Control of Objects with a High Degree of Freedom

He WANG

Doctor of Philosophy

Institute of Perception, Action and Behaviour

School of Informatics

University of Edinburgh

2012

Abstract

In this thesis, I present novel strategies for controlling objects with high degrees of freedom for the purpose of robotic control and computer animation, including articulated objects such as human bodies or robots and deformable objects such as ropes and cloth. Such control is required for common daily movements such as folding arms, tying ropes, wrapping objects and putting on clothes. Although there is demand in computer graphics and animation for generating such scenes, little work has targeted these problems.

The difficulty of solving such problems are due to the following two factors: (1) The complexity of the planning algorithms: The computational costs of the methods that are currently available increase exponentially with respect to the degrees of freedom of the objects and therefore they cannot be applied for full human body structures, ropes and clothes . (2) Lack of abstract descriptors for complex tasks. Models for quantitatively describing the progress of tasks such as wrapping and knotting are absent for animation generation.

In this work, we employ the concept of a *task-centric manifold* to quantitatively describe complex tasks, and incorporate a bi-mapping scheme to bridge this manifold and the configuration space of the controlled objects, called an *object-centric manifold*. The control problem is solved by first projecting the controlled object onto the task-centric manifold, then getting the next ideal state of the scenario by local planning, and finally projecting the state back to the object-centric manifold to get the desirable state of the controlled object. Using this scheme, complex movements that previously required global path planning can be synthesised by local path planning.

Under this framework, we show the applications in various fields. An interpolation algorithm for arbitrary postures of human character is first proposed. Second, a control scheme is suggested in generating Furoshiki wraps with different styles. Finally, new models and planning methods are given for quantitatively control for wrapping/unwrapping and dressing/undressing problems.

Acknowledgements

First I am mostly grateful to my wife who has been supporting me for many years and who has helps me on so many levels. She is a personable friend with whom I share my feelings, she is a cheerful friend with whom I hang out all the time, she is a wise friend who gives constructive suggestions and criticism. I certainly cannot finish my PhD without her.

I would like also thank my parents and grandparents who have played indispensable roles in my life. They have been an important part of every decision I have made, in both my career and life. Nobody can love me in the way they do and nobody would make the sacrifice as they have made.

Dr. Taku Komura, my supervisor, surely deserves my special thanks. He is the most important person in my PhD programme. His passion and dedication influence not only me but all the people who know him. His professional knowledge, academic intuition and rigorous logic for many times saved me from straying. His generous and full-hearted support is the key of the progress of this project.

I would also give my thanks to Dr. Kirill Sidorov, Peter Sandilands and Rami Ali Al-ashqar for their great contributions to the project. I also thank Dr. Edmond S. L. Ho and Dr. Hubert P H Shum for many years' sharing and guidance. I must thank Adam Barnett, Joseph Henry and Xi Zhao for their great input to my research.

In addition, I would like to thank Dr. Subramanian Ramamoorthy. As my second supervisor, he always inspired me in so many ways. I would also like to thank Prof. Sethu Vijayakumar, Prof. Bob Fisher and Prof. Ewan Klein for their guidance and help. Also, I would like to thank Dr. Vittorio Ferrari and Prof. David Marshall as my examiners.

Finally, I have to thank all my friends who have given so much help and with whom I enjoyed a great time during my study.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*He WANG*)

Dedicated to my beloved, wise and charming wife, who has given her support without any reservation, who has accompanied me through all the pain and joy, who has witnessed my effort and who tolerates my ill-temper and childishness, sometimes.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Humans deal with objects with large Degrees of Freedom (DoFs) every day. Whenever we fold our arms, tie our shoe laces or put some clothes on, we naturally design and implement complicated control strategies to accomplish these tasks. However, how these control strategies are planned and carried out is not obvious. As a researcher in robotics, computer graphics and animation, I am particularly interested in actively controlling highly deformable objects, mainly because of two reasons

1. There is demand in these areas because the phenomena associated with this kind of objects are universal. You can see them when we put on a sock, wrap a bundle of flowers or do yoga exercises. All these phenomena involve people actively controlling a highly deformable object (rope, cloth or even themselves).

