ i = 1, 2, 3$ and joint

[2]In general, the space of admitted functions are Sobolev spaces [35].

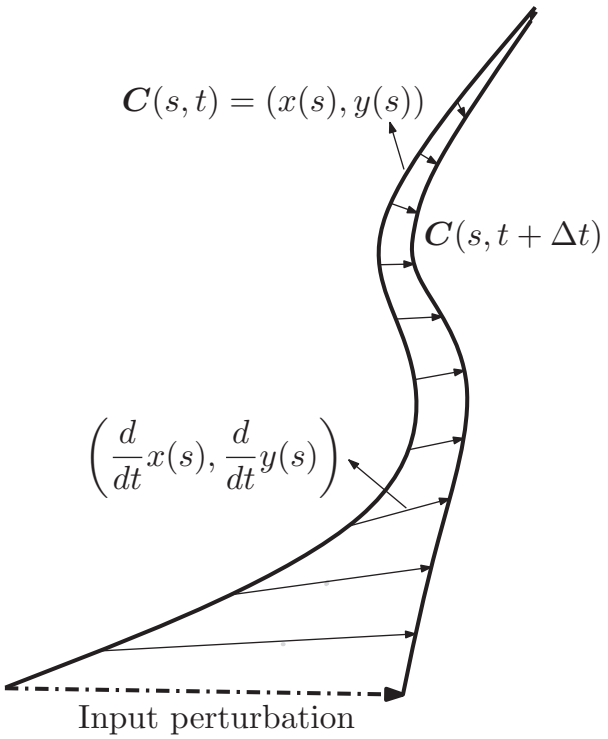



Figure 1.3: $\boldsymbol{L}^2$ norm used for velocity/incremental motion measure of a curve

angles $\theta_i,\ i = 1, 2, 3$. The position of the end-effector, denoted by $(x, y)$ can be written as [38]

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \end{bmatrix} \tag{1.21}$$

In the inverse kinematic s problem, $(x, y)$ is prescribed and the goal is to obtain the three joint angles, $\theta_i,\ i = 1, 2, 3$. The above is a system of 2 equations with 3 unknowns and one can obtain infinitely many $\theta_i,\ i = 1, 2, 3$ for a given $(x, y)$. For motion in 3D space, The position and orientation of the end-effector (namely 6 quantities) are prescribed and the goal is to obtain an *unique* set of joint variables. This is called the *resolution* of redundancy. If this number of actuated joints is much more than six (for 3D) and three (for planar motion), such a robot is called hyper redundant robot.

In the robotics community, resolution of redundancy of hyper-redundant snake-like and other robots with large number of rigid links connected by actuated joints have been a continuing research area (see, for example, [39, 40, 41, 42] and the references therein). One of the earliest techniques involved the use of the manipulator Jacobian matrix to minimize joint rotation, velocity, torque or to avoid obstacles and singularities in the path of the robot [39]. This

approach involves obtaining the pseudo-inverse of the manipulator Jacobian matrix and can have a complexity of $\mathcal{O}(n^3)$ where $n$ is the number of joint variables. Pseudo-inverse based methods are thus not suitable for motion planning when the numbers of links and joints are large. A second approach developed by Chirikjian et al. [40] and Zanganeh et al.[43] involves the use of a backbone curve to approximate the redundant robot and the motion planning is done on the backbone curve. The complexity of their algorithm is $\mathcal{O}(n)$ but in this approach the length of the curve may not be preserved. Various authors have also attempted to utilize this redundancy to optimize suitable objective functions of the robot motion variables, avoid singularities in workspace and to avoid obstacles (see, the review paper by Klein and Huang [44] and works by Liegeois [45], Baillieul et al. [46] and Yoshikawa [47]). In yet another approach by Reznik and Lumelsky [41, 48, 49] and more recently by Ravi et al. [42], the motion planning is done in the task space (instead of in the joint space) using the classical tractrix curve using an $\mathcal{O}(n)$ algorithm. In the tractrix curve-based approach, for a prescribed motion of the *head*, the motion of the *trailing* end of the same segment is computed using the analytical expression of the tractrix curve. For the next segment, the motion of the leading end is assumed to be the computed motion and the motion of its trailing end is again computed using the expressions of the tractrix curve and motion of all segments are computed iteratively in this manner. One key property of a tractrix is that for a given motion of the leader, the motions of the trailing parts die down or attenuates as one moves away from the perturbed end. An additional well-known property of the tractrix is that the velocity of the trailing end is along the line joining the trailing and the leading end and, more importantly, this velocity is the *least* among all possible velocities of the trailing end of the object for a given motion of the leading end. This suggests that the tractrix-based solution can be obtained from a general minimization problem and results in a more natural motion of the robot. In this thesis, we develop such a minimization formulation and solve the problem using calculus of variations. It is shown that we can obtain general results which for the special case of a serial hyper-redundant robot reduce to the tractrix based redundancy resolution.

In the context of motion planning of flexible one-dimensional objects, they can be discretized into a large number of rigid segments connected by joints and tools from redundancy resolution of hyper-redundant robots can be used for simulation. The goal would be to prescribe $(x, y, z)$ (for 3D motion) of a point on the flexible 1D object and obtain the motion of all all other points on the flexible 1D object. In this work, we use the analytical results obtained from minimizing the distance measure discussed in the previous section to obtain realistic and efficient motion planning and rendering of the motion of flexible one-dimensional objects.

# Obstacle avoidance for flexible one-dimensional objects and hyper-redundant robots

In the motion planning of flexible one-dimensional objects and hyper-redundant robot, one of the key aspect is avoiding obstacles. The problem of obstacle avoidance for point bodies or mobile objects and robots approximated by a point has been studied in depth and many solutions have been proposed (see the review paper by Hwang et al. [50] and the references contained therein). The various methods can be broadly classified into three main categories depending on their approaches. The first approach involve mapping obstacle information and geometry into the workspace of a robot and partitioning the space into free and occupied spaces and then finding a obstacle free path for the robot in this space (see, for example, [51, 52]). A second approach is based on a method termed as the dynamic window approach (see, for example, [53]) where graph theoretic constructions and search is used solve the so-called find path problem. A third extensively used approach uses artificial potentials (see Khatib [54] and later extensions of the concept by others [55, 56]) where a repulsive virtual force field is generated around the obstacle locations and then a minimal potential energy path is computed. The computed path effectively avoids the obstacles and can be directly implemented in the control algorithm and hardware used by the robot. In addition, to the three mentioned approaches, due to the need for effective and real-time obstacle avoidance for mobile and stationary robots, researchers have attempted to use Voronoi diagrams [57], artificial neural networks [58], polyhedral interference detection based on computational geometry [59], reinforcement learning algorithms [60], dynamic programming [61] and optimal control (see, for example, references [62],[63],[64],[65]).

Unlike motion planning for point objects, there is less work on obstacle avoidance algorithms for entire manipulators, redundant or otherwise. In redundant manipulators, the main approach is to carefully choose one of the infinitely many solutions such that interference with the obstacles is avoided. Freund [66] used redundancy to trace a spatial trajectory avoiding obstacles whereas Nakamura [67] proposed an algorithm to avoid obstacles by placing restriction on joint angles indirectly while using the pseudo-inverse of the manipulator Jacobian to resolve the redundancy. In the configuration space based approach (see, for example, [68, 69, 70]) the spatial description and geometry of the obstacles and the robot manipulator are used to partition the configuration space of the manipulator into interference free zones. These are in turn used for motion planning and obstacle avoidance. Obstacle avoidance for a redundant robot have also been attempted with instantaneous Jacobian [71], artificial neural networks [72] and optimal control [73]. However these methods do not explicitly address the extended and

articulated nature of a hyper-redundant robot's physical structure.

For hyper-redundant manipulators or one dimensional flexible objects modeled as hyper-redundant robots with large number of links and degrees of freedom, almost all of the above mentioned approaches are not useful. This is due to demand for large computational power for real-time simulation and visualization of the motion. To overcome this problem, Reznik and Lumelsky [49] proposed the use of a classical tractrix curve combined with an iterative obstacle avoidance algorithm based on active sensing of the environment. They claim efficient real-time simulation for hyper-redundant robots and obstacles in two- and three-dimensions. Subsequently, Choset [74] proposed a follow-the-leader approach for obstacle avoidance combined with generalized Voronoi graph. These methods essentially divide the motion planning problem into two distinct phases - one in free space and one in the vicinity of obstacles. Chirikjian and Burdick [75] proposed an discrete summation model using differential geometry to constrain the manipulator into obstacle free zones called tunnels. An interesting approach, called obstacle aided locomotion, has been used in reference [76] wherein the obstacles are used to generate reaction forces for locomotion, thus mimicking natural snakes, and in a sense obviating the problem of obstacle avoidance. In reference [33], authors have proposed a tractrix based motion planning algorithm based on optimization.

The primary drawback of some of these methods is that a significant amount of engineering and algorithmic intuition is required to formulate the algorithm for motion planning. For example, in the Jacobian-based methods, it has to be ensured that algorithmic singularities are either not encountered or at least addressed explicitly, which limits the applicability to well-known environments. In the modal approach, the choice of modal functions is a non-trivial task, where several sets of modes may need to be defined to span the workspace. Some of these methods are also computationally very expensive. For example, configuration space methods are generally not computationally feasible for hyper-redundant robots due to the high dimensions of the configuration space. In general, except for the follow-the-leader approach, none of the methods provide intuitive solutions from the perspective of a human machine interface, which can be a serious drawback during, for example, tele-operation by a human operator. The follow-the-leader approach, while being effective in the immediate vicinity of an obstacle, is inefficient in relatively open spaces.

In this work, obstacle avoidance is incorporated as constraints in the velocity minimization scheme discussed in Section 1.2 and a constrained Lagrangian formulation is used. The algorithm is computationally efficient as it breaks down the obstacle avoidance problem for the $n$-degree-of-freedom (DOF) hyper-redundant manipulator to $n$ obstacle avoidance problems for the 1-DOF rigid links. It is also shown that in free space, the motion is along the link (as in

the tractrix based approach) and in the vicinity of the obstacle the link moves along the local normal to the obstacle surface.

## Contributions of the thesis

The main contribution of the thesis are as follows:

- A theoretical framework for formulating the motion planning of one dimensional flexible objects/hyper-redundant robots as an optimization problem is developed. This allows the use of techniques from calculus of variations resulting in analytical expressions of solutions in certain scenarios. This provides invaluable insights on the solution characteristics like the tangential character of local velocities at any point on the curve in the ensuing motion.
- The relationship of tractrix and incremental motion/velocity minimization of a generic curve in 2D/3D has been obtained analytically. This allows the break the $n$-DOF problem into $n$ 1-DOF problems, thereby drastically reducing computational requirements for the motion planning. More quantitative results have been provided in Chapter 2.
- Analytical expressions on the relationship between control polygon lengths and curve lengths for a generic quadratic and cubic splines have been derived.
- An algorithm for generating approximately length-preserving motion of spline curves have been formulated. The use of the algorithms results in more efficient and realistic rendering of the motion of one-dimensional flexible objects.
- Obstacle representation and avoidance has been successfully incorporated into the minimization based approach and optimal obstacle avoidance maneuvers have been generated.
- The algorithm for motion planning and obstacle avoidance has been validated on a fabricated 12-DOF snake robot and shown to be practically feasible.

## Preview

The thesis is organized as follows: in Chapter 2, we present the calculus of variations based approach for minimizing the motion of a flexible one-dimensional object and the derivation of the tractrix curve for a straight segment. In Chapter 3, the analytical expressions relating the length of a spline curve as a function of the angle between the adjacent segments of the underlying control polygon and the adaptive algorithm for approximate length-preservation of a moving spline curve and numerical simulation results are presented. Chapter 4 deals with the

formulation of obstacles into the minimization framework, describes the construction of a 12-link hyper-redundant robot and experimental results which validate the algorithms developed in Chapters 2 and 3. Finally in Chapter 5 we present the conclusions and present the scope for future work in this area.

# Chapter 2

# Minimization Based Motion Planning

## Introduction

This chapter deals with the notion of length preservation and velocity minimization to arbitrary planar and spatial curves modeling one-dimensional flexible objects. Using formal tools from calculus of variations, we minimize an $\boldsymbol{L}^2$ norm between two configurations of a curve subject to the length preservation constraint. We show that, for any planar or spatial curve subject to preservation of the length, the $\boldsymbol{L}^2$ norm gets minimized when the velocity of any point on the curve is along the tangent at that point. For the special case of a straight line, we show that the results are identical to the classical tractrix solution. Subsequently, by means of a limiting argument, we prove that for a arbitrary curve, the motion dies out as one moves away from the perturbed end and the effect of the motion or disturbance is localized. We also show that when the motion is along a straight line the entire curve, independent of its initial shape, deforms and eventually aligns with the motion direction and becomes a straight line. Both these features results in a more "natural" looking motion of the curve and the flexible one-dimensional object modeled by the curve. Finally, we also demonstrate that other metrics such as the angular motion (i.e., the bending of the flexible object) and stiffness can also be minimized and each of these minimizations lead to different unique solutions.

The chapter is organized into six sections. For the sake of completeness, we present the motion of a single straight rigid body, the associated concept of a tractrix and present its key properties in Section 2.2. In Section 2.3, we use calculus of variations to derive analytical results dealing with the motion of an arbitrary curve. It is shown that the general results yield the tractrix solution when the curve is a straight line. Subsequently, an algorithm to simulate the motion of a flexible object modeled as a curve or as a piece-wise linear segments is presented. In Section 2.4, other metric functions which could also be used for minimization are presented.

The simulation results are described and discussed in Section 2.5. The chapter is summarized in in Section 2.6.

# The tractrix formulation

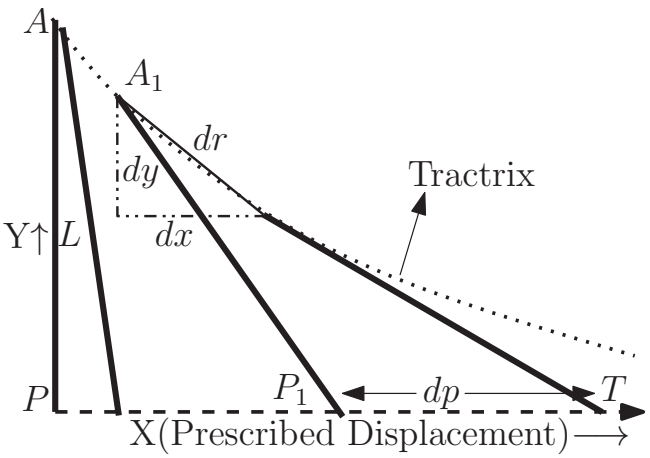



Figure 2.1: Motion of link AP when leading end P moves along PT($X$-Axis)

Consider a rigid link $AP$ of length $L$ initially lying along the $Y$ axis as shown in Figure 2.1. Let the end $P$, the leading end, be moved along the $X$ axis. Let $A_1P_1$ denote an arbitrary configuration of the rigid rod. Without any constraint on the motion of end $A$, i.e., the trailing end, it can trace an arbitrary curve in the $X-Y$ plane. For example, $A$ can move parallel to the $X$ axis and this corresponds to a pure translation of the rod. Now consider a constraint that the velocity *is along the rod at every instant*. Under such a constraint, the motion of point $A$ can be described by the ordinary differential equation

$$\frac{dy}{dx} = \frac{-y}{\sqrt{L^2 - y^2}} \tag{2.1}$$

where $(x, y)$ are the coordinates of point $A$ and the denominator $\sqrt{L^2 - y^2}$ arises from using the length preservation constraint $x^2 + y^2 = L^2$.

Equation (2.1) has a closed-form solution [32] given by

$$x = L \log \frac{y}{L - \sqrt{L^2 - y^2}} - \sqrt{L^2 - y^2} \tag{2.2}$$

which in terms of a parameter $p$ can be written as

$$x(p) = p - L \tanh\left(\frac{p}{L}\right), \quad y(p) = L \operatorname{sech}\left(\frac{p}{L}\right) \tag{2.3}$$

The curve described by Equations (2.2) or (2.3), traced by the end $A$, is called the tractrix and this curve is known to have several interesting properties. We list some of the main ones.

- The magnitude of the instantaneous velocity of $A$ is the minimum of all possible velocities when $A$ traces the tractrix([31]). This follows from the following reasoning:

  From elementary mechanics, the velocity of $A$ can be written as

  $$\mathbf{V}_A = \mathbf{V}_P + \boldsymbol{\omega} \times \boldsymbol{PA} \tag{2.4}$$

  where $\boldsymbol{\omega}$ is the angular velocity and $\boldsymbol{PA}$ denotes the vector from $P$ to $A$ with magnitude $L$. From Equation (2.4) it follows by Cauchy-Schwarz inequality that $|\mathbf{V}_A| \leq |\mathbf{V}_P| + |\boldsymbol{\omega} \times \boldsymbol{PA}|$ and $0 < |\mathbf{V}_A| \leq |\mathbf{V}_P|$. The equality holds if $\boldsymbol{\omega}$ is zero, i.e., when the rigid body is lying on the $X$ axis and $\mathbf{V}_A$ is parallel to $\mathbf{V}_P$.

  Denoting the infinitesimal motion of $A$ by $dr$ and the motion of $P$ by $dp$ (shown in Figure 2.1), we can write $dr \leq dp$ (refer [31] for an alternate algebraic proof).

- The position of $A$ or the trailing end can be obtained in closed-form in terms of hyperbolic functions as shown in Equation (2.3).

- The motion of $P$ need not be along the $X$ axis or in the plane. The formulation can be extended to motion along any direction and in 3D (for details see [31, 42]).

The above properties were first used by Reznik and Lumelsky [41, 48, 49] for resolution of redundancy in hyper-redundant serial robots. In their work, for a desired motion of the end-effector or leading end, the motion of the trailing end was computed. The computed trailing end motion was used as the desired motion of the previous link and thus recursively, the motion of all links, down to the first link, was computed. In comparison to the pseudo-inverse[44] and modal approaches[40] mentioned earlier, the strategy led to a more natural motion since due to the fact that $dr \leq dp$, the motion of the links of the hyper-redundant robot tends to *die out* as one moves from the end-effector to the fixed base. In addition, the redundancy resolution is in *task space variables* and is of *linear complexity*. More recently Sreenivasan et al. [31] used the tractrix based approach to simulate and visualize more natural motion(locally dying out perturbations and minimal overall motion) of snakes and ropes, and Ravi et al. [42] experimentally demonstrated the more natural motion on a planar eight-link hyper-redundant robot.

In the next section, we present an extension of the minimization of motion of a (rigid) straight line segment to arbitrary continuous curves (flexible one-dimensional object) in a plane

and 3D by using tools from calculus of variations.

## Optimizing motion of a continuous curve

Consider a planar curve of length $L$, parametrized by its arc length $s$, and one of its tips is given a prescribed motion. As shown in Figure 2.2, the arbitrary motion of one end is assumed to be given by two independent continuous functions $(T_x(t), T_y(t))$. The curve at any instant $t$ can be written in terms of a spatio-temporal parametrization as

$$C : (T_x(t) + x(s,t), T_y(t) + y(s,t)) \tag{2.5}$$

The terms $(x(s,t), y(s,t))$ define the curve configuration relative to the perturbed tip, i.e., the curve configuration when viewed from a moving coordinate system attached to the perturbed tip. Note that, in this proposed parametrization, the functions $x(0,t) = 0$ and $y(0,t) = 0$ at any instant $t$. This is due to the fact that at $s = 0$ (leading end), the absolute displacements are completely specified by the predefined functions $(T_x(t), T_y(t))$. For the infinitesimal dis-

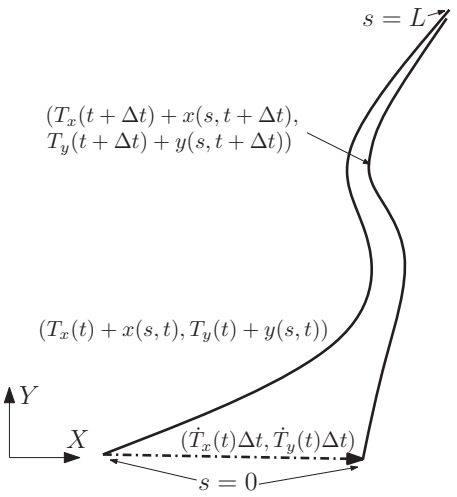



Figure 2.2: Infinitesimally displaced configurations along with perturbation function in continuous case

placement shown in Figure 2.2 from time $t$ to $t + \Delta t$, the magnitude of the velocity vector of an arbitrary point $(x(s,t), y(s,t))$ on the curve is given by

$$|V| = \sqrt{\left(\frac{dT_x}{dt} + \frac{\partial x(s,t)}{\partial t}\right)^2 + \left(\frac{dT_y}{dt} + \frac{\partial y(s,t)}{\partial t}\right)^2} \tag{2.6}$$

One natural choice of a metric would be the velocity magnitude. However, this is for motion of a single point on the curve. To arrive at a similar metric for the motion of the curve[1], we can integrate the velocity magnitude over the length of the curve and this is given by

$$L_2 \ : \int_0^L \sqrt{\left(\frac{dT_x}{dt} + \frac{\partial x(s,t)}{\partial t}\right)^2 + \left(\frac{dT_y}{dt} + \frac{\partial y(s,t)}{\partial t}\right)^2} ds \tag{2.7}$$

We also wish to impose the constraint that the length of the curve is preserved during the motion, and this can be written as

$$\int_0^L \left(\sqrt{\left(\frac{\partial x(s,t)}{\partial s}\right)^2 + \left(\frac{\partial y(s,t)}{\partial s}\right)^2} - 1\right) ds = 0 \tag{2.8}$$

It may be noted that the above constraint only enforces the constancy of overall length of the curve. This does not guarantee local length preservation and hence, the one dimensional curve may expand or contract locally all the while maintaining the overall curve length same.

The variational problem now reduces to one of minimization of the $\boldsymbol{L}^2$ metric in Equation 2.7 over a specified time interval $[0, T]$ and can be stated as follows:

$$\begin{aligned}
&\underset{x(s,t),y(s,t)}{Min\ I} \ : \int_0^L \int_0^T \sqrt{\left(\frac{dT_x}{dt} + \frac{\partial x(s,t)}{\partial t}\right)^2 + \left(\frac{dT_y}{dt} + \frac{\partial y(s,t)}{\partial t}\right)^2} dt ds \\
&Subject\ to \\
&\qquad \Lambda(t) : A = \int_0^L \left(\sqrt{\left(\frac{\partial x(s,t)}{\partial s}\right)^2 + \left(\frac{\partial y(s,t)}{\partial s}\right)^2} - 1\right) ds = 0 \\
&Data : x(s,0), y(s,0), T_x(t), T_y(t), x(0,t) = 0, y(0,t) = 0
\end{aligned} \tag{2.9}$$

where $\Lambda(t)$ is the Lagrangian multiplier corresponding to the length-preserving constraint. The Lagrangian for the above optimization, is written as

$$\mathcal{L} \ = I + \Lambda(t) A \tag{2.10}$$

[1]The defined *metric* is not a measure of distance between two rigid body configurations – such a metric does not exist as has been shown by Angeles [36]. For a single rigid link, we show in Section 2.3.1 that this metric is related to the minimization of velocity of the distal end for a prescribed velocity at the proximal end.

and the corresponding Euler-Lagrange equations [77, 78] are as follows:

$$\frac{\partial \mathcal{L}}{\partial x} - \frac{\partial}{\partial s}\left(\frac{\partial \mathcal{L}}{\partial x'}\right) - \frac{\partial}{\partial t}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}}\right) = 0 \tag{2.11a}$$

$$\frac{\partial \mathcal{L}}{\partial y} - \frac{\partial}{\partial s}\left(\frac{\partial \mathcal{L}}{\partial y'}\right) - \frac{\partial}{\partial t}\left(\frac{\partial \mathcal{L}}{\partial \dot{y}}\right) = 0 \tag{2.11b}$$

Note that $x' = \dfrac{\partial}{\partial s}x(s,t)$ and $\dot{x} = \dfrac{\partial}{\partial t}x(s,t)$ and similar notations are used for the variable $y(s,t)$. By using Equations (2.11a) and (2.11b), and eliminating the Lagrange multiplier $\Lambda(t)$ by division, we obtain the following:

$$\frac{\frac{\partial}{\partial s}y\left(s,t\right)}{\frac{\partial}{\partial s}x\left(s,t\right)} = \frac{\frac{d}{dt}T_y\left(t\right) + \frac{\partial}{\partial t}y\left(s,t\right)}{\frac{d}{dt}T_x\left(t\right) + \frac{\partial}{\partial t}x\left(s,t\right)} = \frac{\frac{\partial}{\partial t}\left(T_y\left(t\right) + y\left(s,t\right)\right)}{\frac{\partial}{\partial t}\left(T_x\left(t\right) + x\left(s,t\right)\right)} \tag{2.12}$$

Though quantitative results are specific to the equation of the assumed curve, i.e., choice of $(x(s,t), y(s,t))$, qualitatively, we can make the following *general* and *key* observations.

- The Lagrange multiplier has the units of velocity and this can be shown as follows:

  The Euler-Lagrange Equations (2.11a) and (2.11b) can be written as

  $$\Lambda(t)y'\kappa_s = -(\dot{y} + \dot{T}_y(t))\kappa_t \tag{2.13}$$

  $$-\Lambda(t)x'\kappa_s = (\dot{x} + \dot{T}_x(t))\kappa_t \tag{2.14}$$

  where $\kappa_s$ and $\kappa_t$ are given by

  $$\begin{aligned}\kappa_s &= \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{\frac{3}{2}}} \\ \kappa_t &= \frac{(\dot{x} + \dot{T}_x(t))(\ddot{y} + \ddot{T}_y(t)) - (\dot{y} + \dot{T}_y(t))(\ddot{x} + \ddot{T}_x(t))}{((\dot{x} + \dot{T}_x(t))^2 + (\dot{y} + \dot{T}_y(t))^2)^{\frac{3}{2}}}\end{aligned} \tag{2.15}$$

  Squaring and adding Equations (2.13) and (2.14), and simplifying, we get

  $$\Lambda(t)\kappa_s = \kappa_t V \tag{2.16}$$

  where $V = \sqrt{(\dot{T}_x + \dot{x})^2 + (\dot{T}_y + \dot{y})^2}$ is the velocity at any point on the curve. Hence, it is clear that the Lagrange multiplier has units of velocity.

  Also, from the form of Equation 2.15, it may be noted that $\kappa_s$ and $\kappa_t$ are spatial and temporal curvatures respectively and relate to each other through the Equation 2.16.