2. Not much work has been done in tackling this problem. A large body of research in computer animation is passive simulation. Usually, they either simulate such objects under very simple constraints such as a piece of cloth free falling or hanging (Baraff and Witkin [1998]; Bridson et al. [2003]) or clothes on characters (Goldenthal et al. [2007]; Wang et al. [2010a]), or introduce user control on top of the simulation (Brown et al. [2004b]). Not much effort has been made into designing systematic approaches of actively controlling these objects to achieve complex tasks such as knotting and wrapping.

A natural question is why existing methods cannot be applied to this problem. The problem is notoriously difficult primarily because: (1) high dimensionality; (2) lack of good descriptions for complex tasks.

The high dimensionality problem is also referred as *the curse of dimensionality* (Bellman [1961, 2003]). This is familiar to researchers who are interested in control problems. In our case, it mainly lies in two folds. First, the configuration space of a highly deformable object is a manifold embedded in a high dimensional space. Since we have little knowledge about the manifold, it usually requires densely sampling to gather the information about the manifold and a global planning algorithm to find out the path. There are mature approaches such as Probabilistic Roadmaps (PRMs) (Kavraki et al. [1996]) and Rapidly-exploring Random Trees (RRTs) (Lavalle et al. [2000]). However, when the DoFs go high, the performance drops significantly and it becomes intractable. Second, global planning algorithms only give valid paths, not ideal ones. Generated motions can be jagged (Wang and Komura [2011]). Although theoretically it is possible to form aesthetic requirements, for instance smoothness and naturalness, into constraints, it makes the system even more complicated and slower.

The second problem is the lack of good descriptions of the tasks. It appears when the task involves multiple objects. In such cases, we need new representations to describe the task. For example, when wrapping a ball with a bag, we need to evaluate how much of the ball is already wrapped and how much is left. Hence, the solution must evolve from single-object-centric to relationship-aware. And this kind of second-level information (let us call the information about individual objects the first-level information) has to be taken into account in the motion control problem.

Whenever the dimensionality of the problem in interest is high, an intuitive idea is to reduce the number of dimensions of the whole scene. Various statistical methods have been developed and introduced to solve problems in character animation or cloth simulation. However, they are not very conveniently applicable for our purpose for two reasons. First, we aim for controlling deformable objects under severe deformations where it is very difficult to capture the motion. Second, even if we manage to get the data, a dimensionality reduction algorithm that is solely based on the data itself fails to grasp the important information: the relationship between objects or different parts of the same object, which is crucial for the description of our tasks. Driven by this idea, my thesis will present new representations that reduce the dimensionality of the problem based on the task descriptions.

1.2 Problem Definition and Goals

1.2.1 Problem Definition

A highly deformable object is an object that has a large number of Degrees of Freedom. Instances studied in this thesis are human bodies, ropes and cloth. Such an object can be represented by generalised coordinates $q = \{q_1, q_2, ...q_n\}$. The configuration space, $C(q)$, is formed by all plausible configurations of the object. The control problem is to find a path from configuration q_a to q_b, $f(q_a, q_b) = \{q_1, q_2, ..., q_n \mid where\ q_1 = q_a\ and\ q_n = q_b\}$. For our problem, we have another layer on top of this definition. The task-dependent configuration space, $T(q_{all})$, where all plausible scenarios consisting of all involved objects are included. One can imagine that $C(q)$ is a manifold where any point is a configuration that satisfies the inner constraints of the object itself. For instance, a piece of cloth has inner constraints such as stretchiness and bending limits. We call it *object-centric manifold*. Meanwhile, we call $T(q_{all})$ *task-centric manifold*. A simple 2D example of a particle moving around an obstacle has one object-centric manifold (for the particle) and a task-centric manifold (for the moving task). The control strategy for accomplishing the task can be found in the bi-mapping between the two manifolds, see Figure 1.1.

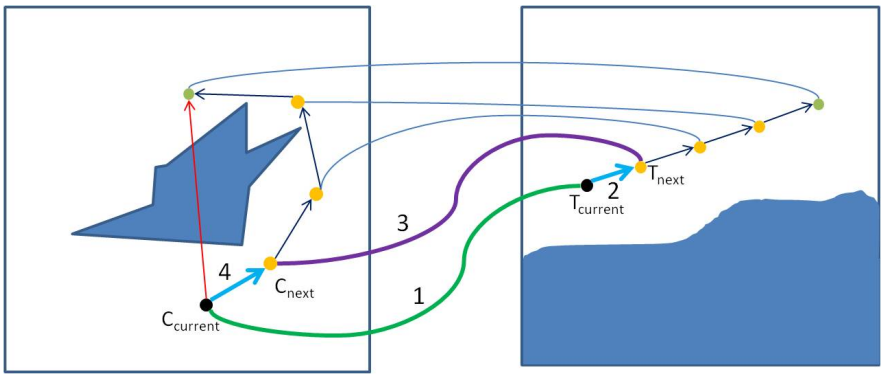

Figure 1.1: The visualisation of manifolds of the 2D particle moving task. Left: the illustration of the manifold, $C(q)$, of the particle where the coloured regions are inaccessible due to the presence of the obstacle; Right: the manifold of the moving task, $T(q_{all})$.

Provided that the particle is placed at the black dot and the task is to move it to the green dot. Figure 1.1 left is the configuration space of the particle. Coloured regions are inaccessible due to the obstacle. As shown by the red line, a linear interpolation scheme requires the particle to pass through the inaccessible region. However, if a

nicely structured manifold can be found to describe the task, it can be reformulated and visualised as the right part of Figure 1.1. Based on the re-parameterisation, the task can be accomplished by linear interpolation in this space. Overall a bi-mapping can be established to plan the motion for the particle:

1. First the current state $C_{current}$ of the particle is mapped onto $T(q_{all})$ and the state of the scenario is evaluated (Line 1).

2. By computing the progress of the task based on the current scenario $T_{current}$, the ideal next state, T_{next}, is found in $T(q_{all})$ (Line 2).

3. After that, the desirable state of the particle, C_{next}, is obtained by mapping the solution T_{next} back onto $C(q)$ (Line 3).

4. Finally, the transition between $C_{current}$ and C_{next} is carried out (Line 4). We call the mapping in step 1 the *forward mapping*, and the mapping in step 3 the *backward mapping*.

For real world applications, the object-centric manifold is usually far more complicated. And the structure of task-centric manifold is not known. Hence, they do not have simple structures such as shown in Figure 1.1. To find such a bi-mapping requires much knowledge of the manifolds. The problem easily becomes intractable when the dimensionality is high. One typical scenario is that when both the particle and the obstacle are replaced by moving deformable objects in 3D. However, we are going to make an important assumption here: although as complex as manifolds are, only some of their subspaces are closely related to the tasks. These subspaces are dominated by some governing properties so that we can use task-related representations and local control algorithms to track the change of the major property in order to design simple and effective control strategies.

Three key elements of this assumption are: **task-related representations**, **governing properties** and **local control algorithms**. Their importance can be understood by going through the four steps of the bi-mapping mentioned above. First, a good task-related representation makes step 1 easy to calculate. The representation converts the path planning problem into a simpler form so that a linear interpolation scheme can be used. More importantly, the progress of the task is quantised. Second, the existence of the governing property greatly reduces the complexity of the planning in step 2. In Figure 1.1, the governing property of this particle moving task is how close the particle is to the goal on the task-centric manifold. So we can control motion easily by moving

the particle directly towards the goal (intermediate positions can be computed via linear interpolation). Finally, the local planning makes sure step 3 and 4 can be conducted quickly and the quality of the generated motion is under control. Every time a new result is computed by linear interpolation in the task-centric manifold, it is mapped back to the object-centric manifold so that the particle can be updated. Because the update is achieved by local motion control, the quality of the motion is easily ensured.

The description is too broad under this setting. So I will further elaborate several typical scenarios in this problem domain as our specific goals of the thesis.

1.2.2 Goals

We narrow down the goals of the research project into three representative problems:

1. Interpolation of arbitrary postures of a human body. A human body is a simple example of deformable object. In animation, a state-of-art technology for making single character animation is keyframing where linear interpolation is used to populate the animation (Parent [2002]). However, whenever occlusions between different body parts are present (see Figure 1.2), linear interpolation will fail. There is no simple and automated way to do this. To solve this problem, we will propose a representation and a planning algorithm that are general enough to handle the interpolation between any two postures.