- The extreme left-hand side of Equation (2.12) is the spatial derivative or the slope at a given $s$ and $t$ and the far right-hand side is the temporal derivative or the velocity vector for a given $s$ and $t$. This implies for the curve, the $\boldsymbol{L}^2$ metric as defined in Equation (1.20) *is minimized if the velocity vector at any* $(s,t)$ *is along the tangent at that point.* In addition, during this minimizing motion, *the total arc length of the curve is preserved.*

## Special case of a straight line

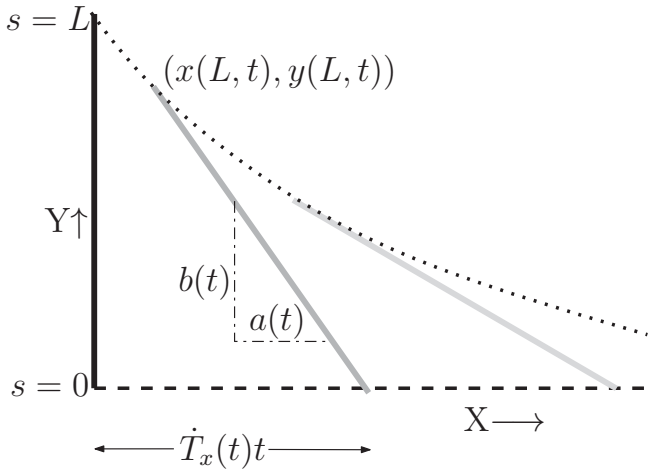



Figure 2.3: Motion of straight rigid curve when leading end moves along $X$-Axis with unit velocity

For the special case when the curve is a straight rigid link and the input perturbation is along the $X$ axis with the perturbed end on the $X$ axis initially, the unit perturbation velocity functions in Equation (2.12) take the form:

$$\frac{d}{dt}T_x(t) = 1, \qquad T_y(t) = 0 \tag{2.17}$$

Since the curve $(x(s,t), y(s,t))$ is a straight line, we have

$$x(s,t) = a(t)s, \qquad y(s,t) = b(t)s \tag{2.18}$$

and since the length of the curve $L$ is preserved, we get

$$\sqrt{(aL-0)^2 + (bL-0)^2} = L \Rightarrow a^2 + b^2 = 1 \tag{2.19}$$

Furthermore, by assuming that the straight rigid link is vertical at $t = 0$, we get $a(0) = 0$ &

$b(0) = 1$. Under the aforementioned conditions, the straight line equations transform as

$$x(s,t) = a(t)s + t, \qquad y(s,t) = b(t)s \tag{2.20}$$

and we also have

$$a^2 + b^2 = 1 \Rightarrow 2a\frac{da}{dt} + 2b\frac{db}{dt} = 0 \Rightarrow \frac{db}{dt} = -\frac{a}{b}\frac{da}{dt},\ b \neq 0 \tag{2.21}$$

Finally, substituting the above in the Equation (2.12), and simplifying, we get

$$\begin{aligned} \frac{b}{a} &= \frac{db/dt\ s}{1 + da/dt\ s} = \frac{-\frac{a}{b}\frac{da}{dt}s}{1 + \frac{da}{dt}s} \Rightarrow s\frac{da(t)}{dt} + (1 - a(t)^2) = 0,\ a \neq 0 \\ \Longrightarrow a\,(t) &= -\tanh\left(\frac{t+K}{s}\right) \text{(where } K \text{ is an integration constant)} \end{aligned} \tag{2.22}$$

Using the initial conditions, we get $K = 0$, and

$$\begin{aligned} x(s,t) =& s\left(\frac{t}{s} - \tanh\left(\frac{t}{s}\right)\right) \\ y(s,t) =& s\ \operatorname{sech}\left(\frac{t}{s}\right) \end{aligned} \tag{2.23}$$

The path (curve) traced by the tip of the link is obtained by substituting $s = L$ in the above equation. Denoting the perturbation in time $t$ with $p$, we get

$$\begin{aligned} x(L,p) =& p - L\ \tanh\left(\frac{p}{L}\right) \\ y(L,p) =& L\ \operatorname{sech}\left(\frac{p}{L}\right) \end{aligned} \tag{2.24}$$

It can be seen that Equation (2.24) is the same as that of a *tractrix* given in Equation (2.3). The above derivation shows analytically that *for a straight single link perturbed along a straight line, the* $\boldsymbol{L}^2$ *metric (as defined in Equation (1.20)) is equivalent to minimizing the velocity of the trailing end.*

As $p \to \infty$ (or $t \to \infty$), we know from elementary calculus that $\tanh(p/L) \to 1$ and $\operatorname{sech}(p/L) \to 0$ which gives

$$x(L,\infty) = p - L, \qquad y(L,\infty) = 0 \tag{2.25}$$

From the above, it is clear that *as time increases, the straight line or the rigid link aligns with*

*the perturbation direction*, in this case the $X$-axis.

## Curve discretized by straight lines

It has not been possible to obtain analytical solution for Equation (2.12) except for a simple straight line as shown in the preceding section. One way to solve the problem for an arbitrary curve is to discretize the curve into finite number of straight line segments. The difference between the continuous (variational) formulation and the discretized formulation is schematically shown in Figure 2.4.

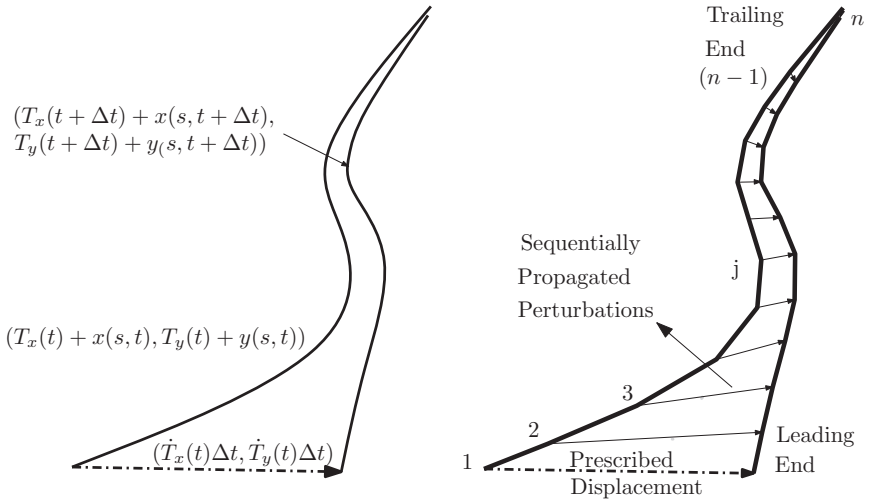



Figure 2.4: Difference between variational and discretized formulations

In Section 2.3.1, we demonstrated that for a straight line the motion of the trailing end with $\boldsymbol{L}^2$ metric or velocity minimization is given by the tractrix Equations (2.24). As shown in the right-hand side of Figure 2.4, a known perturbation is given to the $1^{st}$ point (the leading end) of the initial curve. The $\boldsymbol{L}^2$ metric minimizing the motion of trailing end of the $1^{st}$ segment is computed in closed-form with Equation (2.24) and this is the perturbation for the leading end of the $2^{nd}$ segment. Sequential iteration along the linear segments up to the other end, generates the motion of the entire discretized curve or the $n^{th}$ point. This approach is termed as sequential optimization and is the same as the strategy used for resolution of redundancy in hyper-redundant robots (see [41, 48, 49] and [31, 42]). Since for every iteration $dr \leq dp$, the perturbations die down as one moves down the segments and this gives a more 'natural motion' of the curve. In addition, to this 'dying down' or attenuation property, as shown in Equation (2.25), as time progresses, the motion of each segment and hence the curve aligns with the motion of the perturbed end.

In specifying the motion of the leading end in time domain, we have two options – either a

single step in time from zero to $T$ or a series of steps in time. The series of steps is to be used if the given perturbation is large or along a curve. For realistic simulations, the full step needs to be broken into several smaller steps. The first method is called the single-step optimization and the second is called the multi-step optimization. The multi-step optimization is more accurate when compared to the single step because, in the latter, we neglect the intermediate path points and consider only the initial and final states, which induces error. In the former case, we simply need to use the single step (tractrix approach) several time between $t$ and $(t + \Delta t)$ with $\Delta t$ determined by the requirements of the simulation.

## Spatial motion

For motion in 3D space, we can define the $\boldsymbol{L}^2$ metric as a direct extension of the 2D $\boldsymbol{L}^2$ metric from Equation (1.20). Hence, the minimization problem can be posed as follows:

$$\begin{aligned}
&\underset{x(s),y(s),z(s)}{Min\ I} : \int_0^L \int_0^T \sqrt{\left(\frac{dT_x}{dt} + \frac{\partial x(s,t)}{\partial t}\right)^2 + \left(\frac{dT_y}{dt} + \frac{\partial y(s,t)}{\partial t}\right)^2 + \left(\frac{dT_z}{dt} + \frac{\partial z(s,t)}{\partial t}\right)^2} dtds \\
&\text{Subject to } \Lambda(t) : \ A = \int_0^L \sqrt{\left(\frac{\partial x(s,t)}{\partial s}\right)^2 + \left(\frac{\partial y(s,t)}{\partial s}\right)^2 + \left(\frac{\partial z(s,t)}{\partial s}\right)^2} ds = L \\
&Data : \ x(s,0), y(s,0), z(s,0), T_x(t), T_y(t), T_z(t), x(0,t) = 0, y(0,t) = 0, z(0,t) = 0
\end{aligned} \tag{2.26}$$

One can solve the optimization problem numerically by discretizing the curve into piece-wise straight lines. Alternately, for a discretized flexible 1D object, we can solve the problem as done in reference [31], which is repeated here for convenience of the reader. As shown in figure 2.5, we consider the first straight segment whose one end $(x_h, y_h, z_h)^T$ is being moved to a destination point $(x_d, y_d, z_d)^T$. And the trailing end is $(x_p, y_p, z_p)^T$.

**1)** The three points, $(x_h, y_h, z_h)^T$, $(x_d, y_d, z_d^T$ and $(x_p, y_p, z_p)^T$ lying on the tractrix curve, define a plane.

**2)** Define a coordinate system $\{r\}$ with the $\hat{\mathbf{X}}_r$ along motion of $(x_h, y_h, z_h)^T$ towards $(x_d, y_d, z_d)^T$ and $\hat{\mathbf{Z}}_r$ along the normal to the plane. $\hat{\mathbf{Y}}_r$ completes the right handed system.

**3)** Define the rotation matrix ${}^0_r[\ R\ ]$ with respect to a fixed reference coordinate system $\{0\}$ as ${}^0_r[\ R\ ] = [\hat{\mathbf{X}}_r \quad \hat{\mathbf{Z}}_r \times \hat{\mathbf{X}}_r \quad \hat{\mathbf{Z}}_r]$.

**4)** For the given motion of $(x_h, y_h, z_h)^T$ to $(x_d, y_d, z_d)^T$ in the plane, obtain $(x_r, y_r)$ from the equation of the tractrix given in Equations (2.24). It may be noted that $(x_r, y_r)$ is the

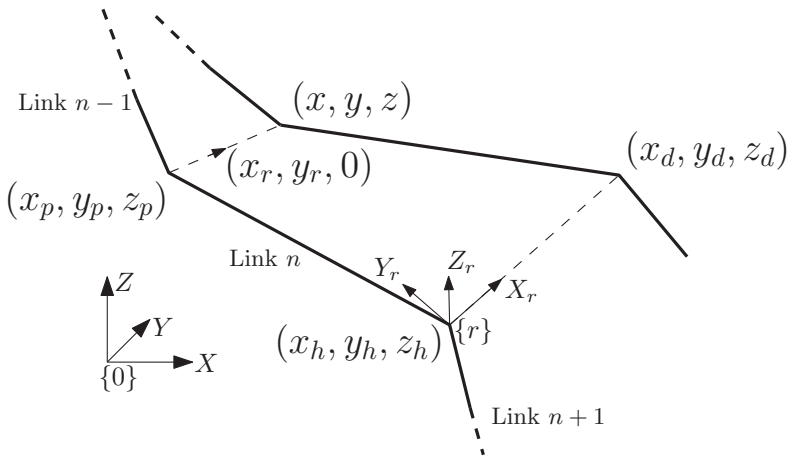



Figure 2.5: Deriving spatial motion from locally in-plane tractrix motions

point $(x, y, z)^T$ in the plane(represented in frame $\{r\}$) and hence the $Z$ coordinate is zero.

**5)** Once $x_r$ and $y_r$ are known, the new position of trailing end of the link(which lies on the tractrix) after motion $(x, y, z)^T$ in the fixed reference coordinate system $\{0\}$ is given by

$$(x, y, z)^T = (x_p, y_p, z_p)^T +^0_r [\ R\ ](x_r, y_r, 0)^T \tag{2.27}$$

The motion of the second segment can be found by setting the destination point of one end as the computed $(x, y, z)^T$ (in step 5) above) and computing the motion of the other end by following steps 1) through 5) above. Proceeding in a similar manner, the displacement of the leading end of the $(i-1)^{\text{th}}$ segment is the displacement of the trailing end of the $i^{\text{th}}$ segment and one can thus iteratively go down to the $n^{\text{th}}$ linear segment and obtain the new configuration of the discretized flexible 1D flexible object. As shown in reference [31], the motion of the flexible 1D object, discretized as $n$ straight segments, can be computed in $\mathcal{O}(n)$ steps.

In Section 2.5, we present several numerical simulation results for the motion of flexible objects in 2D and 3D space. The interesting features of preservation of the length, dying-out of motion and eventual aligning of the flexible 1D object along the perturbation vector is discernible in the simulations.

## Use of other metrics for minimization

In the previous section, we presented a metric, which for the case of single link expresses minimization of velocity of the trailing end for a given input velocity of a leading end. In this section, we present two additional possible metrics. In Section 2.5, we compare these metrics.

## Minimization of body rotation

Consider a straight rigid segment $AP$ and the leading end $P$ is moved to point $Q$. We wish to obtain the motion of the trailing end $A$ such that the straight rigid segment rotates the least. In Figure 2.6, point $A$ is schematically shown to move to $B$ while maintaining the length $|\boldsymbol{AP}| = |\boldsymbol{BQ}|$ and hence the rotation angle is given by $\angle BQR$ where line $QR$ is parallel to $AP$.

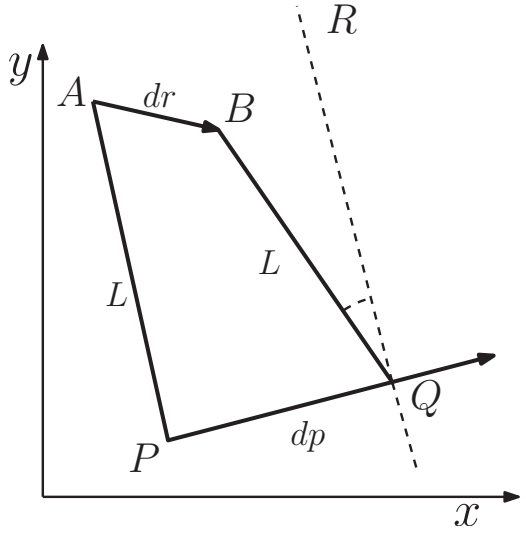



Figure 2.6: Incremental rotation minimization

We now pose a minimization problem:

$$\begin{aligned} \underset{x_B, y_B}{Minimize}: & \ |(\angle BQR)| \\ Subject\ to: & \ |BQ| = L \\ Data: & \ L,\ Step\ Length\ PQ \end{aligned}$$

It may be noted that minimization of the rotation is in the same spirit as the $\boldsymbol{L}^2$ velocity minimization discussed in the previous section. Since a general rigid body motion can be considered to be a combination of translation and rotation, a solution of the above problem in combination with the tractrix approach may result in alternate natural rigid-body motions. It may be also noted that a curve can be discretized into straight rigid segments and the technique of sequential optimization can then be applied for the discretized curve.

### Minimization of body deformation

Consider a rigid body comprising of two rigid line segments $MA$ and $AP$. The end $P$ is again given a perturbation to $Q$. We wish to determine the motion of $A$ and $M$, schematically shown going to $B$ and $N$ respectively in Figure 2.7, such that the $(\angle MAP - \angle NBQ)$ is minimized. This can again be expressed as a minimization problem as follows:

$$\begin{aligned} \underset{x_B, y_B}{Minimize}: & \ |(\angle MAP - \angle NBQ)| \\ Subject\ to: & \ |BQ| = L, |NB| = L \\ Data: & \ L,\ Step\ Length\ PQ \end{aligned}$$

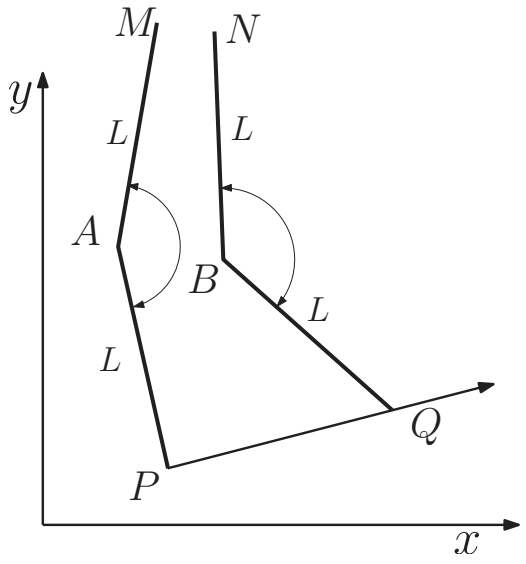



Figure 2.7: Incremental joint rotation minimization

## Results and discussion

In this section, we present numerical simulations to illustrate the theoretical results presented in the previous sections. All symbolic computations required for the results were obtained using MAPLE® [79] and for numerical simulations using the *fmincon* routine of MATLAB® [80].

### Performance comparison of different optimization strategies

The first simulation result deals with a comparison of optimizing the motion of a continuous curve (not discretized into piece-wise linear segments) with the sequential, single and multi-step, optimization discussed earlier. As noted in Section 2.3, we can solve the continuous

optimization for simple curves. Figure 2.8 shows the motion of an initial parabola using different optimization strategies. The chosen initial parabola has the form $y = x^2$ and it is discretized into 20 segments, each of length 0.25 thereby creating a total curve length of 5. Further, for single-step perturbation, we have a motion vector of 0.7 length equally inclined to $X$ and $Y$ axes (45° with each axis) whereas for the multi-step perturbation, we move the curve in the same direction but in 5 steps of 0.14 each.

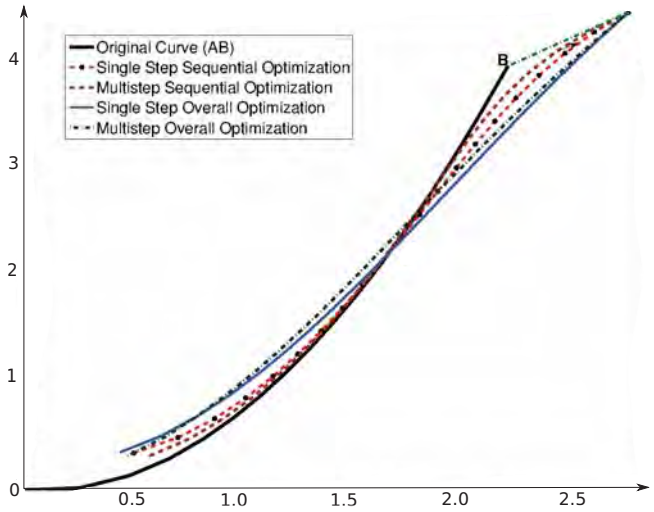



Figure 2.8: Different optimization techniques applied on an initial parabolic curve

To compare the four minimizations, we choose the multi-step sequential optimization as the reference and compute deviations norms from the same. For single-step sequential optimization,

| # | Overall Optimization | | Sequential Optimization | |
|---|---|---|---|---|
| | Single-Step | Multi-Step | Single-Step | Multi-Step |
| $\boldsymbol{L}^2$ Metric error | 1.6982 | 1.6007 | 0.6347 | 0.0000 |
| Time of execution(s) | 33.6124 | 39.5479 | 15.2364 | 22.4861 |

Table 2.1: Comparison of Different Metrics

the $\boldsymbol{L}^2$ norm is 0.6347 and for multi-step overall optimization the value is 1.6007 whereas for single-step overall optimization, it turns out to be 1.6982. This implies that better results are obtained if the discretization in time and along the curve is finer.

As far as CPU times are concerned, the seqquential single step is the fastest as it ignores any of the intermediate configurations. This is followed by sequential multi-step algorithm. The longest times are taken by the overall optimization algorithms because is issues of convergence to optima.

## Solution characteristics of candidate metrics

The second set of numerical results deals with the comparison of the tractrix, the minimization of body rotation and body deformation for a straight line. As explained earlier, selecting appropriate metrics depends greatly on the problem-specific task. In this section, the solution properties for the three metrics used are discussed.

1. The $\boldsymbol{L}^2$ norm minimization yields a tractrix-based solution as already mentioned earlier and shown in Figure 2.9. It may be noted that the initial configuration of the straight line is parallel to the $Y$ axis, the motion is from right to left, and the final configuration of the straight line is parallel to the $X$ axis. The figure clearly demonstrates that the final configuration is getting aligned along the motion direction.

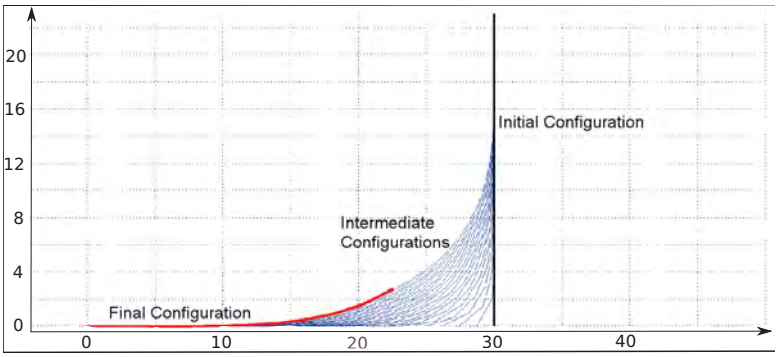



Figure 2.9: $\boldsymbol{L}^2$ norm minimization for a straight line

2. Rotation norm minimization introduces the notion of pure translation mode, which can be seen in the Figure 2.10. In this case also, the motion is from right to left.

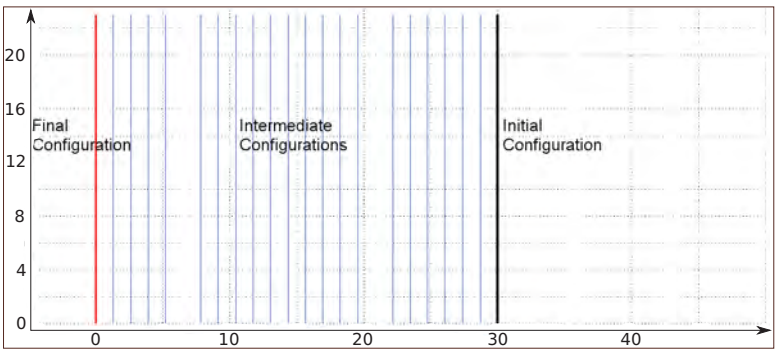



Figure 2.10: Rotation minimization for a straight line

3. The third candidate norm introduced, namely the one to minimize joint rotation, introduces the rigid body behavior into the mathematical framework as can be noted from Figure 2.11. In the Figure 2.11, we can notice that for the first link there is no joint rotation as there is no previous link with respect to which this can be defined. The motion of first link, in this case, has been derived using $\boldsymbol{L}^2$ norm minimization. Clearly, from the figure, we see that the whole body shows a rigid body behavior for minimal body deformation or maximum stiffness.

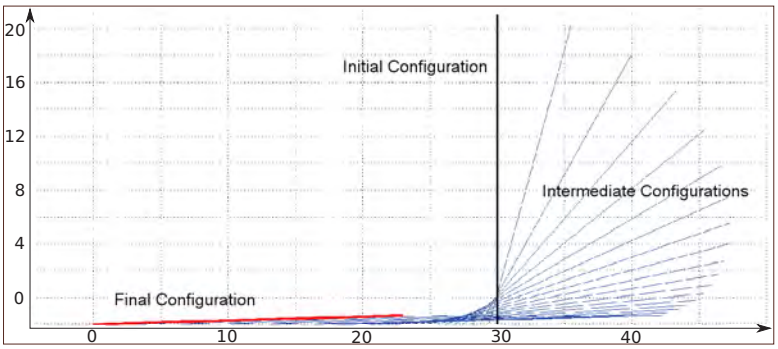



Figure 2.11: Joint rotation minimization for a straight line

The distinct behavior of solutions displayed in Figures 2.9-2.11 indicate the effect of objective chosen on the solution. Depending upon the nature of the problem in hand, proper objective needs to be chosen. For example, if the application is for a flexible robotic manipulator chain, the main objective will be to minimize the joint rotations so as to minimize motor actuation and power consumption, in which case metric defined in Section 2.4.2 for minimizing joint rotations may be more appropriate. However, if the problem is concerned with trajectory planning of a locomotive pulling coaches, then the minimizing the motion or $\boldsymbol{L}^2$ norm of velocities over time is more appropriate.

## Planar and spatial simulation of a generic curve

In this section, we present simulation and visualization of the motion of an arbitrarily chosen curve, whose arbitrarily chosen point is moved along an arbitrary path in 2D and 3D space. Three cases, namely that of an arbitrary planar curve, a parabola and a curve in 3D space are used to illustrate the features of the velocity $\boldsymbol{L}^2$ norm minimization based schemes. In this set of simulations, we use multi-step sequential optimization.

In the planar sequential case, the input curve is discretized into 80 links of 0.05 length each and is perturbed in 250 steps of 0.075 each over a length of 8.75 units along a generic path. For overall optimization in 2D, the curve is again a parabola of the form $y = x^2$ discretized

into 20 links, each of length 0.25, and is given the motion in 6 steps with average motion step length 0.4 units. In the case of 3D, an arbitrary curve of length 30 is discretized into 20 links and perturbed in 2650 steps of 0.1, thus making a total motion step length of 265 units.

Figure 2.12 shows the snapshots of the planar motion from multi-step sequential optimization and Figure 2.13 shows planar motion of a parabola in generic direction using multi-step overall optimization. It may be noted that in Figure 2.12, though the curve is looped, as it is arc length parametrized and out of plane dimension is neglected, no computational complexity/singularity arises. Also, during the motion, the unwinding of the loop acts as perturbation/disturbance absorber for the portion of curve lying downstream with respect to the portion of the curve lying upstream.

Finally, Figure 2.14 show the snapshots of spatial motion and the accompanying videos shows the entire motion for the last two cases. It may be mentioned the length of curve, 4 and 30 units for 2D and 3D curve, remain constant during the motion. It can be clearly seen that the dying-out property is present and this clearly leads to a more natural looking motion of the curve. One can also observe that the eventual motion of the curve aligns with the direction of the motion of the input end. It may be noted that one can see a similar "dying-out" and "aligning" motion in a ribbon being moved by a gymnast during floor exercises.

## Summary

This chapter presented a new approach for the simulation and visualization of the motion of one-dimensional flexible objects using calculus of variations and constrained optimization. It was analytically proved that classical tractrix based solutions are direct consequence of a minimization of the $\boldsymbol{L}^2$ norm of tail-velocity. Subsequently, using sequential optimization to a curve discretized by piece-wise linear segments, a much simpler and computationally more robust algorithm was developed for the simulation of arbitrary flexible one-dimensional objects whose length is preserved during motion. It was shown that the minimization results in a natural 'dying out' and aligning motion of the flexible object. An important feature of the proposed algorithm is that it is a purely *kinematics-based* solution and it does not require assuming values for mass, stiffness or damping of the flexible objects as in dynamics-based approaches. Further, it was shown that the tractrix-based approach is one of the many possible length-preserving transformations for a smooth, arc length parametrizable curve. Through two additional candidate norms, the fact was demonstrated that objective function chosen affects the solution drastically. Therefore, the proposed approach can give the optimized and appropriate solutions for any given functional, which in turn can be decided by the given problem.

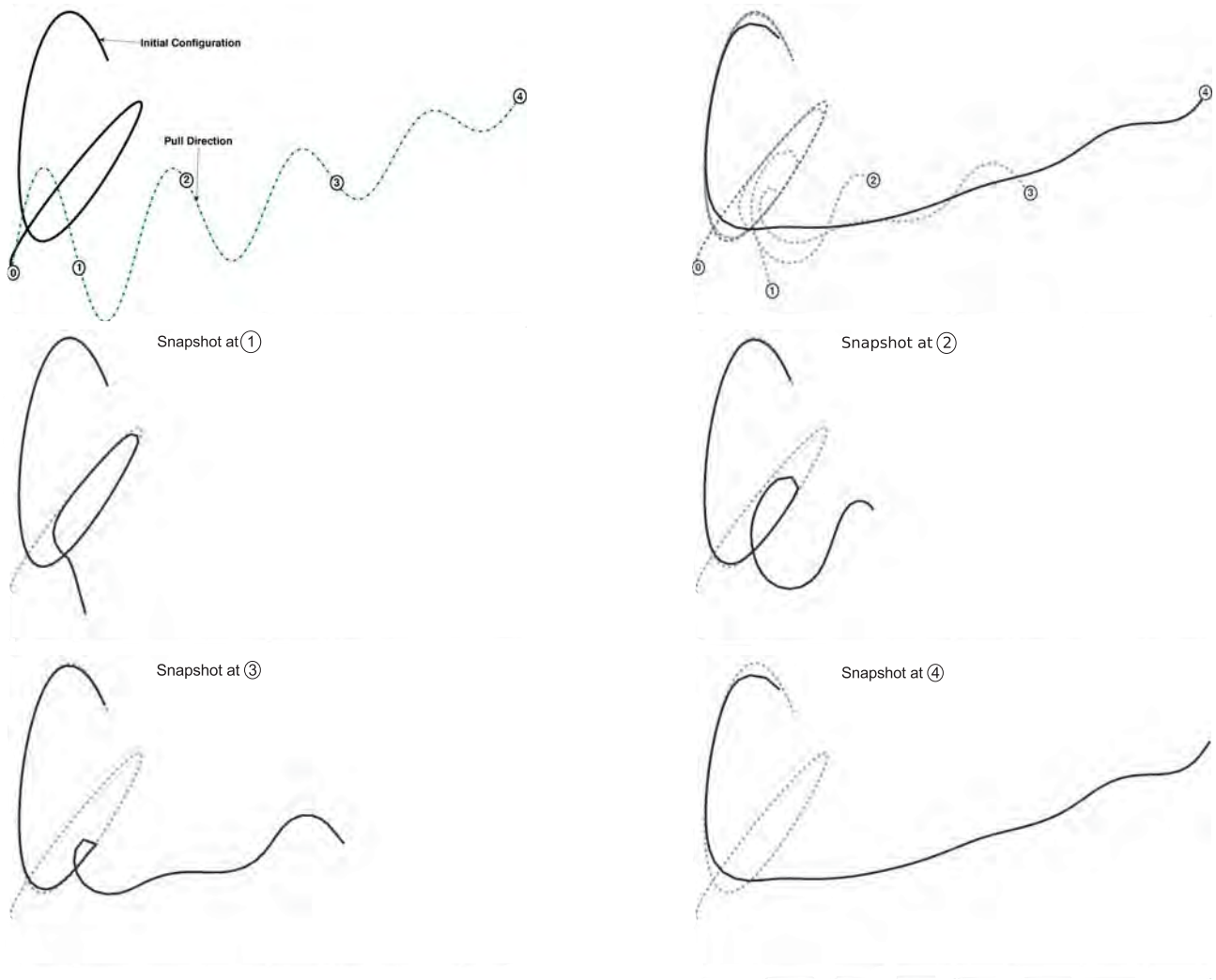



Figure 2.12: Multi-step sequential optimization of an arbitrary curve moved in generic direction

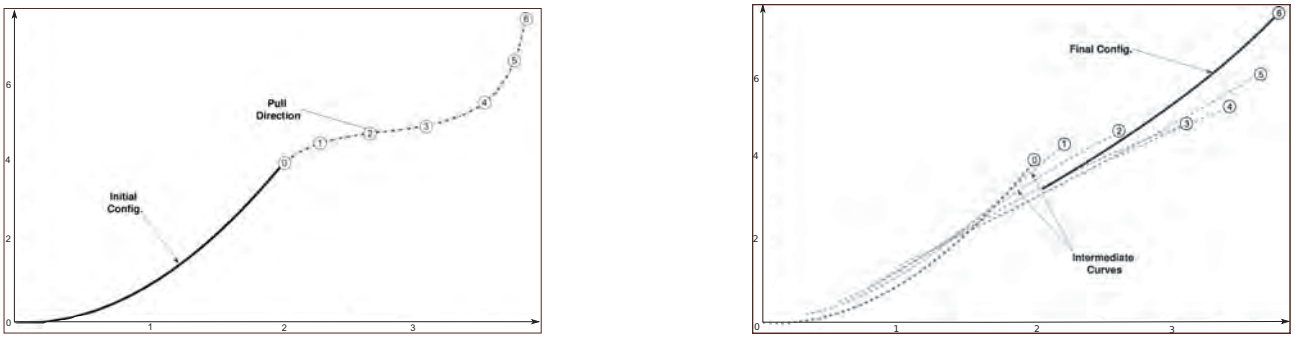



Figure 2.13: Multi-step overall optimization of a parabola moved along a generic direction in 2D

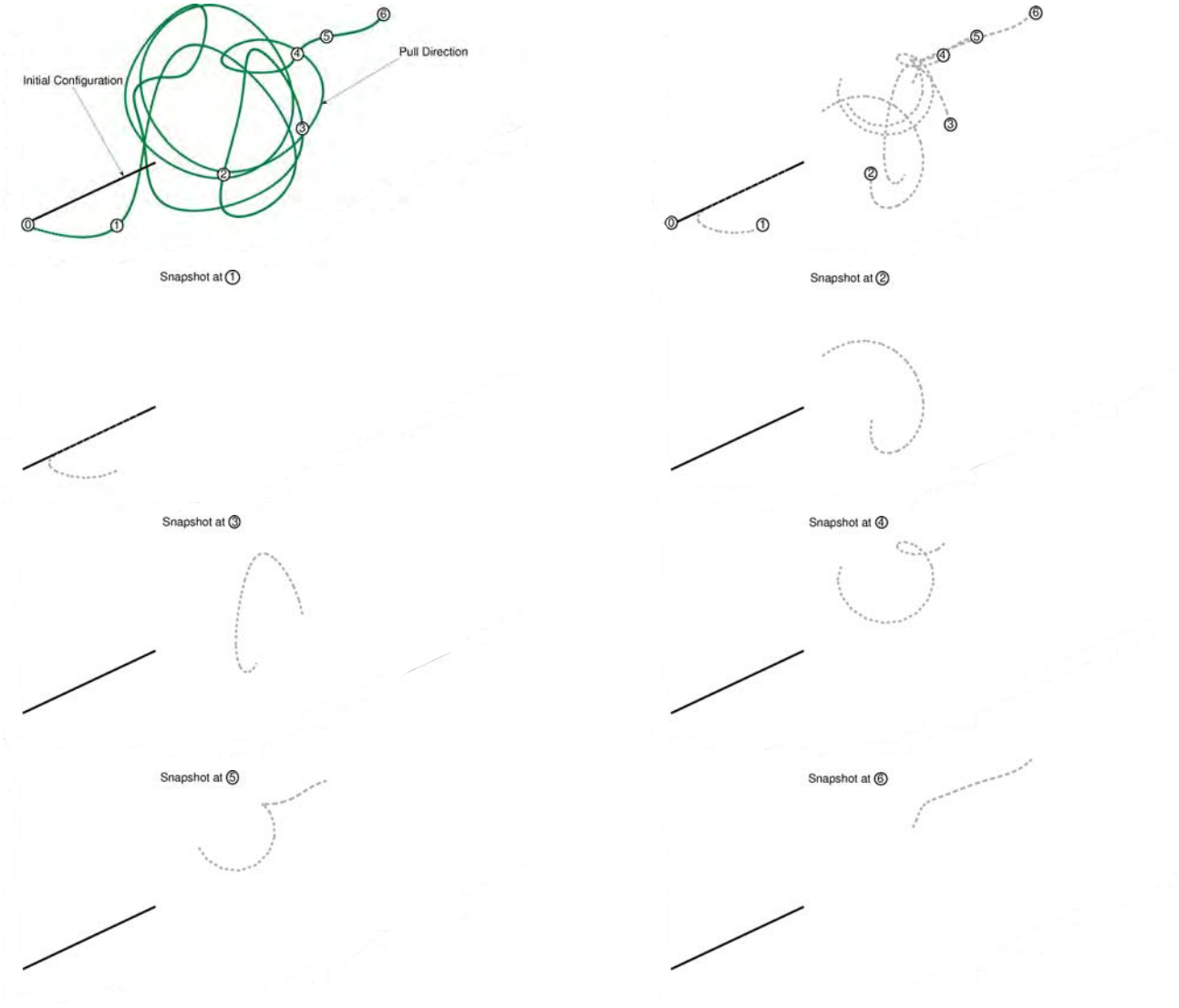



Figure 2.14: Multi-step sequential optimization of an arbitrary curve in generic direction in 3D

# Chapter 3

# Motion Planning of B-Spline Curve

## Introduction

As mentioned in the previous chapter, the flexible one-dimensional object can discretized by straight segments and then the tractrix based approach can be used to obtain a more natural motion. It was also mentioned in the previous chapter that the complexity of the algorithm is $\mathcal{O}(n)$ where $n$ is the number of straight segments. To obtain realistic motion and acceptable rendering, the number of segments can be very large and this can increase the computational burden. In this chapter, the flexible one-dimensional object is modeled as a B-spline curve which can be defined from a control polygon with very few sides. Instead of using the curve or large number of discretized segments for motion planning, the tractrix based algorithm is used on the straight segments of the control polygon. Unfortunately, the motion of the control, and specifically when the angle between two successive segments changes, the length of the curve changes. In this chapter, we obtain new analytic expressions for change in length of a B-spline curve from an initial configuration as the angle between two adjacent segments of its control polygon is changed and present an adaptive algorithm to approximately preserve the length of the curve as the tractrix based algorithm is applied on the control polygon sides for motion planning. The adaptive algorithm uses sub-division and merging of the sides of the control polygon so that a prescribed error tolerance on the length of the curve is maintained at all times during the motion. Since the tractrix based algorithm has a complexity of $\mathcal{O}(n)$ where $n$ is the number of segments used to represent the flexible 1D object and the number of sides of the underlying control polygon is much less than $n$, the complexity of the algorithm can be termed as $\mathcal{O}(1)$. The algorithms for natural and realistic motion planning, the adaptive altering of the control polygon and the mathematical results are illustrated using numerical examples where an arbitrary curve is moved along a generic direction with a prescribed length error tolerance.

The efficiency of the developed algorithms are also demonstrated with the numerical examples.

The chapter is organized as follows: in Section 3.2 we present the new analytic expressions and results for the length of a quadratic B-spline and cubic B-spline curve in terms of the angle between two adjacent segments of the control polygon are presented. The notion of moving the generating control polygon and resulting change in the length of the spline due to motion of the control polygon is also presented. In Section 3.3, we present an algorithm to adaptively subdivide and merge edges of a control polygon to maintain the length of a curve to within a specified length error. In Section 3.4, we present numerical results illustrating our approach for efficient and realistic motion simulation and visualization of motion of flexible one-dimensional objects. In Section 3.5 we present the summary of this chapter.

## Splines and control polygon

As mentioned earlier, the key idea is to move the segments of the control polygon instead of the elements of the discretized curve (poly-line) to reduce computation and enable real-time simulation and visualization.

The dying out and eventual alignment with the input motion features give the tractrix based approach a more natural and physically realistic motion of the motion of a flexible one-dimensional object. The complexity of $\mathcal{O}(n)$ makes it amenable for efficient simulation and realistic visualization of the motion. The $n$ term in the complexity $\mathcal{O}(n)$ can be further reduced if instead of discretizing the flexible object with a large number of rigid linear segments, we approximate the flexible one-dimensional object with a B-spline and we apply the tractrix based motion strategy to the line segments in the control polygon. Figure 3.1-(b) shows a flexible one-dimensional object discretized by several line segments and in Figure 3.1-(c), the same flexible 1D object is represented by a spline curve, its control polygon and an open uniform knot vector [1] - this guarantees that the spline interpolates the first and last control points, thereby ensuring that the ends of the spline match the ends of the flexible 1D object. The number of segments in the control polygon is typically much less than the number of linear segments used to realistically discretize the flexible 1D object($L^2$ error is below a threshold) - in the illustration the flexible one-dimensional object is discretized by 16 linear segments but the control polygon shown in Figure 3.1-(c) has only 7 segments. However, it is well-known that as the control polygon changes, the spline curve and its length changes [1]. This is illustrated in Figure 3.2: the left most spline curve of length $L_{C_1}$ is generated by the control polygon $CP_1$ and as the sides of the control polygon is moved to $CP_2$ keeping $L_{CP_1} = L_{CP_2}$, one can clearly see that the length of the spline curve changes. It can be observed from Figure 3.2 that as the angle between the adjacent segments decrease the length of the spline decreases and the upper

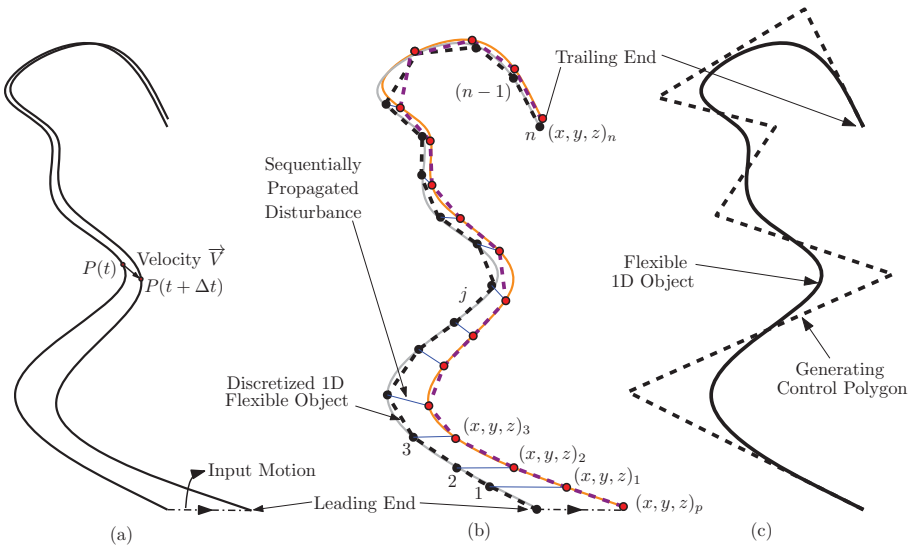



Figure 3.1: (a)One dimensional flexible object, (b) Tractrix based motion planning and (c) Generating control polygon

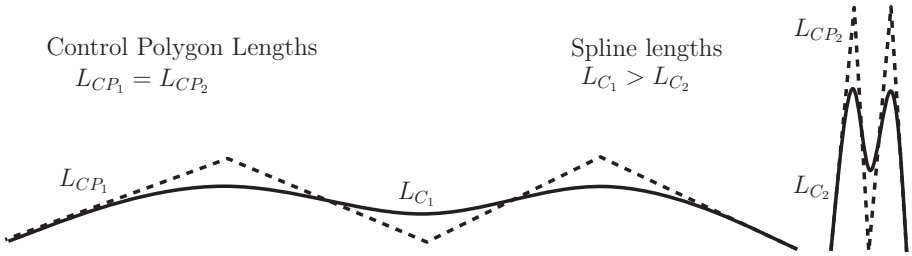



Figure 3.2: Spline length with length preserving transformations of control polygon

bound of curve length is the length of the control polygon. We provide mathematical proofs of these two observations next, first for a quadratic B-spline and then for a cubic B-spline. Finally, arguments for any higher degree 3D splines are provided.

## Quadratic spline

For a quadratic spline shown in Figure 3.3, the three consecutive points $P_1$, $P_2$ and $P_3$ always lie on a plane. It may be noted that the analysis is not restricted to planar quadratic splines as the next three points can lie on a *different* plane and the entire curve can be spatial. Without loss of generality, the coordinates of the three points can be assumed to be $[L_1, 0]^T$, $[0, 0]^T$ and $[L_2 \cos\theta, L_2 \sin\theta]^T$, respectively, where $L_1$, $L_2$ are the lengths of the two sides of the control polygon and $\theta$ is the angle between the adjacent sides of the control polygon. The set of control points $(P_1, P_2, P_3)$ generate the part of the spline shown in Figure 3.3 for the parameter interval $u \in (u_i, u_{i+1})$. The length of the spline curve for $u \in (u_i, u_{i+1})$, is given by

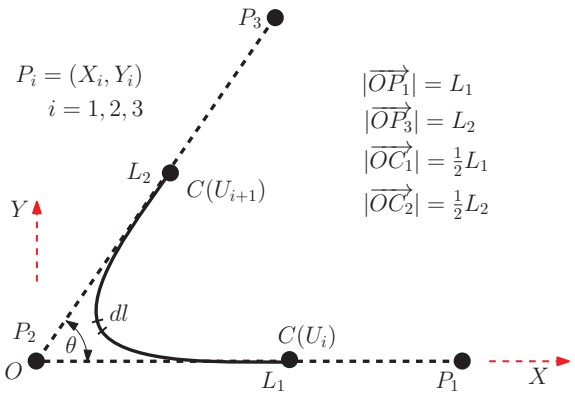



Figure 3.3: Two segments of the control polygon of a quadratic spline

$$l(\theta) = \int_{u_i}^{u_{i+1}} \left( \left( \sum_{i=1}^{3} \frac{dN_{i,p}(u)}{du} X_i \right)^2 + \left( \sum_{i=1}^{3} \frac{dN_{i,p}(u)}{du} Y_i \right)^2 \right)^{\frac{1}{2}} du, \tag{3.1}$$

where $l(\theta)$ means that the spline curve length depends on the included angle $\theta$.

In an open-uniform knot vector of the form $[u_1\ u_1\ u_1\ u_2\ u_3 \ldots u_{m-1}\ u_m\ u_m\ u_m]$ with $u_1 \leq u_2 \leq u_3 \leq \cdots \leq u_m$, without loss of generality, an intermediate knot interval $(u_i, u_{i+1})$ $(i \neq 1, 2, m-2, m-1)$ can be reduced to $(0, 1)$ by appropriate scaling and translation of parameter $u$. The interpolation functions $N_{i,p}$ for a quadratic spline with $p = 2$ and for the knot interval $(0, 1)$ are given by

$$N_{1,2} = \frac{1}{2}(1-u)^2,\ N_{2,2} = -u^2 + u + \frac{1}{2} \text{ and } N_{3,2} = \frac{1}{2}u^2. \tag{3.2}$$

Using the above, $l(\theta)$ can be simplified to

$$l(\theta) = \int_0^1 \sqrt{(-L_1(1-u) + L_2 u\cos\theta)^2 + (L_2 u \sin\theta)^2}\ du. \tag{3.3}$$

and for $L_1 = L_2 = L$, $l(\theta)$ is given by

$$l(\theta) = \frac{1}{\sqrt{32L^2(1+\cos\theta)}} \left( \sqrt{8L^4(1+\cos\theta)} + L^2(-1+\cos\theta) \ln \frac{\sqrt{2} - \sqrt{1+\cos\theta}}{\sqrt{2} + \sqrt{1+\cos\theta}} \right),\ 0 < \theta < \pi. \tag{3.4}$$

From above, $\lim_{\theta \to \pi} l(\theta) = L$ and this agrees with known result for a quadratic curve (see pp. 82

in [1]). From Equation (3.4), $(L - l(\theta))$ is maximum when $\theta \to 0$ and the maximum difference is 50% (see Figure 3.6).

For the general case of a quadratic spline with $l_1 \neq l_2$, the expression for the curve length is more complicated and is given by

$$
\begin{aligned}
l(\theta) &= \int_0^1 dl = \\
&\sqrt{l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta} \left( \frac{(l_1^5 + 2l_1^3 l_2^2 + 2l_1^2 l_2^3 + l_2^5) + (l_1^3 l_2^2 + l_1^2 l_2^3)\cos 2\theta}{2(l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta)^{\frac{5}{2}}} \right) \\
&- \frac{l_1^2 l_2^2 (l_1^2 + l_2^2)\sin^2\theta}{2(l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta)^{\frac{5}{2}}} \ln \frac{-l_1^2 - l_1 l_2 \cos\theta + l_1\sqrt{l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta}}{l_2^2 + l_1 l_2 \cos\theta + l_2\sqrt{l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta}} \\
&+ \frac{l_1 l_2 \cos\theta (3l_1^3 + l_1 l_2^2 + l_2(l_1^2 + 3l_2^2))\sqrt{l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta}}{2(l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta)^{\frac{5}{2}}} \\
&+ \frac{2l_1^3 l_2^3 \sin^2\theta \cos\theta}{2(l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta)^{\frac{5}{2}}} \ln \frac{l_2^2 + l_1 l_2 \cos\theta + l_2\sqrt{l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta}}{-l_1^2 - l_1 l_2 \cos\theta + l_1\sqrt{l_1^2 + l_2^2 + 2l_1 l_2 \cos\theta}}, \ 0 < \theta < \pi.
\end{aligned}
\tag{3.5}
$$

For case when $l_1 \neq l_2$ and $\theta \to 0$ (the curve folding and overlapping with itself), the curve length is given by $\lim_{\theta\to 0} l(\theta) = \dfrac{l_1^2 + l_2^2}{2(l_1 + l_2)}$. In the case of $l_1 \neq l_2$ and $\theta \to \pi$ (curve straightening out to a line), we have $\lim_{\theta\to\pi} l(\theta) = (1/2)(l_1 + l_2)$, which is another well known result for a quadratic b-spline curve (see pp. 78, [1]). From the analytical expression of $l(\theta)$ in equation (3.5), the maximum and minimum curve length occurs when $\theta \to \pi$ and when $\theta \to 0$, respectively.

In general, for a quadratic spline with $n$ segments in the control polygon, the total curve length can be computed as

$$
l_c = \left( \frac{1}{2}(l_s + l_e) + \sum_{i=1}^{n-1} l_i(\theta_i) \right), \tag{3.6}
$$

where $l_s$, $l_e$ denotes length of starting and ending segment of the control polygon, respectively, $l_i$ denotes the length of the $i^{\text{th}}$ control polygon segment and $l_i(\theta_i)$ denotes the length of the portion of the curve defined by the $i^{\text{th}}$, $(i+1)^{\text{th}}$ and $(i+2)^{\text{th}}$ control points.

## Cubic splines

In the case of the quadratic spline, the three points generating the spline define a plane and locally the spline is planar in a knot interval. This is not valid for a cubic spline since the four generating points and the resulting cubic spline need not be planar. Figure 3.4 shows four points $P_1$, $P_2$, $P_3$, $P_4$ and the two included angles $\theta_1$ and $\theta_2$ between the three segments of a

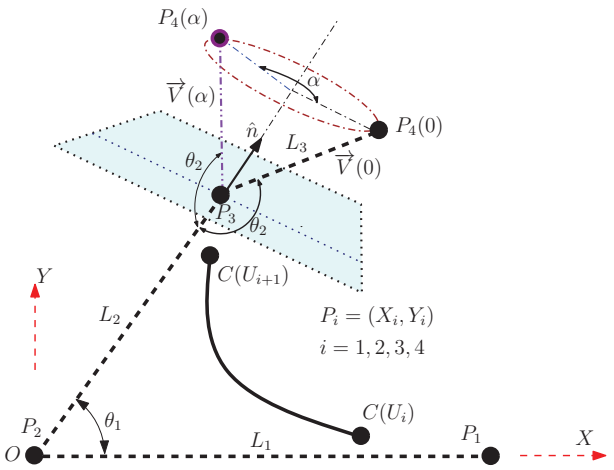



Figure 3.4: Three segments of the control polygon and planarity

control polygon for a cubic spline. The first three points define a plane and the fourth point can lie anywhere on a cone with apex at $P_3$ and slant length $|\boldsymbol{P}_3\boldsymbol{P}_4|$ equal to $l_3$. We denote the vector on the surface of the cone along $\boldsymbol{P}_3\boldsymbol{P}_4$ by $\boldsymbol{V}(\alpha)$, where $\alpha$ is the angle of rotation about an unit vector $\hat{\boldsymbol{n}}$ along the cone axis lying along $\boldsymbol{P}_2\boldsymbol{P}_3$. For $\alpha = 0$, the point $P_4$, denoted by $P_4(0)$, lies on the plane formed by $P_1$, $P_2$ and $P_3$ and the vector from $P_3$ to $P_4(0)$ is denoted by $\boldsymbol{V}(0)$. From Figure 3.4, the angle between $\hat{n}$ and $\boldsymbol{V}(0)$ is seen to be $\pi - (\theta_1 + \theta_2)$ and we can write

$$\boldsymbol{V}(0) = [-L_3 \cos(\theta_1 + \theta_2) \;\; - L_3 \sin(\theta_1 + \theta_2) \; 0]^T . \tag{3.7}$$

The vector $\boldsymbol{V}(\alpha)$ can be obtained as

$$\boldsymbol{V}(\alpha) = R(\alpha)\boldsymbol{V}(0), \tag{3.8}$$

where $R(\alpha)$ is a rotation matrix given by the Rodrigues' rotation formula

$$R(\alpha) = I_3 + (\sin\alpha)K + (1 - \cos\alpha)K^2, \tag{3.9}$$

with $I_3$ denoting a $3 \times 3$ identity matrix and $K$ given by

$$K = \begin{bmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{bmatrix},$$

is a skew-symmetric matrix which represents the cross product operation with vector $\hat{\boldsymbol{n}} = [n_1, n_2, n_3]^T$.

From Figure 3.4, the vector $\hat{\boldsymbol{n}}$ is given by $[\cos\theta_1, \sin\theta_1, 0]^T$ and we can write the vector locating the point $P_4(\alpha)$ with respect to the origin of the coordinate system $O$ (or $P_2$) as

$$\boldsymbol{P}_4(\alpha) = \boldsymbol{P}_3 + R(\alpha)\boldsymbol{V}(0) = \boldsymbol{P}_4(0) + \boldsymbol{\Delta P}_4(\alpha), \tag{3.10}$$

where

$$\boldsymbol{\Delta P}_4(\alpha) = \begin{bmatrix} L_3 \sin\theta_1 \sin\theta_2 (\cos\alpha - 1) \\ -L_3 \cos\theta_1 \sin\theta_2 (\cos\alpha - 1) \\ -L_3 \sin\theta_2 \sin\alpha \end{bmatrix}.$$

Using the above, the expression for the spline can be written as

$$\boldsymbol{C}(u,\alpha) = N_{1,3}(u)\boldsymbol{P}_1 + N_{2,3}(u)\boldsymbol{P}_2 + N_{3,3}(u)\boldsymbol{P}_3 + N_{4,3}(u)\left(\boldsymbol{P}_4(0) + \boldsymbol{\Delta P}_4(\alpha)\right), \tag{3.11}$$

and its derivative can be written as

$$\begin{aligned} \boldsymbol{C'}(u,\alpha) = {} & N'_{1,3}(u) \begin{bmatrix} L_1 \\ 0 \\ 0 \end{bmatrix} + N'_{3,3}(u) \begin{bmatrix} L_2 \cos\theta_1 \\ L_2 \sin\theta_1 \\ 0 \end{bmatrix} \\ & + N'_{4,3}(u) \left( \begin{bmatrix} L_2 \cos\theta_1 - L_3 \cos(\theta_1 + \theta_2) \\ L_2 \sin\theta_1 - L_3 \sin(\theta_1 + \theta_2) \\ 0 \end{bmatrix} + \begin{bmatrix} L_3 \sin\theta_1 \sin\theta_2 (\cos\alpha - 1) \\ -L_3 \cos\theta_1 \sin\theta_2 (\cos\alpha - 1) \\ -L_3 \sin\theta_2 \sin\alpha \end{bmatrix} \right). \end{aligned} \tag{3.12}$$

When the spline lies in a plane and $\boldsymbol{\Delta P}_4(0) = 0$, the spline can be written as

$$\boldsymbol{C}(u,0) = N_{1,3}(u)\boldsymbol{P}_1 + N_{2,3}(u)\boldsymbol{P}_2 + N_{3,3}(u)\boldsymbol{P}_3 + N_{4,3}(u)\boldsymbol{P}_4(0), \tag{3.13}$$

and the derivative can be written as

$$\boldsymbol{C'}(u,0) = N'_{1,3}(u) \begin{bmatrix} L_1 \\ 0 \\ 0 \end{bmatrix} + N'_{3,3}(u) \begin{bmatrix} L_2 \cos\theta_1 \\ L_2 \sin\theta_1 \\ 0 \end{bmatrix} + N'_{4,3}(u) \begin{bmatrix} L_2 \cos\theta_1 - L_3 \cos(\theta_1 + \theta_2) \\ L_2 \sin\theta_1 - L_3 \sin(\theta_1 + \theta_2) \\ 0 \end{bmatrix}. \tag{3.14}$$

The square of the elemental length of the cubic spline is given by

$$\begin{aligned} dl^2(\alpha) = \boldsymbol{C'}(u,\alpha)^T \boldsymbol{C'}(u,\alpha) du^2 = dl^2(0) \\ + 2N'_{4,3}(u) L_3 \sin\theta_1 \sin\theta_2 (\cos\alpha - 1) X'(u,0) + \left(N'_{4,3}(u) L_3 \sin\theta_1 \sin\theta_2 (\cos\alpha - 1)\right)^2 \\ - 2N'_{4,3}(u) L_3 \cos\theta_1 \sin\theta_2 (\cos\alpha - 1) Y'(u,0) + \left(-N'_{4,3}(u) L_3 \cos\theta_1 \sin\theta_2 (\cos\alpha - 1)\right)^2 \\ + \left(-N'_{4,3}(u) L_3 \sin\theta_2 \sin\alpha\right)^2, \end{aligned} \tag{3.15}$$

where $(X'(u,0), Y'(u,0))$ denote the $X$ and $Y$ components of $\boldsymbol{C'}(u,0)$ and $Z'(u,0)$, the $Z$ component of $\boldsymbol{C'}(u,0)$ is zero from Equation (3.14). The above equation on simplifying gives

$$dl^2(\alpha) = dl^2(0) + 2N'_{1,3} N'_{4,3} L_1 L_3 \sin\theta_1 \sin\theta_2 (\cos\alpha - 1), \tag{3.16}$$

and by using the Cauchy-Schwarz inequality, we can write

$$dl(\alpha) - dl(0) \leq \sqrt{2N'_{1,3} N'_{4,3} L_1 L_3 \sin\theta_1 \sin\theta_2 (\cos\alpha - 1)}, \tag{3.17}$$

where $(\cos\alpha - 1) \leq 0 \ \forall \alpha \in [0 \ 2\pi]$.