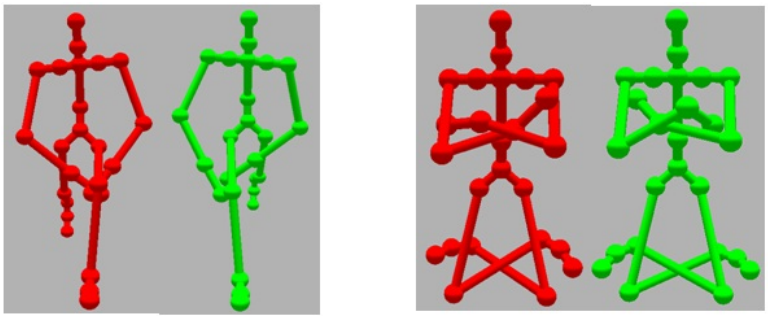

Figure 1.2: Two examples of postures with occlusions between different parts of a human body. In both of them, the red postures are the initial postures and the green ones are the final postures.

2. Constrained knotting and wrapping. Knot theory in mathematics provides good representations and analysis for knots. Nevertheless, when trying to apply them in real world applications, mechanical methods and algorithms are realised. In

Figure 1.3: Examples of various Furoshiki wraps

spite of a number of successful applications (Acker et al. [2005]; Henrich et al. [1999]; Phillips et al. [2002]), it is still not quite straight-forward for animators to make such animations. At the same time, there is not much existing work for wrapping. Inspired by Furoshiki wraps (see Figure 1.3), a real world packaging technique, we aim for control strategies that can be used to generate various wrapping scenarios. Most Furoshiki wraps are generated by a series of knotting and wrapping operations. Since it is very representative, for the sake of simplicity, we will use the term *Furoshiki wrap* instead of constrained knotting and wrapping in the rest of the thesis.

Figure 1.4: Examples of free wrapping. Left: Wrapping a bundle of roses. Right: Wrapping an armadillo model with a bag.

3. Free wrapping. Free wrapping examples are very common in daily life. They are cases when one just wants to put one object inside another like putting a toy into a bag or putting on a sock, see Figure 1.4. They are free wrapping tasks in the sense that there is no constraints saying that which part of the wrapper should wrap which part of the target object. The wrapped object does not have to be a rigid body. Thus we can generalize the problem as one surface wrapping another. Although less constrained, it does not make the control any easier, if not

more complicated. Because in current literature, there is no good way to describe wrapping hence lack of a good representation. For this problem, we will design a topology based control strategy and explain it in detail in Chapter 5.

The class of methods proposed in this thesis is task-centric. They are by no means the ultimate solution in controlling general highly deformable objects. However, we are confident enough to say that they do provide important insights for this area and have the potential to be extended to an even broader range.

Before introducing the methods, we would like to set up an evaluation system and use it later for the assessments of all the methods below.

1.2.3 Key Issues

There are several indicators that can be employed to evaluate the quality of controllers.

1. ***Generality***. This indicator tells how extensively the control design is applicable to certain class of problem. A good controller is supposed to work for a large class of problems. It ideally should capture the nature the problem domain, concisely describe it and lead to a solution that is effective to every individual problem in this domain. The boundaries of methods have to be clarified and fully tested.

2. ***Feasibility***. The method should be easy to design and implement. In addition, robustness, stability and performance are also vital. The controller should deliver steady performance under different settings. The accuracy of the system should remain consistent over time. And the performance has to be high enough so that it generates expected results within a cost-effectiveness range.

3. ***Quality of the generated motions***. This is a very special criterion for computer animation. It is almost purely subjective. Although there is some work to quantitatively measure the naturalness, to our knowledge, there is no generally recognized standards. Nevertheless, we will also discuss this issue over each of the methods.

For different methods, the selection of criteria is a case by case situation. And they will be explained and analysed at the end of every chapter.

1.3 Methodology Overview

In this section, we give an overview of the solutions we propose for the three problems we raised.

1.3.1 Interpolation of arbitrary postures

A commonly used technique in making single character animation is keyframing. For a continuous motion clip, first a series of key frames are given by the user then computers are used to interpolate between adjacent frames. However, when occlusions are present between different body parts, any linear interpolation in the joint angle space is going to fail because of foreseeable penetrations. Usually, this has to be avoided by giving more key frames where body parts are close to each other. Inspired by research in computational geometry, we propose a new model that can capture the essential information about potential collisions. We define an energy function and the energy goes up quickly when body segments come into proximity. Given two postures, the presentation is equipped with a simple local planning algorithm which quickly brings arbitrary configurations into low energy region where interpolation can be conducted and guaranteed to be free from penetrations. See Chapter 3.