For a cubic spline, the basis functions $N_{1,3}$, $N_{4,3}$ and their derivatives are given by

$$N_{1,3} = \frac{1}{6}(1-u)^3, \ N'_{1,3} = -\frac{1}{2}(1-u)^2, N_{4,3} = \frac{1}{6}u^3, \ N'_{4,3} = \frac{1}{2}u^2, \tag{3.18}$$

and hence $dl(\alpha) - dl(0)$ is real and positive. Additionally, the right-hand side of Equation (3.17) is maximum when $\alpha = \pi$ and minimum when $\alpha = 0$. The length of control polygon remains the same for all $\alpha$ and hence the elemental length difference between the cubic spline curve and the control polygon becomes maximum when $\alpha = 0$. This proves the assertion that the worst case difference between the length of the control polygon and the cubic spline is when $\alpha = 0$ (when the four control points lie on a plane) and we do not need to consider the four points in 3D space. Hence, we consider the case of a planar cubic spline and obtain expressions for bounds on this difference.

Figure 3.5 shows the four control points $P_i, \ i = 1, ..., 4$ on a plane and the two included angle $\theta_1$, $\theta_2$ between the first and second, second and third segments, respectively. The initial configuration of the knot vector is assumed to be a clamped open-uniform knot vector of the form $[u_1 \ u_1 \ u_1 \ u_1 \ u_2 \ u_3 \ldots u_{m-1} \ u_m \ u_m \ u_m \ u_m]$, $u_1 \leq u_2 \leq \cdots \leq u_m$. Note that for a uniform non-repeated knot vector, the difference between the curve length and control polygon length is maximum as any repetition of knot vector or subdivision pulls the curve towards the control

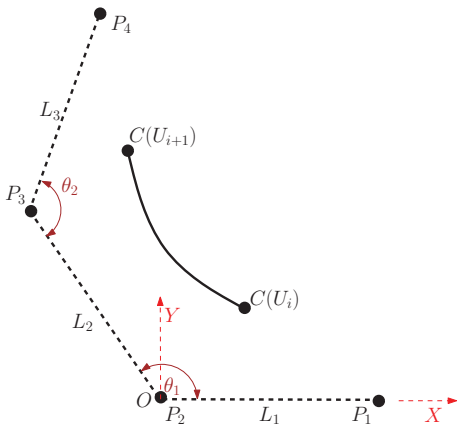



Figure 3.5: Control polygon for a planar cubic spline

polygon thereby reducing the length difference.

From the figure, the points of the spline are $\boldsymbol{P}_1 = [L_1, 0]^T$, $\boldsymbol{P}_2 = [0, 0]^T$, $\boldsymbol{P_3} = [L_2 \cos\theta_1, L_2 \sin\theta_1]^T$ and $\boldsymbol{P_4} = [L_2 \cos\theta_1 - L_3 \cos(\theta_1 + \theta_2), L_2 \sin\theta_1 - L_3 \sin(\theta_1 + \theta_2)]^T$. The elemental length of the cubic spline $\boldsymbol{C_0}(u)$ is given by

$$dl = \left( \left( \sum_{i=1}^{n} \frac{dN_{i,p}(u)}{du} X_i \right)^2 + \left( \sum_{i=1}^{n} \frac{dN_{i,p}(u)}{du} Y_i \right)^2 \right)^{\frac{1}{2}} du, \tag{3.19}$$

where $(X_i, Y_i)$ are the coordinates of $P_i$, $i = 1, ..., 4$.

Substituting the $X$ and $Y$ coordinates of the points on the control polygon and using $N_i'$ to denote $\dfrac{dN_{i,3}(u)}{du}$, we get

$$\begin{aligned} dl &= \sqrt{A + B}\, du, \\ A &= \left( \frac{\partial X}{\partial u} \right)^2 = \left( N_1' L_1 + N_3' L_2 \cos\theta_1 + N_4' \left( L_2 \cos\theta_1 - L_3 \cos\left(\theta_1 + \theta_2\right) \right) \right)^2 \\ B &= \left( \frac{\partial Y}{\partial u} \right)^2 = \left( N_3' L_2 \sin\theta_1 + N_4' \left( L_2 \sin\theta_1 - L_3 \sin\left(\theta_1 + \theta_2\right) \right) \right)^2. \end{aligned} \tag{3.20}$$

For a cubic spline, the basis functions in $u \in [0\ 1]$ are

$$\begin{aligned} N_{1,3} &= \frac{1}{6}(1-u)^3, \ N_{2,3} = \frac{2}{3} + \frac{1}{2}u^3 - u^2, \\ N_{3,3} &= \frac{1}{6} + \frac{1}{2}u - \frac{1}{2}u^3 + \frac{1}{2}u^2 \ \text{ and } N_{4,3} = \frac{1}{6}(1-u)^3. \end{aligned} \tag{3.21}$$

Substituting the above in Equation (3.20), we get

$$
\begin{aligned}
dl &= \sqrt{Pu^4 + Qu^3 + Ru^2 + Su + T}\; du, \\
P &= \frac{1}{4}L_1^2 + \frac{1}{4}L_3^2 + L_2^2 + L_1L_2\cos\theta_1 + \frac{1}{2}L_1L_3\cos(\theta_1+\theta_2) + L_2L_3\cos\theta_2, \\
Q &= -L_2L_3\cos\theta_2 - 3L_1L_2\cos\theta_1 - L_1L_3\cos(\theta_1+\theta_2) - L_1^2 - 2L_2^2, \\
R &= \frac{3}{2}L_1^2 + \frac{5}{2}L_1L_2\cos\theta_1 + \frac{1}{2}L_1L_3\cos(\theta_1+\theta_2) - \frac{1}{2}L_2L_3\cos\theta_2, \\
S &= L_2^2 - L_1^2,\; T = \frac{1}{4}L_1^2 + \frac{1}{4}L_2^2 - \frac{1}{2}L_1L_2\cos\theta_1,
\end{aligned}
\tag{3.22}
$$

and the length of the spline generated in the knot interval [0 1] can be obtained as the integral of the right-hand side in Equation (3.22). Unlike in the quadratic case, for a cubic spline, the analytical form of the integral as a function of $\theta_1$ and $\theta_2$ is not known and it is not possible to find analytical expressions for the difference between the total length of the control polygon and B-spline curve length. The length difference is a surface (being a function of $\theta_1$ and $\theta_2$) and can always be found using numerical integration. We can, however, obtain useful approximations to the integral by considering the following:

- We consider the effect of change of one angle at a time, i.e., $\theta_1$ is varied with $\theta_2$ held constant. This is a reasonable since during the motion of the control polygon, the angles between the adjacent segments are monitored. When the difference in length between spline and control polygon due to a single angle change becomes large, subdivision is used to reduce the difference (see Section 3.3) and hence effect of change in one included angle can be considered.

- We consider the case of a control polygon with equal lengths, i.e., $L_1 = L_2 = L_3 = L$.

With the above two assumptions, the integral simplifies to

$$
\begin{aligned}
l(\theta_1) &= \frac{L}{\sqrt{2}}\int_0^1 \sqrt{1 + (pu^4 + qu^3 + ru^2 - \cos\theta_1)}\; du, \\
\text{where}\quad p &= \cos(\theta_1+\theta_2) + 2\cos\theta_1 + 2\cos\theta_2 + 3, \\
q &= -2\cos(\theta_1+\theta_2) - 6\cos\theta_1 - 2\cos\theta_2 - 6 \text{ and} \\
r &= \cos(\theta_1+\theta_2) + 5\cos\theta_1 - \cos\theta_2 + 3.
\end{aligned}
\tag{3.23}
$$

Since $0 \le u \le 1$, the term $|W| = |pu^4 + qu^3 + ru^2 - \cos\theta_1| \le 1$. Hence, we can approximate

$(1+W)^{\frac{1}{2}}$ as $\left(1+\frac{1}{2}W\right)$ by keeping only the first term in the binomial expansion. The first-order length difference between the control polygon and the curve, in the knot interval [0 1] due to a change in $\theta_1$ (with $\theta_2$ held constant) is given by

$$\begin{aligned} e_{\text{first-order}}(\theta) &= \frac{L}{2\sqrt{2}} \int_{\theta}^{\pi} \int_{0}^{1} \frac{\partial W}{\partial \theta_1} d\theta_1 du \\ &= \frac{1}{120} L\sqrt{2} \left(13\cos\theta - \cos\theta_2 - \cos(\theta+\theta_2) + 13\right), \end{aligned} \tag{3.24}$$

where $e_{\text{first-order}}(\theta)$ denotes the length difference when $\sqrt{1+W}$ is approximated by $1+(1/2)W$. The plot of the exact length difference obtained by numerically integrating right-hand side of Equation (3.22) and the plot of the first-order approximation is shown in Figure 3.6. For comparison, the length difference obtained for a quadratic spline is also shown in the figure.

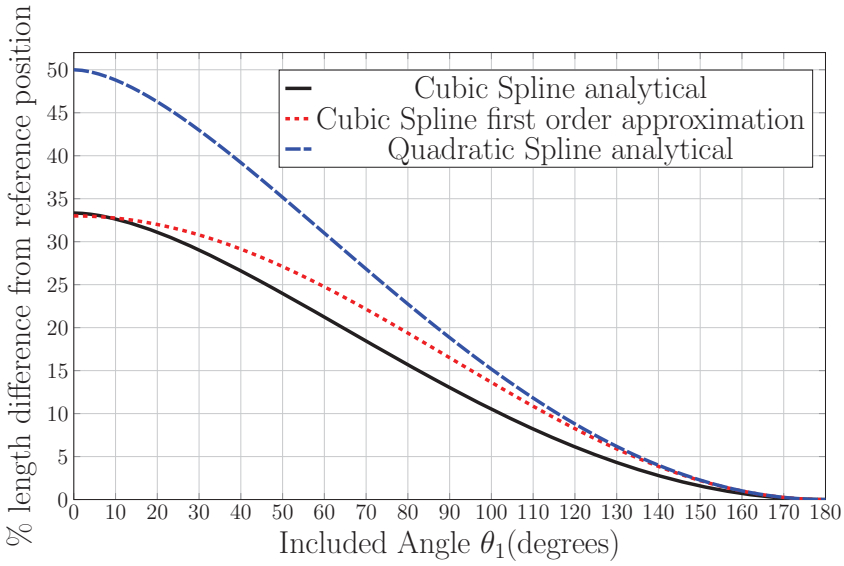



Figure 3.6: Plot of bound on length difference for a quadratic and cubic B-spline

## Summary

We can summarize the results obtained from the quadratic and cubic B-splines as follows:

- For a quadratic B-spline, we can obtain closed-form analytic expression for the curve length as a function of the angle between the adjacent segments. For any angle $\theta$ the change in curve length from the completely stretched out case ($\theta = \pi$) can be obtained

using Equation (3.5) as

$$e_{\text{quadratic B-spline}} = l(\pi) - l(\theta) = \left( \frac{L_1 + L_2}{2} - l(\theta) \right). \tag{3.25}$$

- The difference between the length of the control polygon, $L_{CP} = \sum_{i=1}^{n} L_i$, and the length of the curve $L_C$ is given by

$$E_C = L_{CP} - L_C = \sum_{i=1}^{n} L_i - \left( \frac{1}{2}(L_s + L_e) + \sum_{i=1}^{n-1} l_i(\theta_i) \right). \tag{3.26}$$

- For a cubic B-spline, there is no known closed-form analytic expression for the curve length as a function of the two angles between the three consecutive segments. The length of the curve can be obtained by integrating Equation (3.22) and a numerical plot of change in length with respect to one or both angles can be obtained.

- A first-order approximation of the difference between the length of cubic spline and the length of the generating control polygon, with one angle held constant and equal control polygon leg lengths, can be obtained as shown in Equation (3.24). The first-order approximation is conservative to within a maximum of 3.73% difference from the exact length difference obtained from integration.

- It can be seen by comparing the plots in Figure 3.6 that a cubic spline gives less length difference when the angle between two adjacent segments change – for a $\theta$ value of 100° the length difference for a quadratic spline is 15% where as for a cubic spline it is 10%. A consequence of this result is that less number of subdivisions (see Section 3.3) may be required during the motion of the flexible 1D object when it is modeled with cubic splines.

- Although the plots in Figure 3.6 are for equal lengths, they serve as a design tool whereby the user can choose a threshold angle to limit the change in curve length within desirable limits.

- When a curve is modeled using cubic splines, $\mathcal{C}^2$(curvature) continuity is guaranteed. Majority of real-world problems demand a maximum of curvature continuity or lesser and hence, the results have been obtained only for a quadratic and cubic spline. However, first-order results along similar lines as the cubic spline can also be developed for higher-order splines.

One key consequence of the above results is that as the control polygon is moved the angle between adjacent segments will change and the length of the spline curve will change. The key idea of length preserving motion of the flexible 1D object will not be possible. In the next Section, we present an adaptive algorithm to approximately preserve the length of the spline curve to within a user specified tolerance when the control polygon is moved.

# Approximate length preservation in splines

As discussed in Section 3.2, error in length of the B-spline curve from an initial value can be related to the included angle $\theta$ between two adjacent segments. The key idea in approximate length preservation is to sub-divide a segment of the control polygon when the included angle between two adjacent segments is less than a threshold value and merge the two adjacent segments when the included angle is larger than another threshold value. In the subdivision

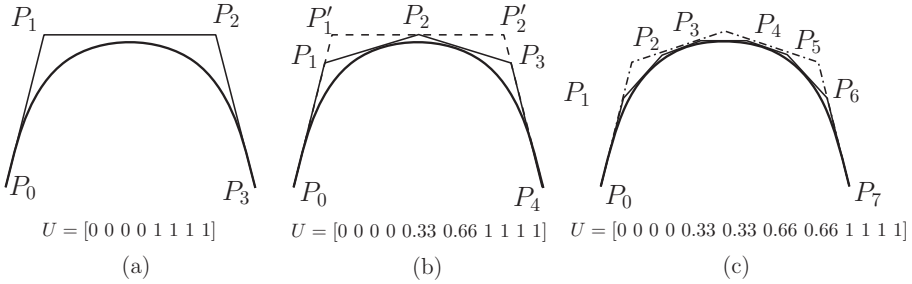



Figure 3.7: Subdivision and merging in splines

step, the control polygon is modified by inserting control points as illustrated in Figure 3.7. As shown, the segment $P_1P_2$ in Figure 3.7 (a) is replaced by two segments $P_1P_2$ and $P_2P_3$ in Figure 3.7 (b) with the original segment $P_1P_2$ shown as $P_1'P_2'$. From the construction and using triangle inequality in Figure 3.7 (b), $|P_1P_1'| + |P_1'P_2| \leq |P_1P_2|$ and $|P_2P_2'| + |P_2'P_3| \leq |P_2P_3|$. Hence the length of control polygon $P_0P_1P_2P_3P_4$ after subdivision is less than the length of the original control polygon $P_0P_1P_2P_3$. If the segments $P_1P_2$ and $P_2P_3$ are further subdivided, as in Figure 3.7 (c), the total length of the modified control polygon will decrease even more. In the limit of infinite subdivisions, the length of control polygon will coincide with the length of the curve.

One effect of subdivision is that the number of control points (and the number of segments) monotonically increases over time depending on the extent of bending/warping of the control polygon during the motion and this increases the computation requirement in the tractrix based motion algorithm. To overcome this problem, we reduce the number of segments in the control polygon when parts of the control polygon stretch out and the included angle is larger than a

pre-defined threshold. The reduction in the number of segments is schematically opposite of subdivision – the sequence for merging is from right to left in Figure 3.7. Note that during merging the length of the resulting control polygon increases.

We discuss in detail and present mathematical results for subdivision and merging in the rest of this section.

## Subdivision in splines

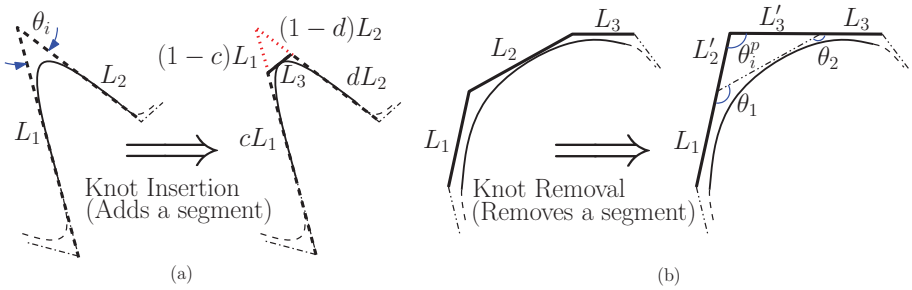



Figure 3.8: Knot insertion and knot removal

Figure 3.8 (a) shows subdivision (also called knot insertion). Assuming the control polygon segments are of lengths $L_1$, $L_2$ and the included angle is $\theta_i$, the length of the portion of control polygon before subdivision ($L_{CP_0}$) and the length after subdivision ($L_{CP_1}$) is given by

$$L_{CP_0} = L_1 + L_2 \text{ and } L_{CP_1} = cL_1 + dL_2 + L_3 \ (0 < (c, d) < 1), \tag{3.27}$$

where by using law of cosines

$$L_3 = \sqrt{((1-c)L_1)^2 + ((1-d)L_2)^2 - 2(1-c)(1-d)L_1L_2\cos\theta_i},$$

and $c$, $d$ are the ratios for subdivision which can be chosen by user. Hence, the decrease in length of the control polygon after subdivision is given by

$$\begin{aligned}\Delta L_{CP} &= L_{CP_0} - L_{CP_1} \\ &= (1-c)L_1 + (1-d)L_2 - \sqrt{((1-c)L_1)^2 + ((1-d)L_2)^2 - 2(1-c)(1-d)L_1L_2\cos\theta_i}.\end{aligned} \tag{3.28}$$

From the above, as the portion of the control polygon stretches out and the included angle $\theta_i \to \pi$, the change in length of the control polygon $\lim_{\theta_i \to \pi} \Delta L_{CP} \to 0$.

During subdivision, the length of the spline curve remains the *same*. As mentioned earlier, the length of the control polygon, $L_{CP}$, is more than the length of the spline curve, $L_C$, and we

can write

$$L_{CP} = L_C + e, \tag{3.29}$$

where $e \geq 0$ is the difference in length between the control polygon and the spline curve. During subdivision, the length of the spline curve is not changed, and from Equations (3.28) and (3.29), we get

$$L_{CP_1} \leq L_{CP_0} \text{ and } e_1 \leq e_0, \tag{3.30}$$

where $e_0$ and $e_1$ are the length differences before and after subdivision, respectively.

The above proves that the length difference between spline curve and control polygon decreases during a subdivision. Most importantly, the angles between the adjacent segments of the control polygon increases, and, as shown in Figure 3.6, increasing the angle reduces the difference in the length between the spline curve and the control polygon. Hence, through subdivision, it is possible to control spline length by setting a threshold angle value $\theta_{th}$. If the angle between any two segments is less than this threshold ($\theta_i \leq \theta_{th}$), then the control polygon is subdivided to obtain two new control points. As $\theta_{th}$ increases, length difference on any elemental segment reduces which in turn reduces the total difference in length. Hence the threshold angle plays a key role in the total difference between length of the control polygon and curve. From the analytical results in the previous section and from Figure 3.6, a suitable $\theta_{th}$ can be chosen to satisfy a desired length error requirement. In extensive simulations (see Section 3.4), it is observed that a threshold angle of 140° gives total difference less than 5%.

## Merging in splines

When two adjacent segments of a control polygon are merged (by knot removal), the number of segments in the control polygon will reduce and the computations required for the tractrix based motion algorithm will reduce. Knot removal is shown schematically in Figure 3.8 (b). As mentioned earlier, in case of knot removal, the length of the control polygon increases and the length before and after knot removal can be written as

$$L_{CP_0} = L_1 + L_2 + L_3 \text{ and } L_{CP_1} = L_1 + L_2' + L_3' + L_3. \tag{3.31}$$

But we know that $\theta_i^p = (\theta_1 + \theta_2) - \pi$ and by using the law of sines,

$$L_2' = -\frac{\sin\theta_2}{\sin(\theta_1 + \theta_2)} L_2 \text{ and } L_3' = -\frac{\sin\theta_1}{\sin(\theta_1 + \theta_2)} L_2. \tag{3.32}$$

From Equation (3.31) and Equation (3.32), the increase in length of the control polygon can be computed as

$$\Delta L_{CP} = L_{CP_1} - L_{CP_0} = -\frac{\sin\theta_2}{\sin(\theta_1+\theta_2)}L_2 - \frac{\sin\theta_1}{\sin(\theta_1+\theta_2)}L_2 - L_2. \tag{3.33}$$

Unlike subdivision, the increase in the length of the control polygon is a function of two angles $\theta_1$ and $\theta_2$ and thus lies on a surface. When $(\theta_1, \theta_2) \to \pi$, $\lim_{(\theta_1,\theta_2)\to\pi} \Delta L_{CP} \to 0$ and we can infer that knot removal should be done when the $\theta_1$ and $\theta_2$ are close to $\pi$.

Knot removal has the additional complexity of not yielding a unique solution/curve (see pp. 179 in [1] for further details). As shown in Figure 3.9, there are two possible solutions for the merged spline curves, which can be derived from the original spline (black). The length of the two spline curves can be computed by integration, and are denoted by $L_{C_1}$ and $L_{C_2}$. We denote the control polygon and curve length before merging by $L_{CP}$ and $L_C$, respectively. Since merging increases the length of the control polygon, we can write

$$L_{CP_1} \geq L_{CP},\ L_{CP_2} \geq L_{CP},\ L_{CP_1} = L_{C_1} + e_1,\ L_{CP_2} = L_{C_2} + e_2, \tag{3.34}$$

where the two control lengths are denoted by $L_{CP_1}$, $L_{CP_2}$, and the difference between the curve length and the control polygon length are denoted by $e_1$ and $e_2$. As the length of the two control polygons is known, the terms $e_1$ and $e_2$ can be computed and the solution with lower difference between the spline and control polygon length is chosen as the new curve definition. For a knot of multiplicity 2 to be removed once, the condition to be satisfied is that two points

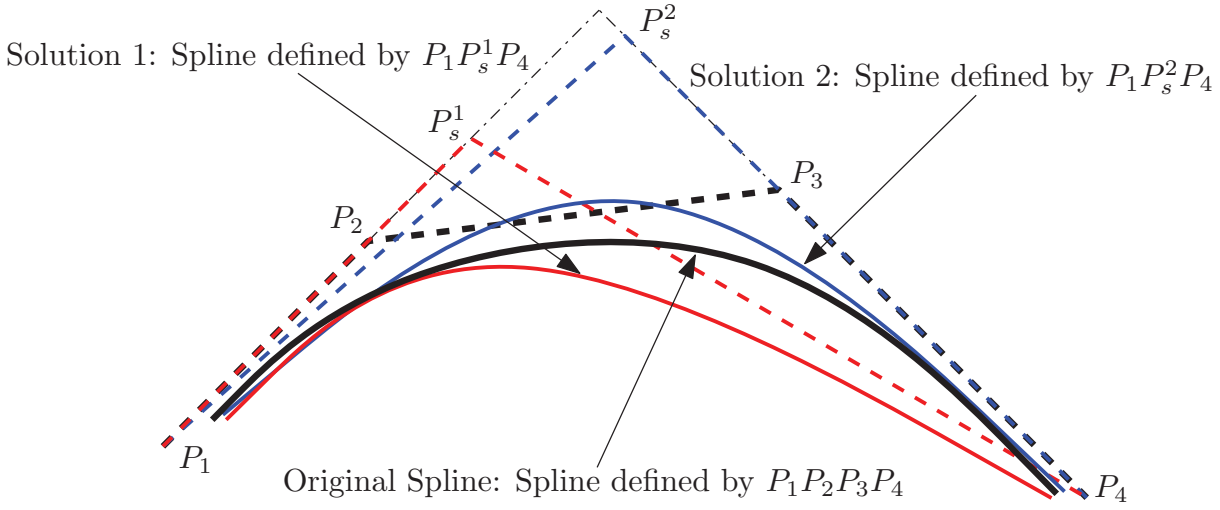



Figure 3.9: Illustration of multiple solutions encountered in merging process

given below

$$\begin{aligned} \boldsymbol{P_s^1} &= \frac{\boldsymbol{P_2} - (1-\alpha_1)\boldsymbol{P_1}}{\alpha_1} \text{ and} \\ \boldsymbol{P_s^2} &= \frac{\boldsymbol{P_3} - \alpha_2 \boldsymbol{P_4}}{1-\alpha_2}, \end{aligned} \tag{3.35}$$

with $\alpha_1 = \dfrac{u_3 - u_1}{u_4 - u_1}$, $\alpha_2 = \dfrac{u_3 - u_2}{u_5 - u_2}$ must coincide [1]. As shown in Figure 3.9, the two points- $\boldsymbol{P_s^1}$ and $\boldsymbol{P_s^2}$, lie on the vector directed along $\boldsymbol{P_1P_2}$ and $\boldsymbol{P_4P_3}$. Substituting for points $\boldsymbol{P_1} = [L_1\ 0]^T$, $\boldsymbol{P_2} = [0\ 0]^T$, $\boldsymbol{P_3} = [L_2\cos\theta_1\ L_2\sin\theta_1]^T$, and $\boldsymbol{P_4} = [L_2\cos\theta_1 - L_3\cos(\theta_1+\theta_2)\ L_2\sin\theta_1 - L_3\sin(\theta_1+\theta_2)]^T$, we get

$$\begin{aligned} \boldsymbol{P_s^1} &= \frac{u_4 - u_3}{u_3 - u_1}\begin{bmatrix} -L_1 \\ 0 \end{bmatrix} \text{ and} \\ \boldsymbol{P_s^2} &= \frac{1}{u_5 - u_3}\begin{bmatrix} L_2\cos\theta_1(u_5-u_3) + L_3\cos(\theta_1+\theta_2)(u_3-u_2) \\ L_2\sin\theta_1(u_5-u_3) + L_3\sin(\theta_1+\theta_2)(u_3-u_2) \end{bmatrix}. \end{aligned} \tag{3.36}$$

Since merging is done only when both the angles $\theta_1$ and $\theta_2$ cross the threshold angle, the limiting case can be taken as $\theta_1 = \theta_2 = \theta_m$. For equi-spaced knots, the above equation simplifies to

$$\boldsymbol{P_s^1} = \begin{bmatrix} -\frac{1}{2}L_1 \\ 0 \end{bmatrix} \text{ and } \boldsymbol{P_s^2} = \begin{bmatrix} L_2\cos\theta_m + \frac{1}{2}L_3\cos 2\theta_m \\ L_2\sin\theta_m + \frac{1}{2}L_3\sin 2\theta_m \end{bmatrix} \tag{3.37}$$

Figure 3.10 shows the two points $\boldsymbol{P_s^1}$ and $\boldsymbol{P_s^2}$ that define the two control polygons. We can

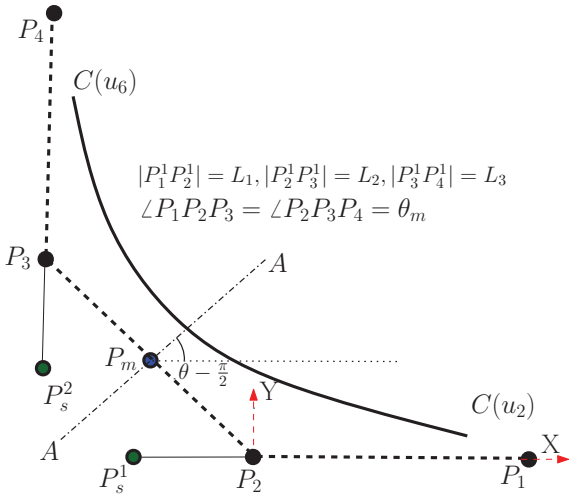
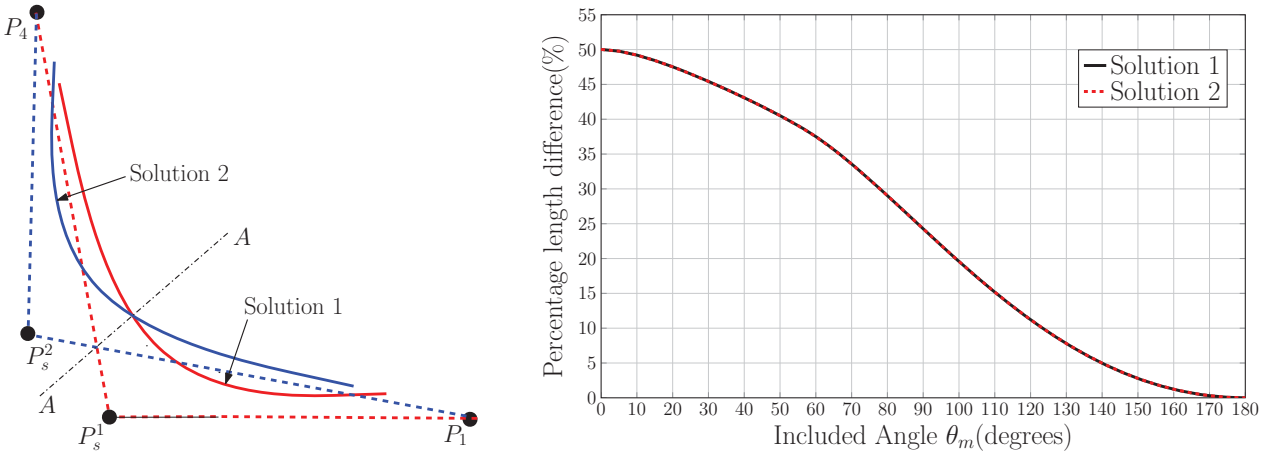



Figure 3.10: Portion of spline to be merged (knot removal)

make the following observations from the above analysis:

- If link lengths $L_1 = L_3 = L$, then in the limiting case mentioned above ($\theta_1 = \theta_2 = \theta_m$), the portion of spline shown in Figure 3.9 is symmetric about an axis $AA$ passing through the midpoint of points $P_2$ and $P_3$. The lengths of the two curves are equal and either of the two curves can be chosen after merging. The two curves and the length difference due to merging as a function of $\theta_m$ are shown in Figure 3.11.




(a) Affine nature of solutions

(b) Percentage length difference plot

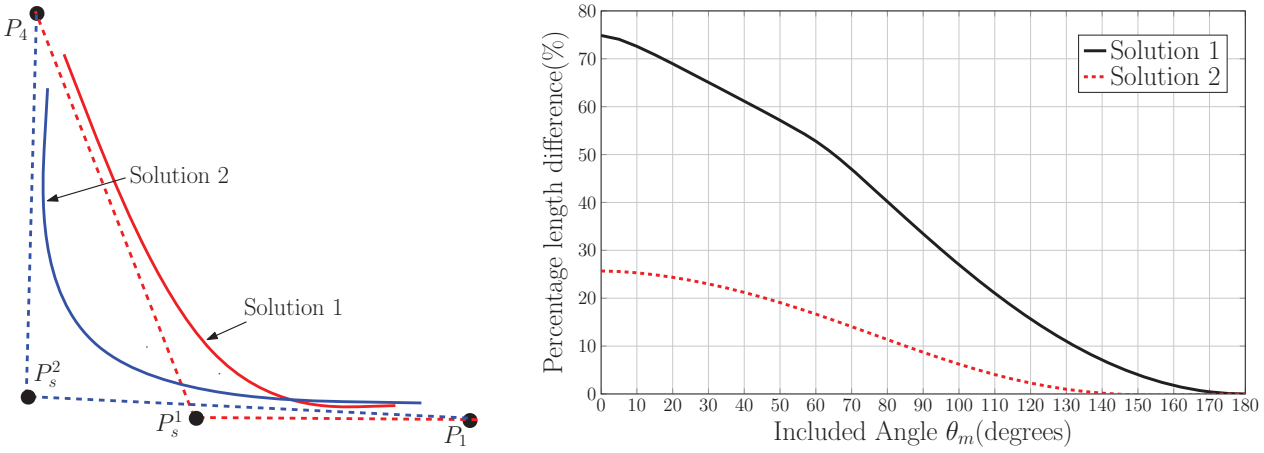
Figure 3.11: Multiple merging solutions when $L_1 = L_3$

- In a more generic case, where $L_1 \neq L_3$, the two solutions are asymmetrical. This is shown in Figure 3.12 for arbitrarily chosen control polygon leg lengths $L_1 = 1$, $L_2 = 3$ and $L_3 = 4$. The difference between the curve length and the control polygon is shown in Figure 3.12(b); during simulations we noted that the smaller of the two differences is chosen.

- Figure 3.12 can be numerically computed and generated for other combination of lengths. Similar to subdivision, the plot of length difference versus $\theta_m$ can be used to choose the threshold angle for merging. From extensive numerical simulations (see also Section 3.4), it was observed that an angle of 160° for merging (knot removal) resulted in a total spline curve length error of less than 5%.

## Algorithm for approximate length preservation

Based on the analysis in Section 3.3, the algorithm followed for approximate length preserving configuration planning for a flexible 1D body is summarized in the flowchart of Figure 3.13.




(a) Asymmetric solutions

(b) Percentage difference plot

Figure 3.12: Multiple merging solutions when in a generic scenario

The inputs to the algorithm are the initial curve configuration, the path (curve) along which the leading end is moved, the velocity of the motion and location of the perturbed point on the curve. Based on these data, the initial control polygon configuration and the open uniform knot vectors are derived and initialized into the algorithm. The output is the motion of the flexible 1D object.

## Steps in the Algorithm

1. Obtain the control polygon by interpolating the vertices/points in the input poly-line. The number of segments in the control polygon is taken to be $n$, degree 3 and a clamped open uniform knot vector is used. The maximum deviation between the curve and the points is set to a predefined threshold for terminating the interpolation algorithm.

2. Initialize length parametrized motion of the leading end and discretized it for smooth visualization of motion.

3. For each incremental motion, starting from leading segment, recursively obtain the motion of all segments using the tractrix equations.

4. Perform threshold crossover checks for each included angle in the control polygon and execute subdivision or knot removal algorithms.

5. Update the new control polygon configurations and knot vectors and continue to next incremental motion of the leading end.

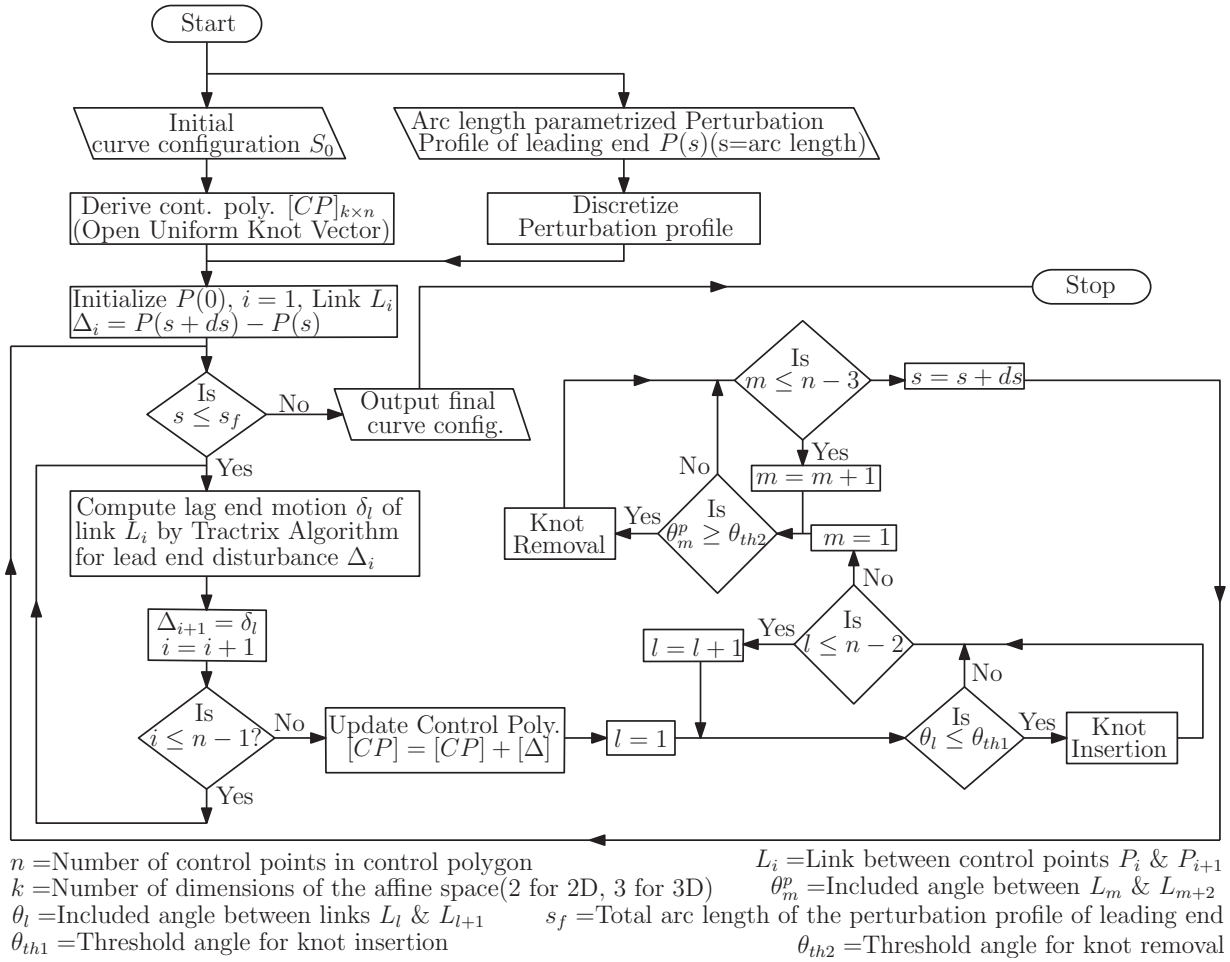



Figure 3.13: Flow chart of the algorithm

6. Repeat steps 3, 4 and 5 till the full motion of leading end is completed.

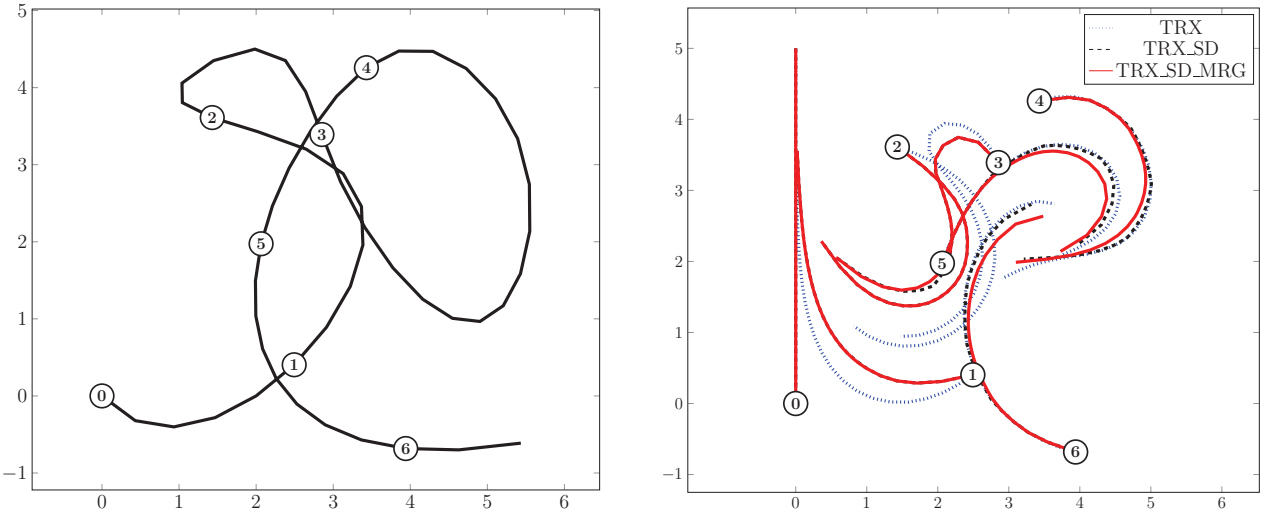
In the next section, we present numerical simulation results illustrating the approach developed in this work.

# Numerical simulation

In this section, we present numerical simulation results for a chosen one-dimensional flexible object with 5 links of unit link length. All simulations were done using the commercial software MATLAB [80] on a PENTIUM quad core PC with 16 Gb RAM running the LINUX operating system. The first two simulations are for an arbitrarily chosen 2D path of the leading end and there are two 3D simulation. We present simulation results for an exact length-preserving tractrix-based algorithm (labeled as TRX), an approximate length preserving tractrix and spline based algorithm with subdivision (labeled as TRX_SD) and finally an approximate length pre-

serving tractrix and spline based algorithm with subdivision and knot removal (labeled as TRX_SD_MRG). In all the simulations the initial configuration of the flexible 1D object is a straight line – there is no restriction on the initial configuration and it is chosen as a straight line to ensure that the *initial* length error between the actual object and the discretized object/curve is zero.

In the first simulation, the leading end is moved along an arbitrarily chosen 2D path in steps of 0.5 length units over 52 steps. The arbitrary path is shown in Figure 3.14 (a). Along the path, seven arbitrary snapshot locations are chosen and these are denoted by ⓪ through ⑥. The configuration of the flexible 1D object at each of the seven snapshot locations of the




(a) 2D path with snapshot locations

(b) Trajectory comparison with different algorithms

Figure 3.14: Simulation a planar curve subjected to generic 2D motion

leading end is shown in Figure 3.14 (b) – as mentioned at the start the flexible 1D object is a straight line (see configuration ⓪ in Figure 3.14 (a)). The configuration of the flexible 1D object computed according to the three algorithms, namely TRX, TRX_SD and TRX_SD_MRG, are shown in Figure 3.14 (b).

In the purely tractrix (TRX) algorithm, the flexible 1D object needs to be discretized into a large number of linear segments to realistically represent the continuous flexible 1D object and to make the motion look smooth. As the number of linear segments, $n$, increases, the computations, as expected, grow in a $\mathcal{O}(n)$ manner – the simulation time grows from 40 seconds for $n = 10$ to about 1680 seconds for $n = 200$. In Figure 3.14(b), the configuration simulations for the tractrix based simulations are done for $n = 20$.

For the approximate length preserving tractrix and spline based algorithms (TRX_SD and

TRX_SD_MRG), the initial number of linear segments in the control polygon is chosen as 6 and the angle threshold for subdivision was chosen as 140°. The threshold angle for removing a control point/knot was chosen as 160°. Snapshots of the curve for different points on the chosen path are shown in Figure 3.14(b) for illustrating the qualitative nature of the results. As mentioned the maximum percentage length difference depends on the chosen threshold angles (140° for subdivision and 160° for knot removal in this simulation) and the number of control points required to ensure that the angle threshold is not exceeded, in this case, turns out to be 20 for subdivision algorithm and 18 for subdivision with merging algorithm as seen from the Figure 3.15. The maximum error introduced by merging at any instant over the whole simulation is 1.8%.

Figure 3.15(b) shows the variation of curve length and Figure 3.15(a) shows number of control points in the control polygon backbone over the simulation duration. As seen, the algorithm adapts to the characteristics of the motion by adding control points as and when required to compensate for the warping of the curve. It also removes control points as and when the curve can be simplified and represented in terms of a lesser number of control points. From the figures, it is observed that the maximum number of control points to represent the curve over the complete duration of simulation is 17 for TRX_SD_MRG algorithm.

Intuitively, the higher the number of control points chosen initially, the lesser will be the error in length between control polygon and the spline. This is demonstrated in Table 3.1 in which results of TRX_SD_MRG simulation run on the planar curve input as 20 line segments is shown. As seen in the results, as the initial number of control points increases, the error in total length over the whole simulation comes down significantly to 4.90%. However, as seen in the results, the number of line segments in the backbone also increases accordingly. Finally,

| No. | Initial number of points in Control Polygon | Max. % Error in Curve Length | Max. number of sides in Control Polygon | Total Simulation Time(sec.) |
|---|---|---|---|---|
| 1 | 7 | 12.65 | 14 | 25.53 |
| 2 | 10 | 8.81 | 14 | 25.80 |
| 3 | 13 | 6.74 | 18 | 36.34 |
| 4 | 16 | 4.90 | 19 | 37.37 |

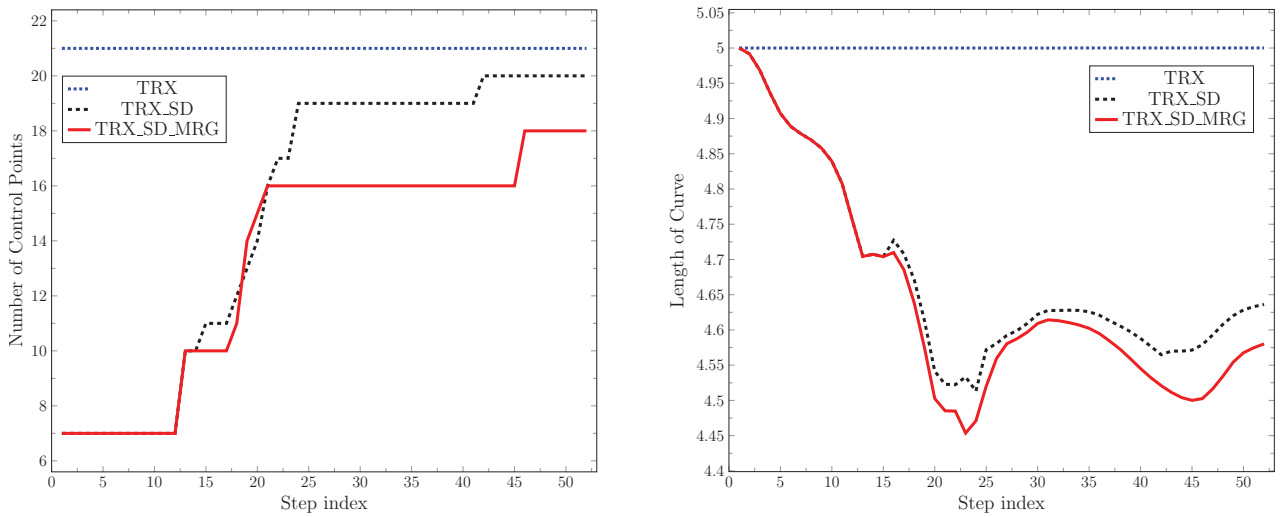
Table 3.1: Algorithm performance comparison for different initial number of control points in the control polygon

to see the effect of the threshold angle, we performed the TRX_SD_MRG simulations on the generic planar curve discretized into 20 segments with different threshold angles for subdivision and knot removal. The results are given in Table 3.2. Clearly, as the threshold angle increases,

the curve sub-divides more frequently thereby resulting in more number of sides in the control polygon. However, the length error keeps on reducing with higher thresholds because the curve subdivides more to keep the length difference within bounds. The simulation results illustrate the theoretical results developed earlier in Section 3.3.

| No. | Subdivision threshold(deg.) | Knot removal threshold(deg.) | Max. percentage diff. in length b/w curve and control polygon | Max. number of sides in control polygon | Total simulation time(sec.) |
|---|---|---|---|---|---|
| 1 | 160 | 170 | 8.60 | 19 | 37.51 |
| 2 | 140 | 160 | 12.65 | 16 | 30.73 |
| 3 | 120 | 150 | 20.35 | 12 | 18.91 |

Table 3.2: Algorithm performance comparison for different threshold angles for subdivision and knot removal




(a) Variation in number of control points

(b) Variation in length of curve

Figure 3.15: Simulation a planar curve discretized by 50 linear segments subjected to generic 2D motion

It may be mentioned that the actual numbers for length error and control points will vary based on the motion. However, the algorithm adaptively computes the lowest number of control points to effectively represent the curve respecting the prescribed error bounds. To illustrate this, a slightly modified trajectory of the leading end is chosen as shown in Figure 3.16. In this trajectory at the end there exists a longer straight segment. The results for the simulation for the new motion are shown below. The number of segments initially chosen was 20 and the merging thresholds are relaxed to 150°. Other parameters were kept exactly the same as before.

As expected, due to the presence of straight trajectory towards the end, the number of control points further reduce to 12 when compared to 18 in the earlier case. However, due to relaxed merging threshold, the maximum error introduced by merging is now seen to be 4%. The

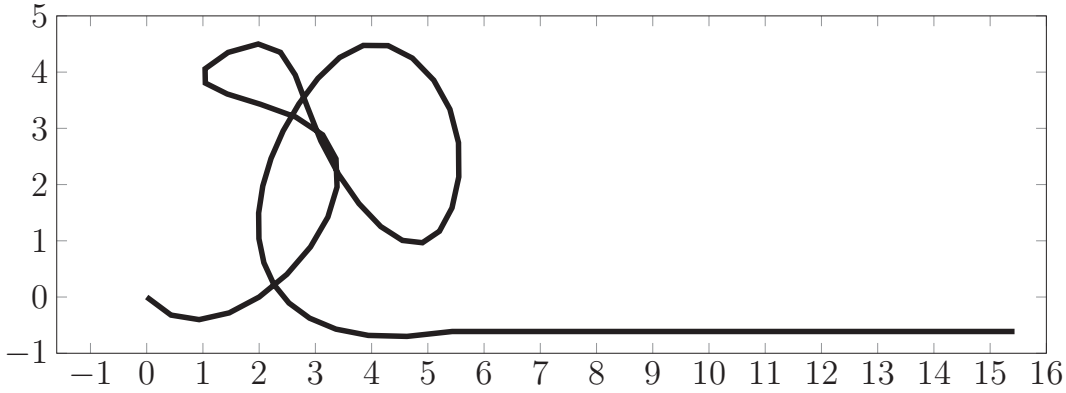
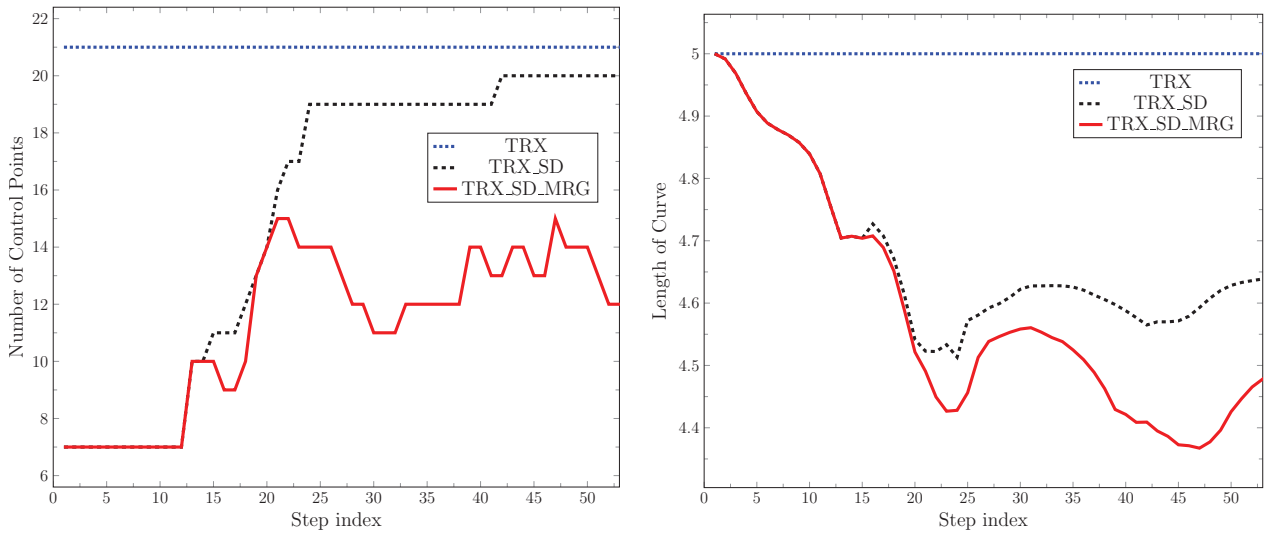



Figure 3.16: Modified motion trajectory for second simulation




(a) Variation in number of control points

(b) Variation in length of curve

Figure 3.17: Simulation of motion of a planar 1D curve with generic 2D motion at the leading end

numerical results shown in above figures clearly shows that the spline based algorithm results in a more natural motion(locally dying perturbations and minimal overall motion) together with a significant reduction in computation time. Additionally, the spline based algorithm is tunable for tolerances on length error specified by user requirements via the threshold values of the angles and the initial number of control points.