1.3.2 Generation of Furoshiki wraps

Wrapping and knotting techniques are used on a daily basis whenever people put on ties or scarves, or wrap up things to carry them. A representative example is Furoshiki wraps. It is a way of gift wrapping by a piece of cloth (usually a square one) in distinctive styles. The manoeuvre of Furoshiki wrap mainly consists of basic operations: winding, knotting and wrapping. Research for these operations is rare in computer animation. Although similar problems are studied in several other areas, the techniques are not ready under the settings of making animation. In this thesis, we will start with a robust local control strategy for primitive knots by introducing an idea called Geodesic Control. This is done by controlling one strand by following a geodesic line over the surface of an object. We first show it is a robust method for simple and primitive knots. Then we model the whole control flow by a Finite State Machine (FSM) to show complex knots can be made. Finally, this method can be extended to strand-surface control. A series of operations of wrapping and knotting are modelled by FSMs and we will show how various Furoshikis can be made by this controller. See Chapter 4.

1.3.3 Free wrapping

Free wrapping aims to describe behaviours of wrapping between objects. This can be interpreted as an object A is surrounded by another object B. Since wrapping is a concept of the spatial relationship between two objects, we first define it mathematically by Winding Numbers. It quantises the progress of wrapping and can be computed easily for primitive shapes (sphere, boxes, and cylinders). Then an extension to arbitrary deformable objects will be proposed by introducing a new electrostatic model. This representation can handle different geometries and topologies. And it is equipped with the simple control strategy for wrapping and unwrapping. See Chapter 5.

1.4 Thesis Structure

The structure of the rest of the thesis is as follows: I will first review some related work in the second chapter. This part covers the state-of-art techniques from various fields that are related to the topic. Then in the following three chapters, I will detail our solutions of the three problems raised above. Finally, conclusions will be made and future work will be discussed.

Chapter 2

Related work

Character motion control is widely studied (Mukundan [2009], Fêdor [2003], Yin et al. [2007]). In contrast, actively controlling objects with larger degrees of freedom is not catching much attention. Generally, the complexity and difficulty of the control problem for both of these issues are shown by the wide spectrum it covers. It involves techniques at different levels, from low-level simulation techniques, local motion control to control based on global path planning. In this chapter, I will review related work within this spectrum from character animation, computational geometry, robotics and physical simulation.

2.1 Motion control in character animation

A part of this thesis is a new interpolation scheme for arbitrary postures and it falls into the research of character animation. In the framework of character animation, there are two layers: underlying model and control. The underlying model provides natural atomic motions. It either comes from data such as Motion Capture Data, or from physical simulation. All these motions are threaded by a high-level control layer. The control algorithms vary from targeting simple tasks like keeping balance to complicated path-planning problems for large crowd simulation scenarios. Our problem lies in this layer. Almost all the work in character animation covers more or less both of the two layers. Next, I will review three classes of work: local motion control, spatial-temporal control and control based on sampling. Local motion control is a class of methods that synthesize movements based on the local state of the system. Spatial-temporal control is where the whole motion is considered by the controller and any local adjustment is propagated along the motion. Control based on sampling cov-

ers the methods in which the control is designed based on sampling the configuration space or work space. During the review, I will explain why they are not suitable for our purpose in the summary of each of them. Finally, there is one special kind of research in this area I would like to review. They are models based the spatial relationship of the characters. And our interpolation algorithm fits into this category.

2.1.1 Local motion control

Local motion controls are usually done in raw representations. Common raw representations are drawn directly from the parameters of the task space or the joint space. Representative local motion control methods in these spaces are Inverse Kinematics (IK) and Proportional-Derivative (PD) control. For these methods, to achieve high level goals, the user input or data is usually required.

2.1.1.1 Local motion control in the task space

Inverse Kinematics is a technique to compute the joint angles given the desired position of the End-Effector. The technique has been used in various systems (Mukundan [2009] Unzueta et al. [2008] Meredith and Maddock [2005] Baerlocher and Boulic [2004] Fêdor [2003]). In Fêdor [2003], the author introduced three methods: An algebraic method based on limb positioning, an iterative method based on Jacobian pseudo-inversion and a heuristic Cyclic Coordinate Descent (CCD) iterative method. In Meredith and Maddock [2005], for the purpose of retargeting motion capture data onto differently scaled models, the author proposed a real-time IK solver with adjusted weights. And in Mukundan [2009], the author compared CCD algorithm and triangulation algorithm on a n-link joint chain, then suggested an improved approach which resembles triangulation algorithm but processes each link only once. In most cases, inverse kinematics problems are usually solved by non-linear programming and have close-form solutions hence can be quickly computed.