The algorithm is easily extendable to three dimensional space without any significant modifications. Figure 3.18 shows the input desired motion of the leading end and simulated snapshots of motion of the complete flexible 1D object using the adaptive spline based subdivision algorithm proposed in this chapter for a generic motion in three dimensions. Below, we describe

| No. | Algorithm used | Initial number of points in control polygon | Maximum percentage difference between curve length and control polygon length | Total simulation time(sec.) |
|---|---|---|---|---|
| 1 | TRX | 15 | 0.0 | 280 |
| 2 | TRX | 25 | 0.0 | 914 |
| 3 | TRX_SD_MRG | 12 | 8.9 | 244 |
| 4 | TRX_SD_MRG | 15 | 4.7 | 300 |

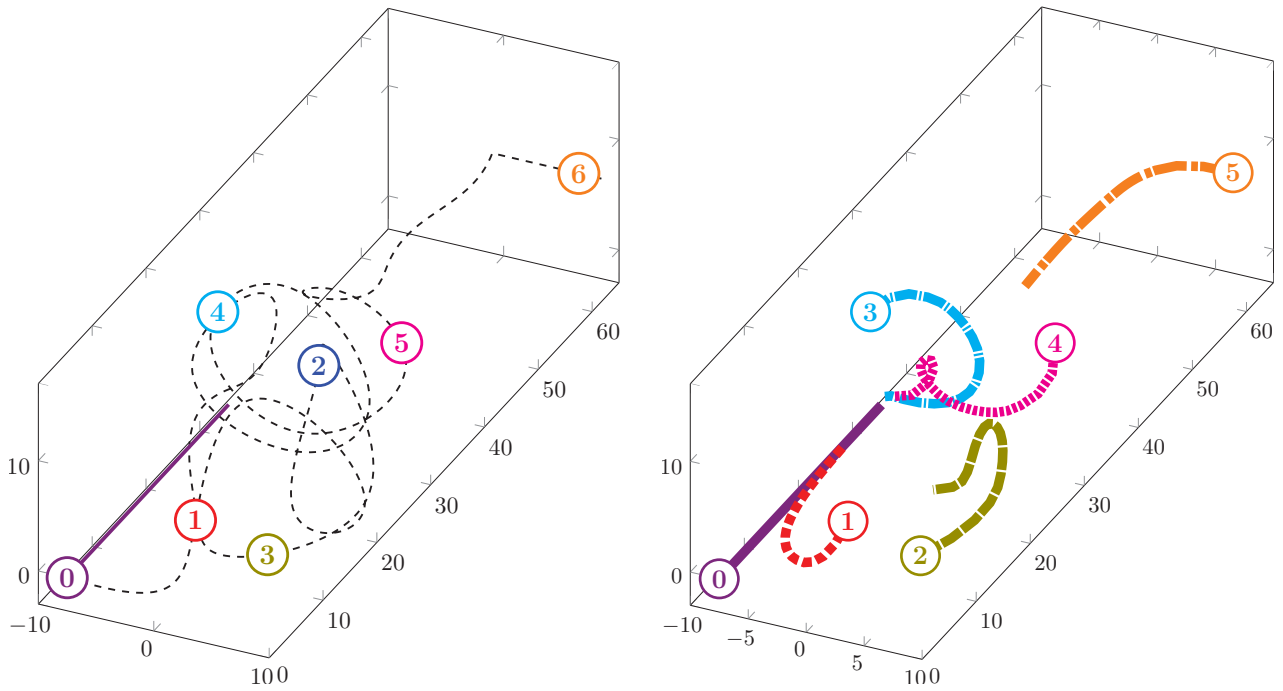
Table 3.3: Algorithm performance comparison for in 3D

results for a set of three simulations carried out for a generic motion in three dimensions. In the first simulation, we used the exact length preserving tractrix based algorithm (TRX) for motion planning and used 15 linear segments to approximate the 1D flexible object. In the second simulation, the number of linear segments is increased to 25 for enhanced smoothness and the third simulation uses the approximate spline based algorithm, the path and snapshots of which are shown in Figure 3.18. The time for simulating with 15 segments is about 280 seconds while it increases to 914 seconds when 25 segments are used to make the motion appear more smoother. In comparison, the third simulation shows an equally smooth motion with the approximate length preserving algorithm (difference less than 10%) in about 244 seconds. It may be noted that the approximate length preserving spline based algorithm takes even smaller time than the exact length preserving algorithm with 15 segments as only a 12 sided control polygon is employed initially. As mentioned in the text, the length error can be brought down below 5% if more number of legs are used to represent the curve in the initial control polygon is used – if a 15 sided control polygon is used and time required is of the order of 300 seconds. Table 3.3 summarizes the simulation results for the 3D trajectory of Figure 3.18.

Results for a second set of simulations shows similar improvement in execution time and smoothness in motion when the leading end is moved along a completely different arbitrary trajectory (Figure 3.19(a)). Motion snapshots of this simulation are shown in Figure 3.19(b).

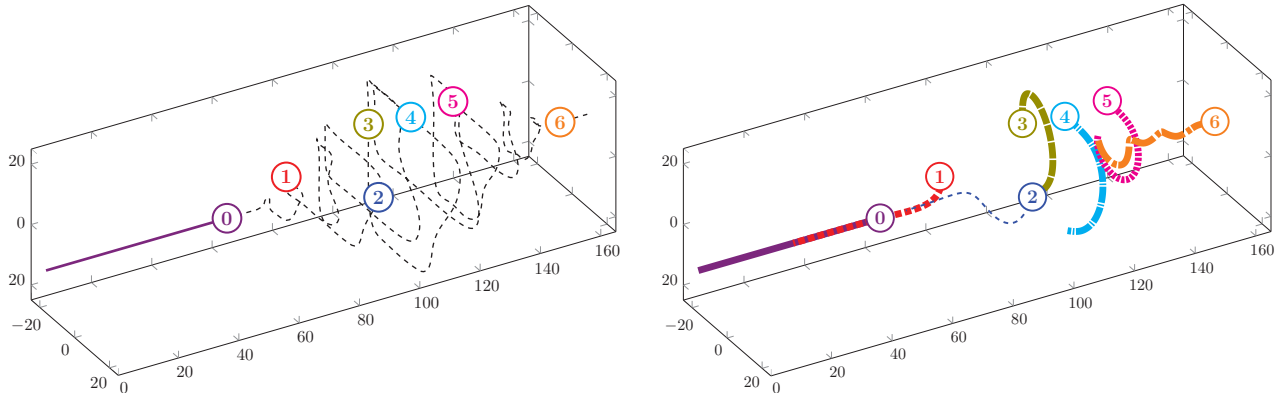
# Summary

In this chapter, we have presented a new paradigm for the simulation and rendering of the motion of 1D flexible objects. The motion of the flexible 1D objects, represented using splines,




(a) A generic 3D motion for the leading end (b) Motion snapshots along the path at various points

Figure 3.18: Motion simulation of an arbitrary curve(A) subjected to a generic 3D motion at the leading end




(a) A generic 3D motion for the leading end (b) Motion snapshots along the path at various points

Figure 3.19: Motion simulation of an arbitrary curve(B) subjected to a generic 3D motion at the leading end

is computed using a tractrix based approach. The tractrix based approach yields a more natural and realistic motion with the motion dying out along the length of the flexible 1D object. The use of splines and adaptive modification of the control polygon leads to an approximate length

preservation with efficient computation and smoother rendering of the motion of the 1D flexible object. An important feature of the proposed algorithm is that it is a purely *kinematics* or *geometry* based algorithm. It is shown that the use of splines and adaptive modification of the control polygon increases computation efficiency and the increased efficiency is more clearly observed when the number of segments in the input is large (more than 20 in the simulations shown in this work). The approach can be easily applied to simulation and realistic rendering of the motion of generic 1D flexible objects such as snakes, chains, ropes and for redundancy resolution in hyper-redundant robotic manipulators.

# Chapter 4

# Motion Planning with Obstacle Avoidance

## Introduction

In Chapter 2, we had presented a tractrix-based approach for motion planning for flexible one-dimensional objects discretized into straight segments. In this chapter we first present an extension of the tractrix-based approach to include obstacle avoidance using the concept of a constrained Lagrangian. The statement of the problem being tackled here is as follows: Given an $n$-link hyper-redundant robot ($n \gg 6$ in 3D & $n \gg 3$ in 2D) and a desired path for the head to follow through a field of smooth, implicitly defined obstacles, plan a natural motion of the entire robot (locally dying perturbations and minimal overall motion), simultaneously avoiding obstacles by gracing them(hence with least possible movement of the robot). In this chapter, we model the obstacles as smooth differentiable surfaces in 3D space and curves in the plane as this enables us to use the calculus of variations approach developed for the motion planning of flexible one-dimensional objects. It is shown that the motion of the links near the obstacles are normal to the obstacles and away from the obstacles the motion is along the link as in the tractrix-based approach. The developed theory is illustrated with numerical simulations. In the second part of the chapter, the details of a 12 link planar hyper-redundant robot prototype are presented. The tractrix-based approach, in free space and in the presence of obstacles, is implemented on the hyper-redundant robot. It is shown that the tractrix-based approach indeed results in a more natural motion of the hyper-redundant robot and avoids the obstacles during its motion.

The chapter organization is as follows. Section 4.2 describes the constrained Lagrangian formulation of the obstacle avoidance problem for an extended one-dimensional body. The

constrained optimization requires that the obstacles are represented by smooth differentiable surfaces and in Section 4.2.1, we describe in brief representation of objects using super-quadrics. Section 4.3 presents simulation results for snake robot moving in a two- and three-dimensional space with obstacles represented by super-quadrics. In this section, the theoretical results related to obstacle avoidance of the flexible one-dimensional object are also presented. Section 4.4 describes simulations and experiments conducted on a 12-degree-of-freedom (DOF) prototype serial robot, and discusses the results of the experiments. Section 4.5 summarizes the contents of this chapter.

# Constrained Lagrangian approach

As shown in Chapter 2, the tractrix motion is the solution to the Cartesian velocity minimization under length preservation constraint. For a flexible one dimensional object of length $L$, the optimization problem constructed is as follows.

$$\begin{aligned}
&\underset{x(s,t),y(s,t)}{Min\ I} \ : \int_0^L\int_0^T \sqrt{\Big(\frac{dT_x}{dt}+\frac{\partial x}{\partial t}\Big)^2+\Big(\frac{dT_y}{dt}+\frac{\partial y}{\partial t}\Big)^2}dtds \\
&Subject\ to \\
&\Lambda(t):\ A=\int_0^L\Big(\sqrt{\Big(\frac{\partial x}{\partial s}\Big)^2+\Big(\frac{\partial y}{\partial s}\Big)^2}-1\Big)ds=0 \\
&Data:\ x(s,0),y(s,0),T_x(t),T_y(t),x(0,t)=0,y(0,t)=0
\end{aligned} \tag{4.1}$$

The above global optimization problem can be broken down into smaller optimization problems on discretized rigid link kinematic chain approximation of the flexible object. If the discretized form of the flexible object made of $(n-1)$ links is $P=[p_1\ p_2\ \dots\ p_n]^T=[X,Y]$ where $X=[x_1(t)\ x_2(t)\ \dots x_n(t)]^T$ and $Y=[y_1(t)\ y_2(t)\ \dots y_n(t)]^T$, then the reduced set of optimization problems are as follows.

$$\begin{aligned}
&\underset{p_{i+1}(t_2))}{Min\ I} \ : \Big(\Delta(T_x+x_{i+1})\Big)^2+\Big(\Delta(T_y+\Delta y_{i+1})\Big)^2 \\
&Subject\ to \\
&\lambda_i:\ \sqrt{(x_{i+1}(t_2)-x_i(t_2))^2+(y_{i+1}(t_2)-y_i(t_2))^2}=L \\
&Data:\ p_i(t_m),p_{i+1}(t_1),T_x(t_m),T_y(t_m),p_1(t_1)=(0,0)^T \\
&\forall i\in[1,n-1]\ \&\ \forall\ t_1\in[0\dots T],\ t_2=t_1+\Delta t
\end{aligned} \tag{4.2}$$

It may be noted that here $m=1,2$.

Given any two dimensional circular obstacle $\mathcal{C} \in \mathbb{R}^2$ (or it's topological equivalent curve), by Jordan-Brouwer Theorem[81], $\mathbb{R}^2 - \mathcal{C}$ has two components, an *interior*($\mathcal{I}$) and an *exterior*($\mathcal{E}$) with $\mathcal{C}$ bounding both(obstacle boundary). For example, given a circular obstacle with center $P_c : (x_c, y_c)$ and radius $R$, the classification of a point $P_i : (x_i, y_i)$ into the three partition sets can be done as follows.

$$(x_i - x_c)^2 + (y_i - y_c)^2 - R^2 \begin{cases} > 0 & \Rightarrow P_i \in \mathcal{E} \\ < 0 & \Rightarrow P_i \in \mathcal{I} \\ = 0 & \Rightarrow P_i \in \mathcal{C} \end{cases} \tag{4.3}$$

The Jordan curve theorem has also been extended to $\mathbb{R}^3$ wherein any topologically equivalent spherical obstacle boundary ($\mathcal{S}$) partitions the points in space into points in the interior set ($\mathcal{I}$), points in the exterior ($\mathcal{E}$) and points on the surface of spherical obstacle ($\mathcal{S}$). The partitioning is based on the value of the implicit representation of the obstacle boundary $f(\boldsymbol{P}) = 0,\ \boldsymbol{P} \in \mathbb{R}^2$ or $\mathbb{R}^3$. For example, for the circular obstacle in Equation (4.3), $f(\boldsymbol{P}) = (x_i - x_c)^2 + (y_i - y_c)^2 - R^2\ \forall \boldsymbol{P} \in \mathbb{R}^2$), the general classification for spatial point $\boldsymbol{P} \in \mathbb{R}^3$ or $\mathbb{R}^2$ for this obstacle is as follows.

$$\begin{aligned} \mathcal{E} &= \{\boldsymbol{P} | f(\boldsymbol{P}) > 0\} \\ \mathcal{I} &= \{\boldsymbol{P} | f(\boldsymbol{P}) < 0\} \\ \mathcal{C} &= \{\boldsymbol{P} | f(\boldsymbol{P}) = 0\} \end{aligned} \tag{4.4}$$

If the implicit obstacle boundary representation $f(\boldsymbol{P}) = 0$ is differentiable, then such obstacles can be incorporated as constraints in optimization problem and classical optimization algorithms like gradient-based methods can be used. In case of a single obstacle with implicit boundary representation $f(\boldsymbol{P}) = 0$, the modified optimization problem for obstacle avoidance takes the following form.

$$\begin{aligned} &\underset{p_{i+1}(t_2))}{Min\ I} \ : \Big(\Delta(T_x + x_{i+1})\Big)^2 + \Big(\Delta(T_y + \Delta y_{i+1})\Big)^2 \\ &\textit{Subject to} \\ &\Lambda_i : \ \sqrt{(x_{i+1}(t_2) - x_i(t_2))^2 + (y_{i+1}(t_2) - y_i(t_2))^2} = L \\ &\beta : \ f(\boldsymbol{P}) > 0 \\ &Data : \ p_i(t_m), p_{i+1}(t_1), T_x(t_m), T_y(t_m), p_1(t_1) = (0,0)^T \\ &\forall i \in [1, n-1]\ \&\ \forall\ t_1 \in [0 \dots T],\ t_2 = t_1 + \Delta t \end{aligned} \tag{4.5}$$

More generally, if there are multiple interfering obstacles $\mathcal{O}_j,\ 1 \leq j \leq p$, each with an exterior $\mathcal{E}_j$, then the intersection of all the individual exteriors gives the permissible space for motion planning, namely $\mathcal{E} = \bigcap_{j=1}^{p} \mathcal{E}_j$. Hence, the most general form of the optimization problem for obstacle avoidance in presence of multiple obstacles $\mathcal{O}_j$ each having differentiable implicit representation $f_j(\boldsymbol{P}) = 0$ is as follows.

$$
\begin{aligned}
&\underset{p_{i+1}(t_2))}{Min}\ I\ : \Big(\Delta(T_x + x_{i+1})\Big)^2 + \Big(\Delta(T_y + \Delta y_{i+1})\Big)^2 \\
&Subject\ to \\
&\Lambda_i :\ \sqrt{(x_{i+1}(t_2) - x_i(t_2))^2 + (y_{i+1}(t_2) - y_i(t_2))^2} = L \\
&\beta_j :\ f_j(\boldsymbol{P}) > 0\ \forall\ j \in [1, p] \\
&Data :\ p_i(t_m), p_{i+1}(t_1), T_x(t_m), T_y(t_m), p_1(t_1) = (0,0)^T \\
&\forall i \in [1, n-1]\ \&\ \forall\ t_1 \in [0 \ldots T],\ t_2 = t_1 + \Delta t
\end{aligned}
\tag{4.6}
$$

It may be noted that in this thesis, existence and knowledge of an obstacle free path for the head/leading end of the curve is assumed and taken as the prerequisite for the following algorithm. Also in this work, we restrict the obstacle shapes to a span a class of objects known as differentiable super-quadrics as they represent superset of differentiable implicit functions making them amenable for use in minimization using calculus of variations and other gradient based methods.

## Differentiable super-ellipses and super-ellipsoids

Super-quadrics were first invented by Piet Hein [82, Chapter 18] and studied by Barr [83]. They are defined as follows.

$$
\left( \left| \frac{x}{a_1} \right|^{\frac{2}{\epsilon_1}} + \left| \frac{y}{a_2} \right|^{\frac{2}{\epsilon_1}} \right)^{\frac{\epsilon_1}{\epsilon_2}} + \left| \frac{z}{a_3} \right|^{\frac{2}{\epsilon_2}} = 1,\ a_i, \epsilon_j \in \mathbb{R}\ \&\ a_i \neq 0 \tag{4.7}
$$

To limit to cases of curves/surfaces with well defined gradients, this work restricts to super-ellipsoids of the following form.

$$
\begin{aligned}
&\left( \left( \left( \frac{x}{a_1} \right)^2 \right)^{\frac{1}{\epsilon_1}} + \left( \left( \frac{y}{a_2} \right)^2 \right)^{\frac{1}{\epsilon_1}} \right)^{\frac{\epsilon_1}{\epsilon_2}} + \left( \left( \frac{z}{a_3} \right)^2 \right)^{\frac{1}{\epsilon_2}} = 1 \\
&0 \leq \epsilon_1 \leq 1,\ 0 \leq \epsilon_2 \leq 1
\end{aligned}
\tag{4.8}
$$

The family of manifolds generated by variation of parameters $\epsilon_1$ and $\epsilon_2$ in Equation (4.8) includes circles, ellipses, rectangles, cylinders, cuboids, cubes, ellipsoids, spheres etc. Some of shapes generated with their parameters are shown in Figures 4.2 and 4.1. Note that $\epsilon_1 = p_2/q_2,\ \epsilon_2 = p_1/q_1$. It may also be noted that some shapes though they have the same exponent parameters, differ in scaling along the axes, namely, the variables $a_i$. Examples are spheres and ellipsoids, cubes and cuboids and this family of curves and surfaces have Equation (4.8) as their exterior-interior partition function.

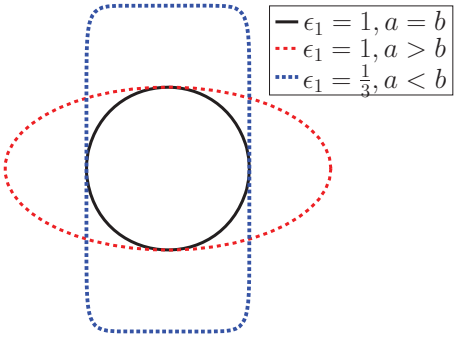



Figure 4.1: Super-ellipses

## Obstacle avoidance

As shown in Chapter 2, the optimization of the motion of the articulated chain (in free space) is equivalent to the iteration of the tractrix curve equations to a single link at a time. Hence, we consider the case of a single link and without loss of generality, consider a single obstacle with boundary represented by implicit function $f(x, y) = 0$. These assumptions simplify the 2D variational problem (4.1) to

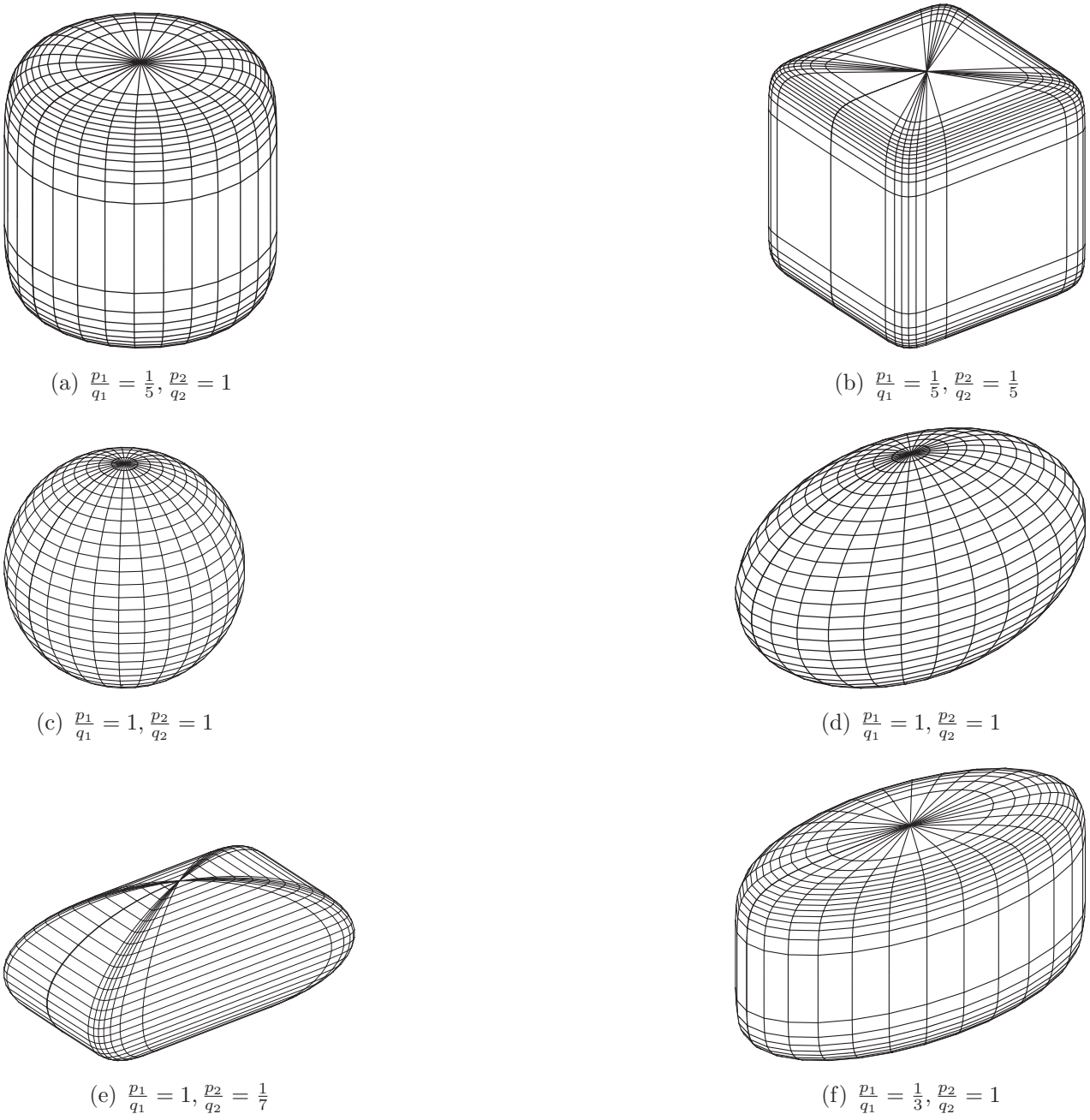
$$\begin{aligned}
&\underset{x(t),y(t)}{Min}\ I\ :\ \int_0^T \sqrt{\left(\dot{T}_x + \dot{x}\right)^2 + \left(\dot{T}_y + \dot{y}\right)^2}\, dt \\
&Subject\ to \\
&\Lambda(t):\ A = x^2 + y^2 - L^2 = 0 \\
&\Omega(t) \geq 0:\ B = f(x, y) \geq 0 \\
&Data:\ x(0), y(0), T_x(t), T_y(t)
\end{aligned} \tag{4.9}$$

Applying the Euler-Lagrange equations on the augmented Lagrangian $\mathcal{L} = I + \Lambda(t)A - \Omega(t)B$ and simplifying, this takes the following form.

$$\dot{T}_x + \dot{x} = R_c \left( \Omega(t) \frac{\partial f}{\partial y} - 2\Lambda(t)x \right) \tag{4.10a}$$

$$\dot{T}_y + \dot{y} = R_c \left( \Omega(t) \frac{\partial f}{\partial x} - 2\Lambda(t) y \right) \tag{4.10b}$$

where $R_c$ is the radius of curvature of the trailing end path and $\Omega(t)$ and $\Lambda(t)$ are the Lagrange multipliers. Clearly, we see that the velocity spans a half space through a linear combination of obstacle outward normals and slope of the link. Figure 4.3 illustrates this.



(a) $\frac{p_1}{q_1} = \frac{1}{5}, \frac{p_2}{q_2} = 1$

(b) $\frac{p_1}{q_1} = \frac{1}{5}, \frac{p_2}{q_2} = \frac{1}{5}$

(c) $\frac{p_1}{q_1} = 1, \frac{p_2}{q_2} = 1$

(d) $\frac{p_1}{q_1} = 1, \frac{p_2}{q_2} = 1$

(e) $\frac{p_1}{q_1} = 1, \frac{p_2}{q_2} = \frac{1}{7}$

(f) $\frac{p_1}{q_1} = \frac{1}{3}, \frac{p_2}{q_2} = 1$

Figure 4.2: Super-ellipsoids

In the absence of obstacles, the motion is a pure tractrix motion with the velocity along the link. In the presence of obstacles, the velocity of the tip lies in the above mentioned half space and ensures obstacle avoidance. The exact direction of the velocity depends on the relative magnitudes of the Lagrange multipliers but the magnitude of the velocity is higher in areas where the path tangent changes rapidly (small $R_c$) and vice-versa.

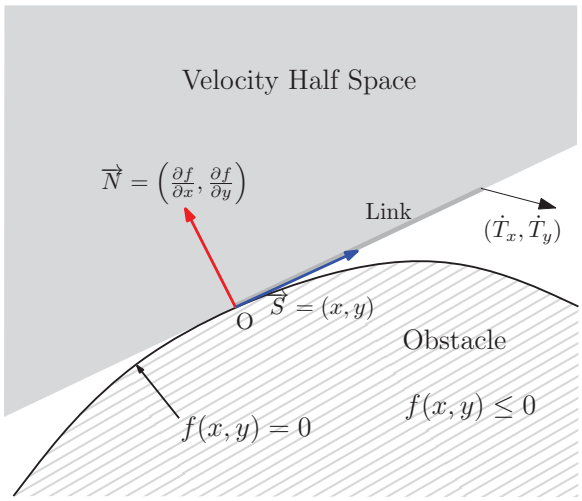



Figure 4.3: Span of calculated link velocity solutions. Note that the calculated velocity is guaranteed to move the body away from the obstacle.

# Simulation results and discussion

In this section, the numerical simulation[1] results are presented for a chosen one-dimensional (1D) flexible object whose leading end is moved along a generic path in two- and three-dimensional space. In all the simulations, the initial configuration of the object is chosen to be a straight line although there is no restriction on initial configuration.

In the first simulation done in two-dimensional (2D) space, a 1D object of length 5 units is discretized into 30 rigid segments connected by rotary joints yielding a hyper-redundant system with 30 degrees of freedom. The leading end is moved along an arbitrarily chosen path in steps of 0.05 length units for 540 steps. The arbitrary path is shown in Figure 4.4. The path is chosen so as to avoid the obstacles and should be continuous(need not be differentiable or tangential to the obstacle boundaries). Along the path, eight arbitrary snapshot locations are chosen denoted by ① to ⑧. The initial configuration (at snapshot ①) of the 1D flexible object is shown in blue.

[1]All simulations were done using the commercial software MATLAB [80] on a Pentium quad core PC with 16Gb RAM running Linux operating system.

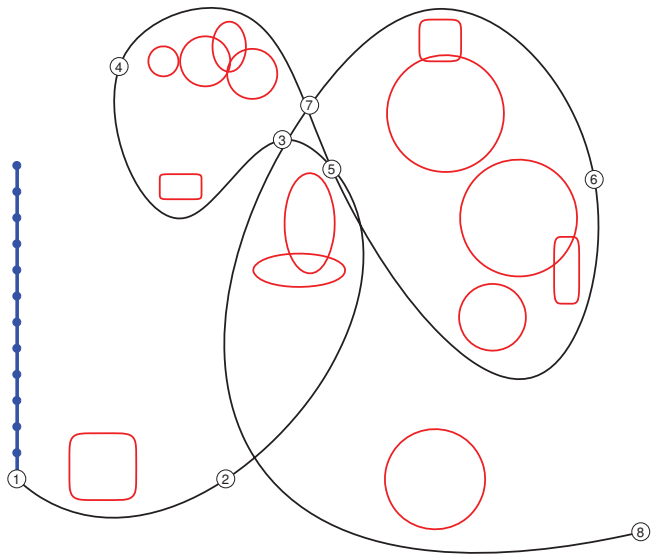
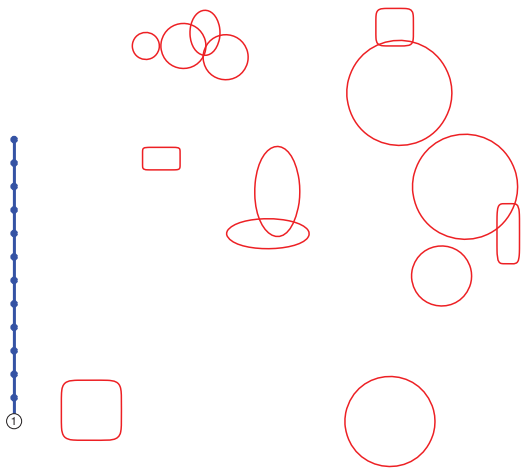


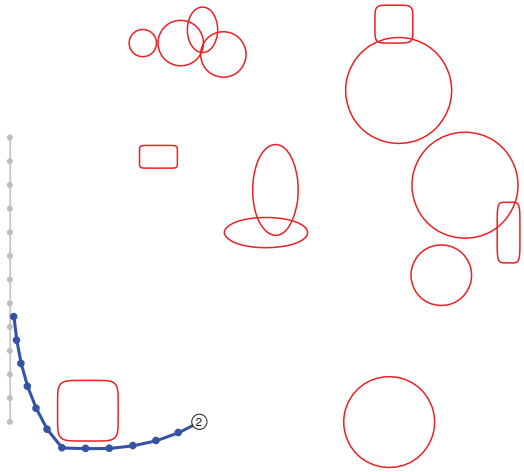

Figure 4.4: Path for 2D simulation with snapshot locations and initial configuration of the flexible 1D object

The configuration of the 1D flexible object at each of the 8 snapshot locations of the leading end are shown in Figure 4.5. As seen here, the obstacles are avoided and motion is minimized as one moves away from the leading end of the flexible object. In other words, tractrix motion is followed in obstacle-free spaces and algorithm automatically switches to obstacle avoidance once the objects are encountered. This is totally determined by active and passive constraints in the optimization problem posed here. Whenever the obstacle lies outside the

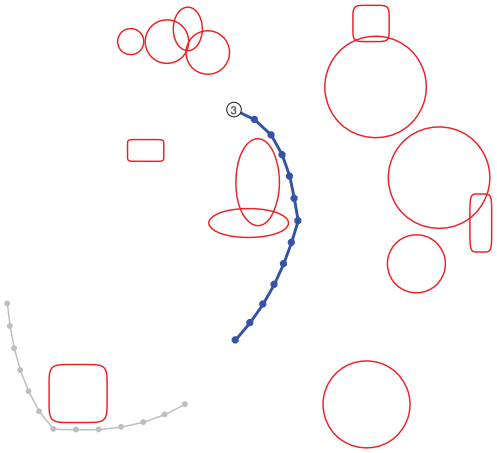
Figure 4.5: Motion snapshots ① to ⑧ for 2D simulation



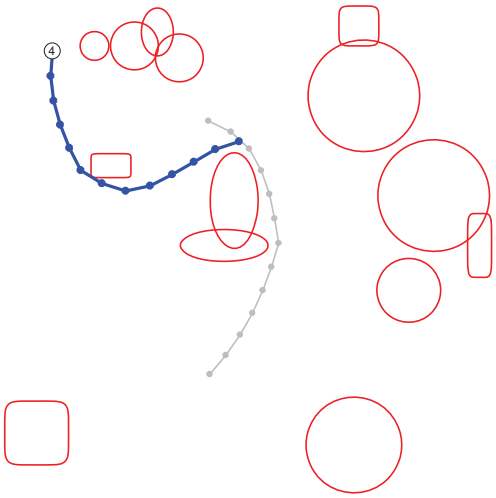

(a) Snapshot at ①



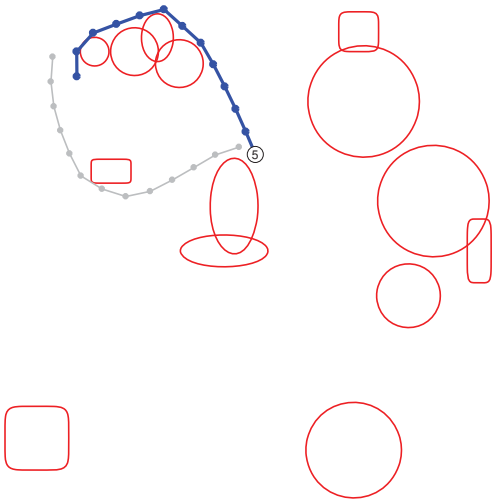

(b) Snapshot at ②



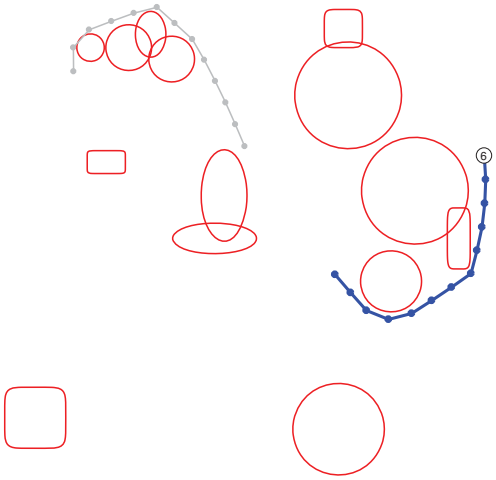

(c) Snapshot at ③



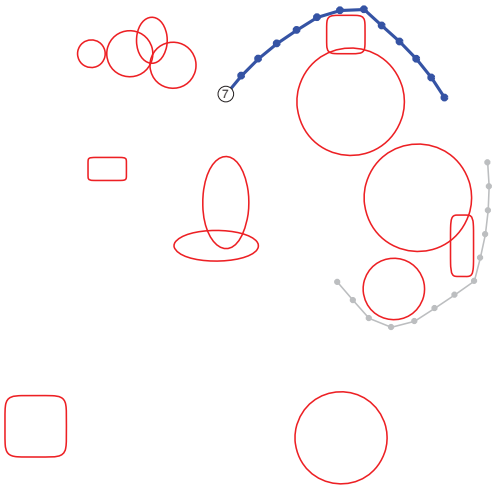

(d) Snapshot at ④



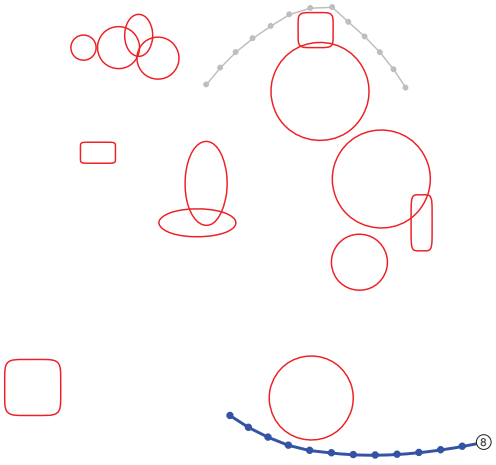

(e) Snapshot at ⑤




(f) Snapshot at ⑥




(g) Snapshot at ⑦




(h) Snapshot at ⑧

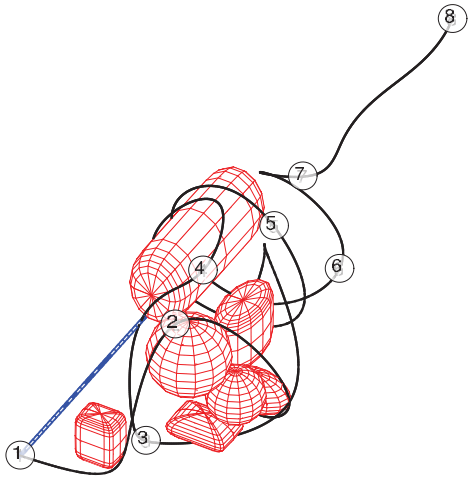
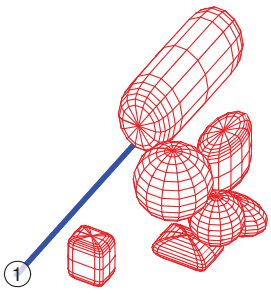


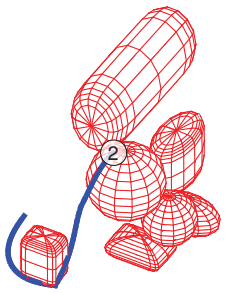

Figure 4.6: Path for 3D simulation with snapshot locations and initial configuration of the flexible 1D object

In the second simulation, the motion planning of the 1D flexible object is done in for an arbitrarily chosen three-dimensional (3D) motion of the leading end. Here too, the initial configuration of the object is chosen as a straight line. A flexible 1D object of length 30 units is discretized into 40 rigid segments with two degree of freedom joints connecting the segments. The leading end is subjected to an arbitrarily chosen motion discretized in to steps of 0.2 length units and the total motion is for 400 steps. The path is in an obstacle field with 7 obstacles of type super-quadrics discussed earlier. It can be seen that the extended body avoids obstacles by gracing them tangentially in worst case. This demonstrates the efficacy of the algorithm.

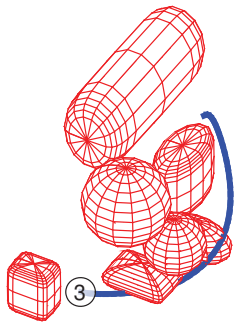
Figure 4.7: Motion snapshots ① to ⑧ for 3D simulation



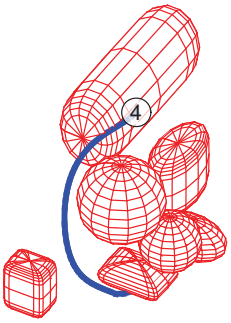

(a) Snapshot at ①



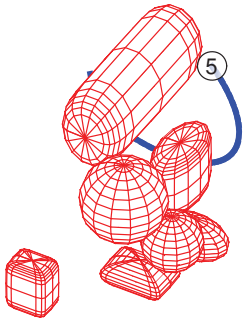

(b) Snapshot at ②



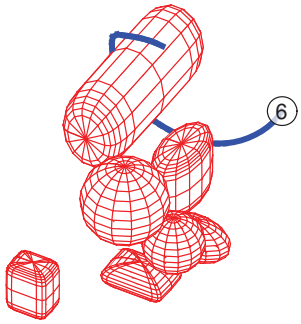

(c) Snapshot at ③



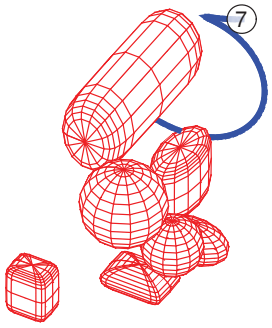

(d) Snapshot at ④



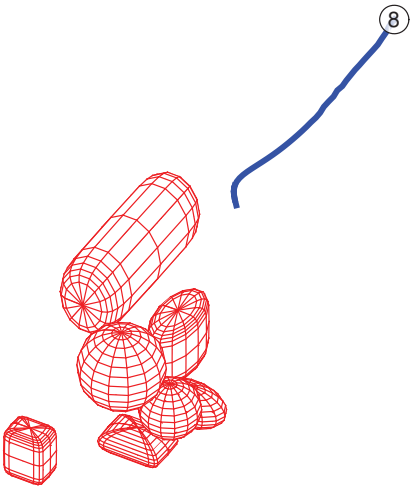

(e) Snapshot at ⑤




(f) Snapshot at ⑥




(g) Snapshot at ⑦




(h) Snapshot at ⑧

## Experimental results and discussion

In this section, the results on test runs carried out on an experimental prototype is discussed. The set-up consists of an 12-link serial robot as shown in Figure 4.8. It has single degree-of-freedom revolute joints connecting each adjacent pair of links. The joints are aligned parallel to each other, allowing for only planar motion. Each link additionally has a powered wheel to enable overall mobility of the mechanism and at each instant the position and orientation of the head in the $XY$-plane is specified. Each link consists of a bracket for two motors, one for the body joint and the other for the wheel motor, as seen in Figure 4.9. The link length, measured from one body joint to the next, is 85 mm. The robot fits into a cylinder of 130 mm diameter. The links were fabricated using a 3D printing machine and the wheels are mounted alternately on either side of the robot. The joints are driven by standard Futaba S3003 RC hobby servos

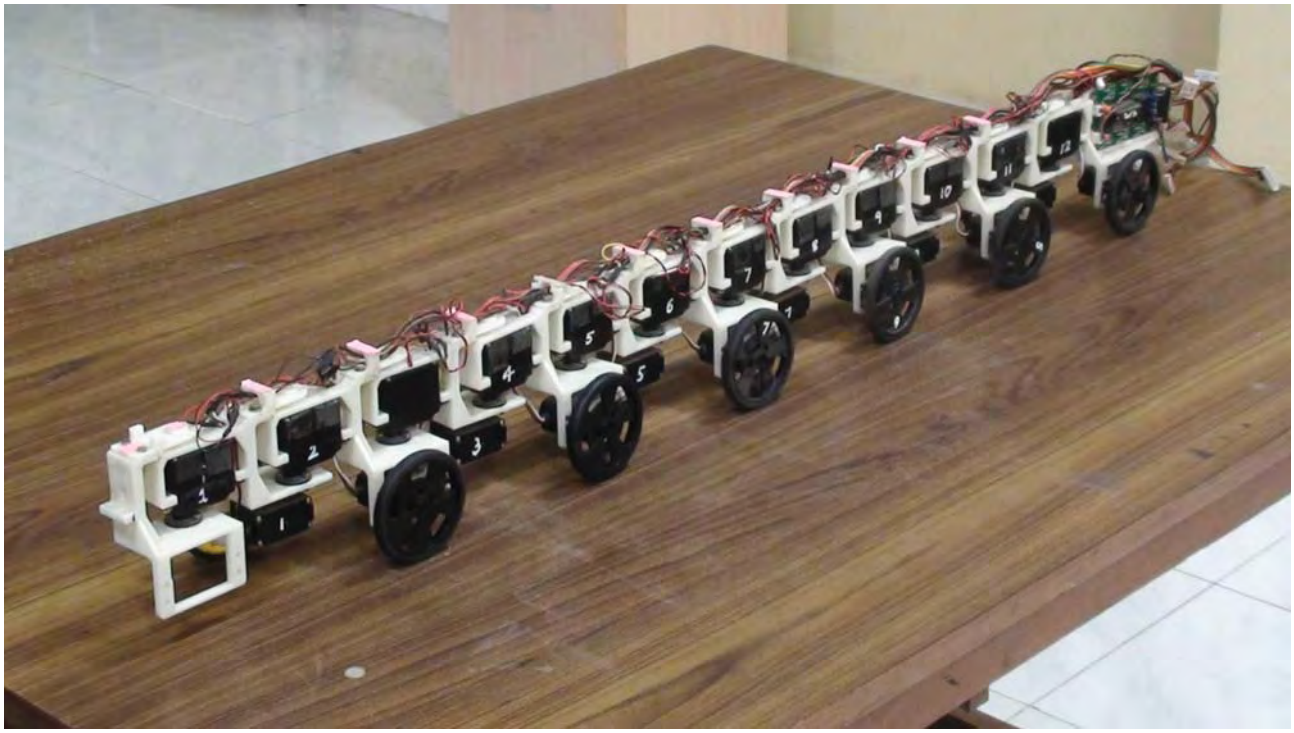


Figure 4.8: Experimental 12-DOF snake robot with wheels for demonstrating obstacle avoidance.

while the wheels are driven by SpringRC SM-S4303R continuous rotation (CR) hobby servos. The servos have a 3-wire interface, with one wire each for positive supply, negative return and command input signal. The servos feature an integrated closed-loop controller to maintain the position/speed according to the command input pulse, thus making a compact actuator ideal for this purpose. On the flip side, the 3-wire interface of the servos does not permit higher level

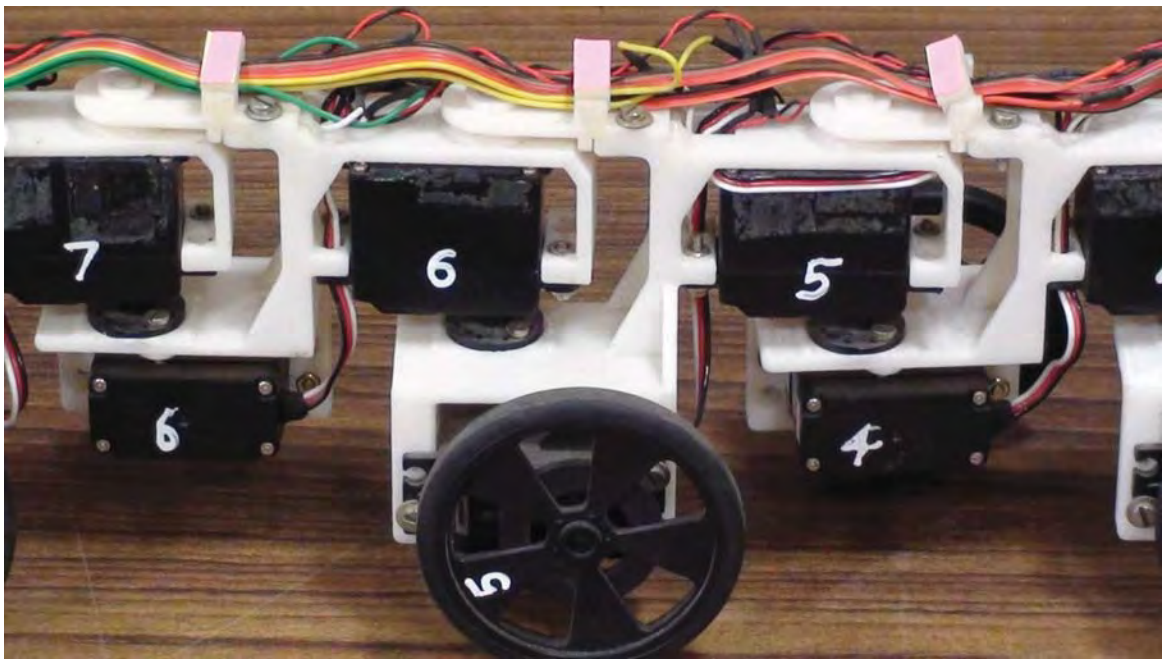



Figure 4.9: Close-up of joint and wheel assembly of the experimental snake robot.

monitoring of the motor position or speed, thus reducing the possibility of effective higher level motion control. The servos take a command position in the form of a pulse-width modulated (PWM) pulse with a nominal time period of $\sim 30$ ms. For the S3003 servos, a pulse width of $\sim$1 ms corresponds to the $-90^\circ$ position and a pulse width of 2 ms corresponds to the $+90^\circ$ position, with the motor centered at 1.5 ms. For the SM-S4303 servos, the motor is at rest at a commanded pulse width of $\sim$1.5 ms. Higher pulse widths cause a clockwise rotation while lower pulse widths cause counter-clockwise rotation, with the speed varying approximately with pulse width. The CR servos were calibrated to determine the relation between pulse width and motor speed and motors with near linear pulse-speed relation were used. These motors were distributed along the length of the body to maintain the balance of the robot when it moves along a straight line. A custom designed PIC18$F$252 micro-controller-based board is used to generate the command pulses for all the 12 joint servo motors. An identical board is used to control the wheel motors. Both the boards also have an RS232 serial interface port to communicate with a PC.

Several experiments were conducted for different choices of obstacle placements and paths. The results of one such experiment are presented below. The experiments were conducted on a planar smooth tiled floor, with known obstacle locations. Obstacles were chosen from among a set of a rectangular object and circular objects of three different sizes. Different numbers, shapes, sizes and relative locations of the obstacles were chosen for each experiment. The initial

orientation of the robot was arbitrarily chosen to be a straight line lying along the $Y$-axis and pointing in the $-Y$ direction. Way-points were manually chosen for the robot to pass between and beyond the obstacles, after inflating the obstacles suitably to account for the robot's width.

In these experiments, the path of the robot's head and the joint configuration of the robot, as the head moves along this path in discrete steps, are calculated using the algorithm as described in 4.2. The sequence of these joint configurations is then fed to the robot through the PC's RS232 interface to the PIC18F252 board controlling the joint servo motors and the wheel kinematics are not considered in these experiments. The calculated wheel speeds are fed to the robot through the PC's RS232 interface to the PIC18F252 board controlling the wheel servo motors. The joint configurations and wheel speeds are synchronized at each step. The algorithm itself is implemented in MATLAB, while the interface to the PIC18F252 boards is implemented in C, both running under Windows XP on an Intel Xeon workstation with 2 GB of RAM.

Figure 4.10 shows a simulated view of the workspace and obstacles, with the calculated path of the robot's head. Figure 4.11 shows a plot of three joint angles angles – joints 1 (head), 6 (middle) and 12 (tail) of the snake robot– over the entire path chosen for the head. It is seen that in the free space, at the start of the path, the motion of the joint angles fall off towards the tail as predicted by the tractrix-based approach. At path points close to the obstacles, all the joint movements are similar and they are such that the entire body of the snake robot avoids the obstacle.

The top half of Figure 4.12 shows a sequence of snapshots of the simulated robot configuration along the path and the bottom half shows the corresponding robot configuration. The path consisted of 12 way-points specified manually for the robot head to pass between the obstacles, with initial and end point orientations specified. The path of the head was calculated by fitting a spline through the 12 way-points and joint configurations were calculated for 132 steps at a spacing of 25 mm along the spline, using the optimization algorithm with obstacle avoidance constraints. The calculation of the spline through all the way-points took 242 ms and the computation of the joint configurations with obstacle avoidance for each step of movement took 244 ms on average, with a worst case time of $\sim$2.23 s for the first step and an average time of 228 ms for subsequent steps. The joint angles and wheel speeds were further interpolated in 30 steps, for smoothness of motion, before being fed to the robot. The commands to the robot take approximately 100 ms to be transferred over a slow (9600 bps) serial link. It may be mentioned that computation times can be improved with code optimization and dedicated hardware.

The main objective of the experiments with carried out with the fabricated hyper-redundant

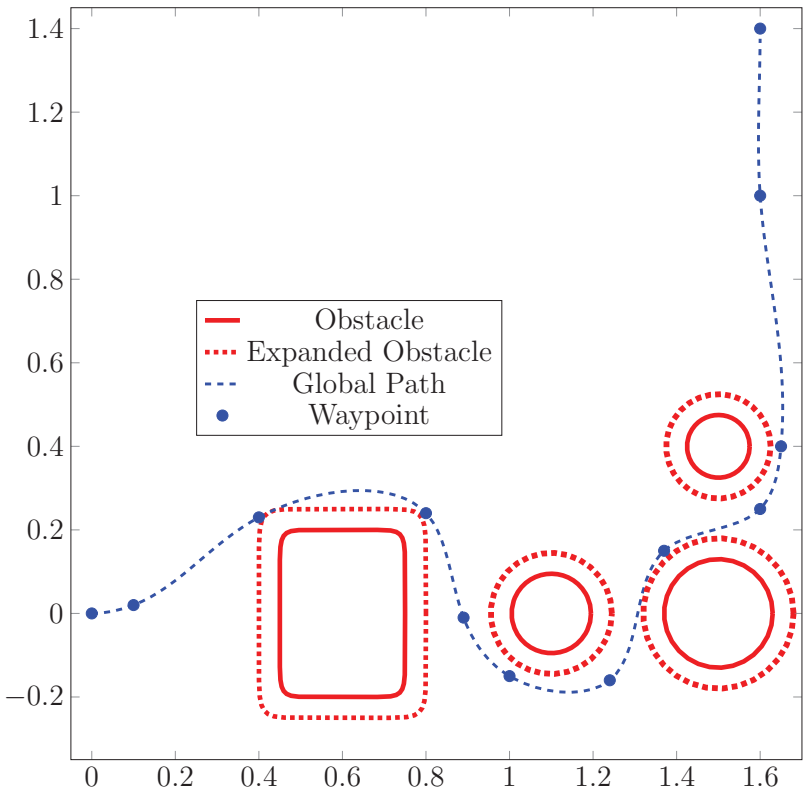



Figure 4.10: Workspace with obstacles and desired path of the robot head.

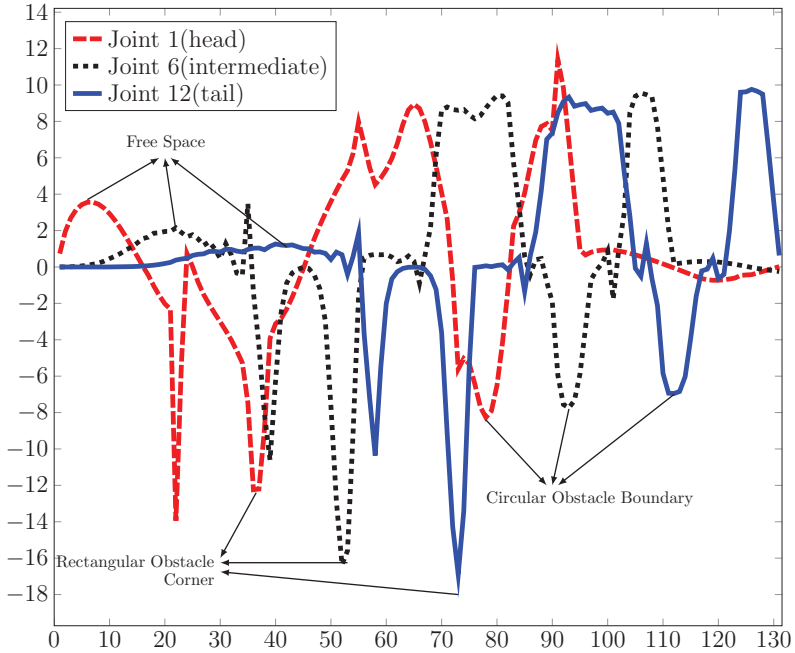
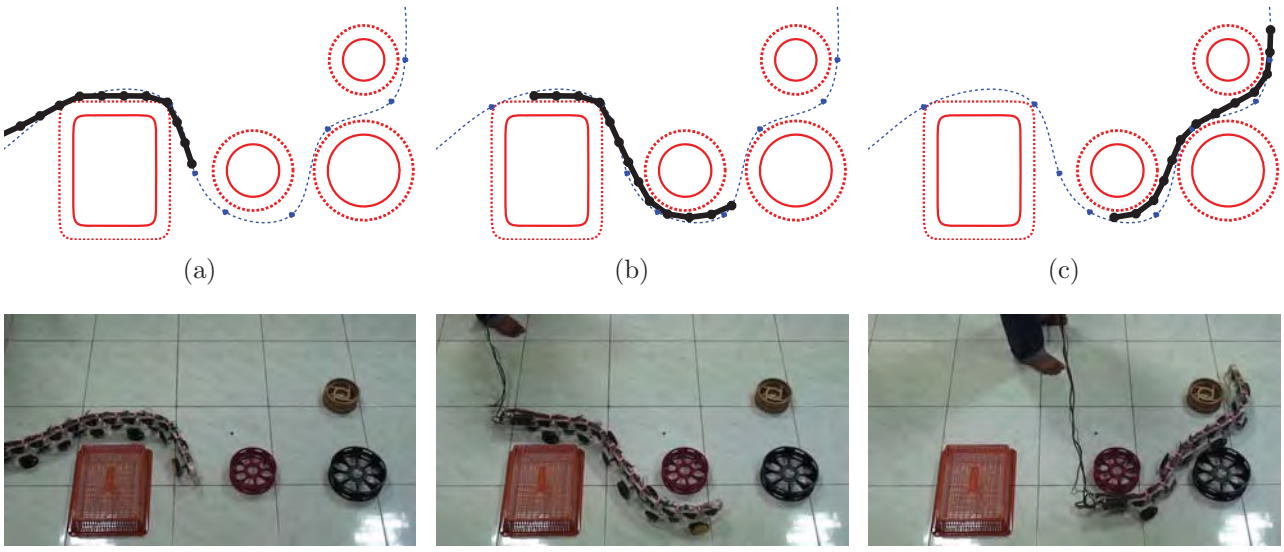



Figure 4.11: Comparison of joint angles at various points in the body.