2.1.1.2 Local motion control in the joint space

Proportional-Derivative Control is a widely used technique for controlling characters in joint space (Neff and Fiume [2002], Zordan and Hodgins [1999, 2002]). It provides a feedback mechanism:

$$\tau = k_p(\theta_d - \theta) + k_v(\dot{\theta_d} - \dot{\theta}) \tag{2.1}$$

where τ is the torque that is going to be applied on the target joint. θ_d and $\dot{\theta}_d$ are the desired angle and desired angular velocity respectively, while θ and $\dot{\theta}$ are the current angle and angular velocity of the target joint. k_p and k_v are coefficients, also called controller gains. If they are large, the joint stiffness is high, it tends to closely follow the desired angle and velocity which can cause overshooting problems; if they are small, the joint becomes sluggish and always lag behind. Tuning the controller gains is not an easy task. It usually involves iterations of trial-and-error. The variation in stiffness is called impedance. Real human motions usually have low impedance (Takahashi et al. [2001]). More details about PD control gain tuning techniques can be found in Ang et al. [2005].

A more natural control is Antagonistic control. It is a mathematic equivalence of PD control. It mimics the two muscles controlling one joint through contraction and extension (Neff and Fiume [2002]):

$$\tau = k_L(\theta_L - \theta) + k_R(\theta_R - \theta) + k_v\dot{\theta} \tag{2.2}$$

where τ is the output torque, θ is the current angle, θ_L and θ_R are the joint limits enforced by two springs connected to the joint, k_L and k_R are gains and $k_v\dot{\theta}$ is the damping term. Although the desired the angle is not shown in this formula, it is controlled by the gains, k_L and k_R.

After establishing the motor, people start to build up higher-level controllers. For PD Control, usually the controller consists of a series of target configurations. Hodgins et al. [1995] used PD servos as basic building blocks for high-level motion control. Zordan and Hodgins [1999, 2002] controlled the character to track motion capture sequences while responding to external perturbations. Other people tried to improve PD controllers by using anticipatory feed-forward control so that low gains can be used for feedback (Cline et al. [2002]). And Yin et al. [2007] designed simple user interfaces to control motions by giving key postures. The system is driven by Finite State Machines (FSMs) where desired postures are states and transitions are actuated by PD controllers. Based on it, Yin et al. [2008] established an optimisation framework to make the interaction between character and the environment more robust. Coros et al. [2008] extended Yin et al. [2007] so that it optimises solutions given certain information about the task domain. Then the solution domain is analysed to make predictions over a short-horizon span. Finally online execution is carried out by sampling the solution domain. Recently, Tan et al. [2011], inspired by Baraff and Witkin [1998], employed an implicit scheme to reformulate PD controller so that even under infinite

gains, the controller remains stable.

2.1.1.3 Summary

Local motion control methods cannot solve our problem because the solution cannot be found by linear interpolation in raw representations. Local motion control methods focus on motion control between similar postures. The assumption here is that the postures to interpolate are so similar that any intermediate posture (by linear interpolation) is valid (non-penetrated) and nature. For example, when planning the motion of a joint with three Dofs, the angles are changed monotonically until they reach their target values respectively. This kind of atomic motion planning has to work with some higher-level control to generate complicated and task-oriented motions. And the higher-level plan is often done manually by the user by giving data or reference postures. For our problem, the two postures can be very different and occlusions are present. In scenarios shown in Figure 1.2, the solution does not exist in the solution space of linear interpolation. At the same time, user input and data is not desirable for us either because we want to free the user from the labourious input. However, if the high-level planning is handled by some other method, the local motion control methods can be useful in synthesising natural motions, hence might be a good complement of our method.

2.1.2 Spatio-temporal control

Space-time optmisation and its variations form this category. Given an initial trajectory, it updates the motion spatio-temporally such that a given criteria is optimised.