(a) (b) (c)

(d) Step 45 (e) Step 65 (f) Step 97

Figure 4.12: Simulated (top) and actual (bottom) configuration of the robot at steps 45, 65 and 97 of the motion

robot was to verify the feasibility of implementing the tractrix based algorithm on a physical prototype. It may be noted that the usage of wheels results in non-holonomic constraints on the wheel-ground contact points of the robot and the wheel slip and other dynamic effects are not taken into account in the tractrix based algorithm. We have tried to minimize these effects by moving slowly and on a hard flat floor. One consequence of the wheel slip and other un-modeled dynamics is that the path traced is not exactly the same as the desired path. Nevertheless, the hyper-redundant robot using the tractrix based motion planning algorithm avoids the obstacles and the path followed by the prototype is reasonably close to the desired path as seen in figure 4.12.

## Summary

In this chapter, an efficient optimization-based approach to motion planning with obstacle avoidance for extended bodies, such as snake-like robots has been proposed. The presented approach yields natural looking motion while avoiding obstacles by tangentially gracing them. An important feature of the proposed algorithm is that it is purely kinematics and geometry based algorithm – it does not use any dynamics or artificial potential field type of constructs. The framework is very general as any obstacle shape modeled by union of objects modeled by first order differentiable implicit equation can be directly incorporated. The numerical results demonstrates that the algorithm is able to efficiently avoid obstacles while maintaining the

nature of tractrix motion diminishing property wherever possible. This approach presented in this work has been numerically simulated to obstacle avoidance in 2D and 3D space and works for obstacle avoidance of extended bodies like snakes, ropes, surgical suturing simulation and in motion planning of hyper-redundant manipulators.

Further, experimental results for a 12-link hyper-redundant robot moving in a field of obstacles have been presented. Results from a prototype 12-link snake robot are seen to be very close to the simulation results and validate the optimization based algorithm. The algorithm is also amenable to efficient implementation with on-board sensing of the obstacles in real-time. In the present setup, the lack of wheel speed and joint position feedback results in a certain amount of wheel slippage at certain points in the path and slippage can be minimized by refined design. The obstacles are represented by smooth, differentiable functions due to the requirement of the gradient based optimization algorithm. However, this is not a serious constraint and one can have obstacles represented by piece-wise smooth functions and one can also use other optimization algorithms.

# Chapter 5

# Conclusions

## Summary

The key contributions of this thesis are in the area of motion planning of flexible one-dimensional objects modeled as curves or discretized as piece-wise straight segments. New theoretical results and algorithms are developed for the motion of points on the curve or the ends of the straight segments. The algorithms lead to a more natural motion and are highly efficient thereby making them amenable to real-time implementation. The algorithms developed for discretized segments are extended to include obstacle avoidance and implemented on a planar 12-link hyper-redundant robot. It is demonstrated through numerical simulations and experiments that the motion planning algorithms indeed give rise to natural motion of one-dimensional flexible objects.

Chapter 1 presents the motivation and a detailed literature survey on the motion planning of flexible one-dimensional objects and hyper-redundant robots. The representation of flexible objects as B-spline curves, its control polygon and known results on the length of a curve are presented. Hyper-redundant robots and the key problem of resolution of redundancy using existing approaches in literature is discussed and various techniques for obstacle avoidance in mobile robots are presented. The scope and contributions of the work is also presented in this chapter.

In Chapter 2, the problem of motion planning of flexible one-dimensional objects is posed as a velocity minimization problem subjected to constraints. Using the tools of calculus of variation, several new analytical results are obtained for curves and flexible 1D objects discretized by straight segments. It is shown that the known tractrix based approach for resolution of redundancy in hype-redundant robots can be derived as special case of the general minimization problem. New results which lead to a more natural motion of the flexible 1D object and

hyper-redundant robots are presented. It is shown that other objective functions can also be used and these results in different approaches for motion planning of flexible 1D objects and hyper-redundant manipulators. Several numerical simulations in plane and three-dimensional space are used to illustrate the theoretical results.

Spline theory is introduced in Chapter 3 to further optimize the motion planning algorithm developed in Chapter 2. A B-spline curve can be represented by a control polygon with very few sides and the key idea for minimizing and better rendering of the motion of the flexible 1D object was to move the sides of the control polygon instead of the large number of discretized segments used to represent the curve. Analytical relations between the length of a B-spline curve and the control polygon length were established and useful error bounds on the length of the curve as a function of the angle between adjacent segments of the control polygon were established. When the B-spline curve is moved, the length of the curve changes. In this chapter, using the developed error bounds, an adaptive algorithm, using sub-division and merging, was designed to approximately preserve the length of the curve. Extensive numerical simulation results were used to demonstrate the effectiveness and increased computational efficiency of adaptive algorithms.

In Chapter 4, the problem of motion planning in a field of obstacles was studied. The obstacle avoidance problem is again posed as an optimization problem with obstacles modeled as smooth super quadrics. It was shown that near an obstacle the motion of the point was normal to the obstacle surface and away from the obstacle, as in the case of the tractrix, the motion was long the tangent. Extensive numerical simulations in 2D and 3D were presented which demonstrated the effectiveness of the developed algorithms. Chapter 4 also presented the details of 12-link planar hyper-redundant prototype robot developed to validate the developed motion planning algorithms for free space motion and motion in the presence of obstacles.The motion of the prototype robot was seen to match closely with the numerical simulation thereby demonstrating that the algorithms can be used for practical implementation.

On a note of caution, there exist pathological cases also where this algorithm might fail. One typical case will be when the leading end is being pushed(perturbation vector is along the link towards trailing link) around an obstacle, due to existence of two symmetric solutions. This ia a singularity/degenerate case for this algorithm. However, there are ways around these special cases. A possible resolution is to add additional constraints which might be suitable for the problem in hand, so as to make one solution more preferable(cost effective) over the other. Another resolution would be to slightly perturb the solution toward one feasible solution so that the optimization converges to that solution due to the slightly differential cost gain in that direction.

At the time of submitting this thesis, the author hasn't been able to locate any publicly available data concerning locomotion studies on different real snakes so as to compare with the results from the proposed tractrix-based algorithm. Also, the author was unsuccessful in finding any common benchmark standards for comparison-neither data from different algorithms nor codes. Hence, a quantitative comparison with existing algorithms couldn't be carried out.

## Scope for future work

In Chapter 2, lack of analytical expressions for arbitrary curves and the norms have forced us to employ numerical methods. This brings along with it inherent issues, namely ones related to convergence, stability and sensitivity to initial guess in the numerical optimization. Although a rigorous mathematical proof of convergence, stability and sensitivity to initial guess is not available for the theoretical approach presented in this chapter, the situation is not hopeless. This is based on following specific observation and reasoning – (a) we did not face any of these issues during the numerous numerical simulations we performed on a large number of arbitrarily chosen curves pulled along arbitrary directions in 2D and 3D space, and (b) for a straight line there exists analytical solutions without any of the above mentioned issues and an arbitrary curve can be considered to be the limiting case of a large number of linear segments. However, closed-form analytical expressions (as in the case of a straight line) for some simpler planar and spatial curves are expected to strengthen the theoretical approach.

In Chapter 3, we have been able to obtain analytical results for a quadratic spline and only bounds on length of the cubic and higher-order splines. However, it will be better if analytical expressions can be found in terms of the length of the sides of the control polygon and angles between them for cubic and other splines. This can provide an exact expression for curve length for any arbitrary degree of interpolation function.

In Chapter 4, results from a prototype 12-link hyper-redundant robot are seen to be very close to the numerical simulation results and validate the optimization based algorithm. Implementing this algorithm with on-board sensing of the obstacles in real-time will be the ideal route and will form a part of future work in this space. Additionally in the current setup, the lack of wheel speed and joint position feedback results in a certain amount of wheel slippage at certain points in the trajectory. Extensions planned to this work include an in-depth analysis of the Lagrangian multipliers and also improving the mechanism to reduce slippage in motion.

# Bibliography


[1] LA Piegl and W Tiller. *The NURBS book*. 1997 (cit. on pp. 2, 37, 40, 51, 52).

[2] S Bernštein. "Démonstration du théoreme de Weierstrass fondée sur le calcul des probabilities". In: *Comm. Soc. Math. Kharkov* 13 (1912), pp. 1–2 (cit. on p. 3).

[3] Haskell Brooks Curry and Isaac J Schoenberg. "On Pólya frequency functions IV: the fundamental spline functions and their limits". In: *Journal d'analyse mathématique* 17.1 (1966), pp. 71–107 (cit. on p. 4).

[4] Isaac J Schoenberg. "Contributions to the problem of approximation of equidistant data by analytic functions". In: *IJ Schoenberg Selected Papers*. Springer, 1988, pp. 3–57 (cit. on p. 4).

[5] Lyle Ramshaw. "Blossoms are polar forms". In: *Computer Aided Geometric Design* 6.4 (1989), pp. 323–358 (cit. on p. 4).

[6] Maurice G Cox. "The numerical evaluation of B-splines". In: *IMA Journal of Applied Mathematics* 10.2 (1972), pp. 134–149 (cit. on p. 4).

[7] Carl De Boor. "On calculating with B-splines". In: *Journal of Approximation Theory* 6.1 (1972), pp. 50–62 (cit. on p. 4).

[8] Carl De Boor et al. *A practical guide to splines*. Vol. 27. Springer-Verlag New York, 1978 (cit. on p. 4).

[9] Bernard Dacorogna et al. *Introduction to the Calculus of Variations*. Vol. 13. World Scientific, 2004 (cit. on p. 6).

[10] Kelly Ward et al. "A survey on hair modeling: styling, simulation, and rendering". In: *IEEE Transaction on Visualization and Computer Graphics*. 2006, pp. 213–234 (cit. on p. 6).

[11] U. Kühnapfel, H. K. Çakmak, and H. Maaß. "Endoscopic surgery training using virtual reality and deformable tissue simulation". In: *Computers & Graphics* 24.5 (2000), pp. 671–682. ISSN: 0097-8493 (cit. on pp. 6, 7).

[12] H. Kang and J.T. Wen. "Robotic knot tying in minimally invasive surgeries". In: *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. Vol. 2. IEEE. 2002, pp. 1421–1426 (cit. on pp. 6, 7).

[13] Andrew Nealen et al. "Physically based deformable models in computer graphics". In: *Computer Graphics Forum*. Vol. 25. 4. Wiley Online Library. 2006, pp. 809–836 (cit. on p. 7).

[14] Ronen Barzel. "Faking Dynamics of Ropes and Springs". In: *IEEE Comput. Graph. Appl.* 17 (3 1997), pp. 31–39. ISSN: 0272-1716 (cit. on p. 7).

[15] Elke Hergenröther and Patrick Dähne. "Real-time virtual cables based on kinematic simulation". In: *In Proceedings of the WSCG*. Citeseer. 2000 (cit. on p. 7).

[16] Hasan Dogu Taskıran and G Ugur. "Physically-based Simulation of Hair Strips in Real-Time". In: *WSCG'2005* (), pp. 153–156 (cit. on p. 7).

[17] Uğur Güdükbay, Bülent Özgüç, and Yilmaz Tokad. "A spring force formulation for elastically deformable models". In: *Computers & Graphics* 21.3 (1997), pp. 335–346 (cit. on p. 7).

[18] Suriya Natsupakpong and M Cenk Çavuşoğlu. "Determination of elasticity parameters in lumped element (mass-spring) models of deformable objects". In: *Graphical Models* 72.6 (2010), pp. 61–73 (cit. on p. 7).

[19] Mireille Grégoire and Elmar Schömer. "Interactive simulation of one-dimensional flexible parts". In: *Computer-Aided Design* 39.8 (2007), pp. 694–707 (cit. on p. 7).

[20] Jonas Spillmann and Matthias Teschner. "Cosserat nets". In: *Visualization and Computer Graphics, IEEE Transactions on* 15.2 (2009), pp. 325–338 (cit. on p. 7).

[21] Dinesh K. Pai. "STRANDS: Interactive Simulation of Thin Solids using Cosserat Models". In: *Computer Graphics Forum* 21.3 (2002), pp. 347–352. ISSN: 1467-8659 (cit. on p. 7).

[22] Mark Moll and Lydia E Kavraki. "Path planning for deformable linear objects". In: *Robotics, IEEE Transactions on* 22.4 (2006), pp. 625–636 (cit. on p. 7).

[23] Julien Lenoir et al. "Adaptive resolution of 1D mechanical B-spline". In: *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*. ACM. 2005, pp. 395–403 (cit. on p. 7).

[24] Adrien Theetten et al. "Geometrically exact dynamic splines". In: *Computer-Aided Design* 40.1 (2008), pp. 35–48 (cit. on p. 7).

[25] Rony Goldenthal et al. "Efficient simulation of inextensible cloth". In: *ACM Transactions on Graphics (TOG)* 26.3 (2007), p. 49 (cit. on p. 7).

[26] David Baraff and Andrew Witkin. "Large steps in cloth simulation". In: *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. ACM. 1998, pp. 43–54 (cit. on p. 7).

[27] Huamin Wang, James O'Brien, and Ravi Ramamoorthi. "Multi-resolution isotropic strain limiting". In: *ACM Transactions on Graphics (TOG)*. Vol. 29. 6. ACM. 2010, p. 156 (cit. on p. 7).

[28] A Mikchevitch, J-C Léon, and A Gouskov. "Flexible beam part manipulation for assembly operation simulation in a virtual reality environment". In: *Journal of Computing and Information Science in Engineering* 4.2 (2004), pp. 114–123 (cit. on p. 7).

[29] Joel Brown, Jean-Claude Latombe, and Kevin Montgomery. "Real-time knot-tying simulation". In: *The Visual Computer* 20.2-3 (2004), pp. 165–179 (cit. on p. 8).

[30] Zhixun Su, Ling Li, and Xiaojie Zhou. "Arc-length preserving curve deformation based on subdivision". In: *Journal of Computational and Applied Mathematics* 195.1-2 (2006), pp. 172–181. ISSN: 03770427 (cit. on p. 8).

[31] S. Sreenivasan, Piyush Goel, and Ashitava Ghosal. "A real-time algorithm for simulation of flexible objects and hyper-redundant manipulators". In: *Mechanism and Machine Theory* 45.3 (2010), pp. 454–466. ISSN: 0094-114X (cit. on pp. 8, 19, 25–27).

[32] Hugo Steinhaus. *Mathematical snapshots*. New York: Oxford University, 1969. ISBN: 9780195001174 (cit. on pp. 8, 18).

[33] Midhun S Menon, GK Ananthasuresh, and Ashitava Ghosal. "Natural motion of one-dimensional flexible objects using minimization approaches". In: *Mechanism and Machine Theory* 67 (2013), pp. 64–76 (cit. on pp. 8, 14).

[34] Izrail Moiseevich Gelfand, Sergeĭ Vasilevich Fomin, and Richard A Silverman. *Calculus of variations*. Courier Corporation, 2000 (cit. on p. 9).

[35] Robert A Adams and John JF Fournier. *Sobolev spaces*. Vol. 140. Academic press, 2003 (cit. on p. 10).

[36] J Angeles. "Is there a characteristic length of a rigid-body displacement?" In: *Mechanism and Machine Theory* 41.8 (Aug. 2006), pp. 884–896. ISSN: 0094114X (cit. on pp. 10, 21).

[37] P.R. Halmos. *Finite-Dimensional Vector Spaces*. Undergraduate Texts in Mathematics. Springer New York, 1993. ISBN: 9780387900933 (cit. on p. 10).

[38] Ashitava Ghosal. "Robotics: fundamental concepts and analysis". In: (2006) (cit. on p. 11).

[39] Yoshihiko Nakamura. *Advanced Robotics: Redundancy and Optimization*. 1st. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990. ISBN: 0201151987 (cit. on p. 11).

[40] Gregory S Chirikjian and Joel W Burdick. "A modal approach to hyper-redundant manipulator kinematics". In: *Robotics and Automation, IEEE Transactions on* 10.3 (1994), pp. 343–354 (cit. on pp. 11, 12, 19).

[41] Dan Reznik and Vladimir Lumelsky. "Motion Planning With Uncertainty For Highly Redundant Kinematic Structures I." Free Snake" Motion". In: *Intelligent Robots and Systems, 1992., Proceedings of the 1992 lEEE/RSJ International Conference on*. Vol. 3. IEEE. 1992, pp. 1747–1752 (cit. on pp. 11, 12, 19, 25).

[42] V. C. Ravi, Subrata Rakshit, and Ashitava Ghosal. "Redundancy Resolution Using TractrixSimulations and Experiments". In: *Journal of Mechanisms and Robotics* 2 (3 2010), pp. 1–15 (cit. on pp. 11, 12, 19, 25).

[43] Kourosh E Zanganeh and Jorge Angeles. "The inverse kinematics of hyper-redundant manipulators using splines". In: *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*. Vol. 3. IEEE. 1995, pp. 2797–2802 (cit. on p. 12).

[44] C.A. Klein and C.H. Huang. "Review of pseudoinverse control for use with kinematically redundant manipulators". In: *Ieee Transactions On Systems Man And Cybernetics* 3 (1983), pp. 245–250 (cit. on pp. 12, 19).

[45] A. Liegeois. "Automatic supervisory control of the configuration and behavior of multibody mechanisms". In: *IEEE Transactions on Systems, Man, and Cybernetics* 7.12 (1977), pp. 868–871 (cit. on p. 12).

[46] J. Baillieul, J. Hollerbach, and R. Brockett. "Programming and Control of Kinematically Redundant Manipulators". In: *Proc. 23rd IEEE Conf. on Decision and Control. Las Vegas. Nevada*. New York: IEEE, 1984, pp. 768–774 (cit. on p. 12).

[47] Tsuneo Yoshikawa. "Manipulability of Robotic Mechanisms". In: *The International Journal of Robotics Research* 4.2 (1985), pp. 3–9 (cit. on p. 12).

[48] Dan Reznick and Vladimir Lumelsky. "Sensor-based motion planning for highly redundant kinematic structures. II. The case of a snake arm manipulator". In: *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*. IEEE. 1993, pp. 889–894 (cit. on pp. 12, 19, 25).

[49] Dan Reznik and Vladimir Lumelsky. "Sensor-based motion planning in three dimensions for a highly redundant snake robot". In: *Advanced Robotics* 9.3 (1994), pp. 255–280 (cit. on pp. 12, 14, 19, 25).

[50] Yong K Hwang and Narendra Ahuja. "Gross motion planning-A survey". In: *ACM Computing Surveys (CSUR)* 24.3 (1992), pp. 219–291 (cit. on p. 13).

[51] Rodney A Brooks. "Planning collision-free motions for pick-and-place operations". In: *The International Journal of Robotics Research* 2.4 (1983), pp. 19–44 (cit. on p. 13).

[52] Tomas Lozano-Perez. "Automatic planning of manipulator transfer movements". In: *Systems, Man and Cybernetics, IEEE Transactions on* 11.10 (1981), pp. 681–698 (cit. on p. 13).

[53] Dieter Fox, Wolfram Burgard, Sebastian Thrun, et al. "The dynamic window approach to collision avoidance". In: *IEEE Robotics & Automation Magazine* 4.1 (1997), pp. 23–33 (cit. on p. 13).

[54] O Khatib. "Real-time obstacle avoidance for manipulators and mobile robots". In: *Int. J. Rob. Res.* 5 (1 1986), pp. 90–98. ISSN: 0278-3649 (cit. on p. 13).

[55] Jerome Barraquand, Bruno Langlois, and Jean-Claude Latombe. "Numerical potential field techniques for robot path planning". In: *Systems, Man and Cybernetics, IEEE Transactions on* 22.2 (1992), pp. 224–241 (cit. on p. 13).

[56] Elon Rimon and Daniel E Koditschek. "Exact robot navigation using artificial potential functions". In: *Robotics and Automation, IEEE Transactions on* 8.5 (1992), pp. 501–518 (cit. on p. 13).

[57] Micha Sharir, Chee K Yap, et al. "Retraction: A new approach to motion-planning". In: *Proceedings of the fifteenth annual ACM symposium on Theory of computing*. ACM. 1983, pp. 207–220 (cit. on p. 13).

[58] Roy Glasius, Andrzej Komoda, and Stan CAM Gielen. "Neural network dynamics for path planning and obstacle avoidance". In: *Neural Networks* 8.1 (1995), pp. 125–133 (cit. on p. 13).

[59] John W Boyse. "Interference detection among solids and surfaces". In: *Communications of the ACM* 22.1 (1979), pp. 3–9 (cit. on p. 13).

[60] Tony J Prescott and John EW Mayhew. "Obstacle avoidance through reinforcement learning". In: *NIPS*. 1991, pp. 523–530 (cit. on p. 13).

[61] Sukn-Hwan Suh and Kang G Shin. "A variational dynamic programming approach to robot-path planning with a distance-safety criterion". In: *Robotics and Automation, IEEE Journal of* 4.3 (1988), pp. 334–349 (cit. on p. 13).

[62] Elmer G Gilbert and Daniel W Johnson. "Distance functions and their application to robot path planning in the presence of obstacles". In: *Robotics and Automation, IEEE Journal of* 1.1 (1985), pp. 21–30 (cit. on p. 13).

[63] Johnson YS Luh and Chyuan S Lin. "Optimum path planning for mechanical manipulators". In: *Journal of Dynamic Systems, Measurement, and Control* 103.2 (1981), pp. 142–151 (cit. on p. 13).

[64] Satish Sundar and Zvi Shiller. "Optimal obstacle avoidance based on the Hamilton-Jacobi-Bellman equation". In: *Robotics and Automation, IEEE Transactions on* 13.2 (1997), pp. 305–310 (cit. on p. 13).

[65] Commuri Seshadri and Arindam Ghosh. "Optimum path planning for robot manipulators amid static and dynamic obstacles". In: *Systems, Man and Cybernetics, IEEE Transactions on* 23.2 (1993), pp. 576–584 (cit. on p. 13).

[66] E Freund. "Path control for a redundant type of industrial robot". In: *Proc. Of VII International Sump. On Industrial Robots* (1977), pp. 234–241 (cit. on p. 13).

[67] Hideo Hanafusa, Tsuneo Yoshikawa, and Yoshihiko Nakamura. "Analysis and control of articulated robot with redundancy". In: *IFAC, 8th Triennal World Congress*. Vol. 4. 1981, pp. 1927–1932 (cit. on p. 13).

[68] Tomas Lozano-Perez. "Spatial planning: A configuration space approach". In: *Computers, IEEE Transactions on* 100.2 (1983), pp. 108–120 (cit. on p. 13).

[69] Michael S Branicky and Wyatt S Newman. "Rapid computation of configuration space obstacles". In: *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*. IEEE. 1990, pp. 304–310 (cit. on p. 13).

[70] Tomas Lozano-Perez. "A simple motion-planning algorithm for general robot manipulators". In: *Robotics and Automation, IEEE Journal of* 3.3 (1987), pp. 224–238 (cit. on p. 13).

[71] Anthony A Maciejewski and Charles A Klein. "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments". In: *The international journal of robotics research* 4.3 (1985), pp. 109–117 (cit. on p. 13).

[72] Yunong Zhang and Jun Wang. "Obstacle avoidance for kinematically redundant manipulators using a dual neural network". In: *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 34.1 (2004), pp. 752–759 (cit. on p. 13).

[73] Michael Edwin Kahn and B Roth. "The near-minimum-time control of open-loop articulated kinematic chains". In: *Journal of Dynamic Systems, Measurement, and Control* 93.3 (1971), pp. 164–172 (cit. on p. 13).

[74] Howie Choset and Wade Henning. "A follow-the-leader approach to serpentine robot motion planning". In: *Journal of Aerospace Engineering* 12.2 (1999), pp. 65–73 (cit. on p. 14).

[75] Gregory S Chirikjian and Joel W Burdick. "A geometric approach to hyper-redundant manipulator obstacle avoidance". In: *Journal of Mechanical Design* 114.4 (1992), pp. 580–585 (cit. on p. 14).

[76] Aksel Andreas Transeth et al. "Snake robot obstacle-aided locomotion: Modeling, simulations, and experiments". In: *Robotics, IEEE Transactions on* 24.1 (2008), pp. 88–104 (cit. on p. 14).

[77] Robert Weinstock. *Calculus of variations: with applications to physics and engineering*. Courier Corporation, 1952 (cit. on p. 22).

[78] Donald Smith. *Variational Methods in Optimization*. New York: Dover Publications, 1998. ISBN: 0486404552 (cit. on p. 22).

[79] Laurent Bernardin et al. *Maple programming guide*. Citeseer, 2011 (cit. on p. 29).

[80] The MathWorks, Inc. *The Student Edition of MATLAB: Student User Guide*. The MATLAB curriculum series. Mathworks, 1992, pp. xiv + 494. ISBN: 0-13-856006-4 (cit. on pp. 29, 55, 69).

[81] Edwin H Spanier. *Algebraic topology*. Vol. 55. 1. Springer Science & Business Media, 1994 (cit. on p. 65).

[82] Martin Gardner. *Mathematical carnival: a new round-up of tantalizers and puzzles from Scientific American*. Vintage Books, 1977 (cit. on p. 66).

[83] Alan H Barr. "Superquadrics and angle-preserving transformations". In: *IEEE Computer graphics and Applications* 1.1 (1981), pp. 11–23 (cit. on p. 66).

# Publications from the thesis

## Journals

[1] Midhun S Menon, GK Ananthasuresh, and Ashitava Ghosal. "Natural motion of one-dimensional flexible objects using minimization approaches". In: *Mechanism and Machine Theory* 67 (2013), pp. 64–76.

[2] Midhun S. Menon, B. Gurumoorthy, and Ashitava Ghosal. "Efficient simulation and rendering of realistic motion of one-dimensional flexible objects". In: *Computer-Aided Design* 7576 (2016), pp. 13 –26.

## Conferences

[1] Midhun S Menon et al. "Simulation of Length-Preserving Motions of Flexible One Dimensional Objects using Optimization". In: *Proceedings of the 13th IFToMM World Congress*. Guanajuato, Mexico, 2011.

[2] Midhun Sreekumar Menon, B. Gurumoorthy, and Ashitava Ghosal. "Advances in Robot Kinematics". In: ed. by Jadran Lenarčič and Oussama Khatib. Ljubljana, Slovenia: Springer International Publishing, 2014. Chap. Efficient Resolution of Hyper-Redundancy Using Splines, pp. 447–456.

[3] Midhun Sreekumar Menon and Ashitava Ghosal. "Obstacle avoidance for hyper-redundant snake robots and one dimensional flexible bodies using optimization". In: *Proceedings of the 14th IFToMM World Congress*. Taipei, Taiwan, 2015.