The human body is under-actuated, it needs to apply forces to the environment and makes use of the reaction force (under Newton's third law of motion) to change its global position and orientation. Hence, perceiving the target character motion and the environment as a whole has the potential to solve many problems met by methods mentioned above. This requires a framework that comprehensively brings all factors under the same system and leads to a solution that satisfies all constraints. Space-time optimisations are designed for this purpose. Because physical validity is not a sufficient condition of natural motion (Wei et al. [2011]), a common pattern of this kind of methods consists of an objective function and constraints. The objective function contains all the expectations towards the final motion and the solution is the minimiser of the function with respect to all the constraints. Wei et al. [2011] called it "minimal principle". Such criteria are like minimal energy, minimal torque, minimal momentum

and so on. A general formulation is:

$$\underset{q}{\text{argmin}}\{O_1 + O_2 + ... + O_j\} \quad subject\ to \quad \{C_1, C_2, ..., C_k\} \tag{2.3}$$

where $O_j, j \in [1,m]$ and $C_k, k \in [1,n]$ are objective function terms and constraints.

The first work which brought the idea into computer graphics community is Witkin and Kass [1988] (Wei et al. [2011] Geijtenbeek et al. [2011]). In this paper, the author generalised a framework for producing physically valid animation by specifying four key elements: task (what to do), conditions (how to do it), physical structure and resources (the actuation of the system). It laid the ground for many of the coming optimisation-based methods (Liu [2008]; Liu and Popović [2002], Safonova et al. [2004], Abe et al. [2004]).

This method has also been applied on walking motions. Liu and Popović [2002] synthesised natural motions by enforcing a set of linear and angular momentum constraints. While in Liu et al. [2005], more biomechanical factors including tendons, ligaments and muscles are considered to produce natural animation. Similar modelling can also be found in Semwal et al. [2006] and Lee and Terzopoulos [2006]. High level features like motion styles are also targeted by researchers. In Silva et al. [2008], the author incorporated styles of the reference motion and balance control and solved it as a Quadratic Programming (QP) problem. Their model was simplified into a three-link human skeleton and foot contacts was modelled by a joint.

Due to the intense interactions between the character and the environment, robust controllers for complex environments are also designed such as walking on uneven or slippery terrains (Ye and Liu [2008], da Silva et al. [2008], Wang et al. [2009, 2010b]). In Wang et al. [2009], the author modified the SIMBOCON (Yin et al. [2007]) by optimising power consumption, angular momentum as well as other important features normally seen in human walking like active toe-off, passive knee swing, etc. da Silva et al. [2008] tried to generate physically valid motions by tracking a reference motion under different environmental settings. For simulating an even more responsive and intelligent character, Jain et al. [2009] made the character reaching out for objects like walls and bars to keep balance in a drastically changing environment.

The difficulty of solving such a system increases as the number of constraints and objective function terms goes up. To mitigate this problem, de Lasa and Hertzmann [2009] and de Lasa et al. [2010] used prioritized objectives. The top objective is always satisfied by minimising the objective function while the other objectives are achieved under the constraint that they do not violate all previously satisfied objectives.

2.1.2.1 Summary

Spatio-temporal control cannot solve our arbitrary posture interpolation problem. Admittedly, these methods plan motions based on more global information than local motion control methods thus have many advantages. For instance, they consider the whole motion instead of only the current frame. And it has the ability of incorporating more factors such as contacts and perturbations into motion control. Any change can be correctly propagated across the whole motion. Nevertheless, Spatio-temporal control cannot solve our problem because they assume the task space is empty so that no complicated planning is needed. For example, in Witkin and Kass [1988], the user can make a lamp model jump to a certain point by giving the position of that point. They assume the space between the lamp's initial position and the highest position is empty so that the path is solely dependent on the dynamics. For our problem, this is not the case. There are several actively controlled body parts moving in the same task space. Spatio-temporal control does not capture enough information to resolve occlusions. However, it is also a good complement of our method because our method can give a valid motion as a good initial guess of the optimisation.

2.1.3 Control based on sampling

Control methods based on sampling are very powerful tools for motion control. They are usually built up on a sampling method which is equipped with a local motion controller. After sampling, a data structure is constructed from samples and paths over the learned manifold are found by global path planning algorithms. In animation, they can be divided into two categories based on the sampling methods: automatic sampling (Lavalle et al. [2000], Shapiro et al. [2007]) and pre-sampling (Mukai and Kuriyama [2005], Hsu et al. [2005]). In this section, I will review existing methods that fall into these two categories.

2.1.3.1 Control based on automatic sampling

Control methods based on automatic sampling targets problems similar to ours. To plan a motion sequence between two configurations, they usually require sampling the configuration space and an algorithm to find paths.

Rapidly-exploring Random Trees (RRTs) (Lavalle et al. [2000]) is a sampling method for global motion planning. Different variations are present. Yamane et al.

[2004] proposed a framework for a character to manipulate objects. The environmental setting in this paper is quite general. The character can move objects through confined spaces. They used RRTs in the object's configuration space and use Inverse Kinematics to plan the movements of the character. In order to generate natural motions, they resorted to motion capture data and biased the IK solution towards those solutions. Shapiro et al. [2007] suggested another framework that mainly handles full body motions instead of focusing on arm motions like Yamane et al. [2004]. They used a bidirectional version of RRT and did not use example motions. However, they had to employ some heuristics to make motion more natural. Although their results are less likely to be as equally natural, their method is faster. Koga et al. [1994] targeted problems like re-grasping with both hands of a character. For the motion planning for hands, they suggested a Randomised Path Planner (RPP) to sample the task space around the target object.

As powerful as these methods are, they cannot be directly used for our purpose, mainly for two reasons: (1) These tools are usually heavy and expensive. The performance drops significantly when the dimensionality of the problem goes high. (2) They only give valid motions, not ideal motions. The generated motions can be jagged and the motion style can be unnatural. In addition, there is no easy way to incorporate aesthetic requirements into the framework without even further compromising the performance.

2.1.3.2 Control based on pre-sampled data

Data driven approaches fall into this category. They work on pre-sampled data, either from motion capture data or user input. This kind of work comprises a large body of research work in character animation. Next, I will review the literature in areas including motion blending, motion streaming and hybrid models.

Blending sample data

Motion blending is generating new motions by blending existing ones. A general formulation of blending two motion sequences can be found in Mukai and Kuriyama [2005]:

$$\mathbf{M}(\mathbf{c}) = \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \mathbf{b_i}(\mathbf{c})\mathbf{M_i} \tag{2.4}$$

where n is the number of motion sequences, c is the control parameters of motions in an abstract space and b is some kernel function.

For example, a linear interpolation scheme based on two motions can be written

as:

$$\mathbf{M}(\mathbf{c}) = \alpha\mathbf{M_1}(\mathbf{c}) + (\mathbf{1}-\alpha)\mathbf{M_2}(\mathbf{c}) \quad \mathbf{c} \in [\mathbf{c_{t_0}}, \mathbf{c_{t_1}}] \tag{2.5}$$

where M_1 and M_2 are two given motions both normalized into scale $t \in [0,1]$, c_{t_0} and c_{t_1} are two sets of parameters when $t_0 = 0$ and $t_1 = 1$. Finally, α is the blending control and $\alpha = \frac{c_{t_0}}{c_{t_1} - c_{t_0}}$.

Usually, the duration and speed of different motions are not synchronised. Then direct blending leads to unnatural motions (Shum [2010]). Before blending, motions are usually aligned by Dynamic Time Warping which has been used in speech recognition. The general idea is to find correspondences between two motions such that a global cost function measuring the difference of two motions are minimised. This technique was used in Bruderlin and Williams [1995] which is one of the early works of motion blending. The authors treated the input motions, parameterised by joint angles, as sequences of signals. Then they use multi-resolution filter to process the motion clips and interpolate them. However, motions with distinctive styles are hard to synchronise and blend. Kovar and Gleicher [2004] suggested a distance metric to find "close" motions. Different motions are organised into groups. With enough samples, they could create a continuous space of motions by blending. Different criteria are also used for warping. Both Park et al. [2004] and Ménardais et al. [2004] used foot contacts to establish correspondences. Park et al. [2004] used an incremental time warping and motion blending to establish a motion transition graph which was used for online motion planner. They also blended joint angles based on quaternion algebra. Hsu et al. [2005] built up an optimisation framework to do both time and space warping, and used a Linear Time Invariant (LTI) model to translate different styles across a spectrum of motion sequences.

Aside from deterministic methods, statistical methods have also been used for motion interpolation. They are mainly related to multivariate analysis. Techniques like K-means and Expectation-Maximisation (Molina-Tanco et al. [2000] and Lee et al. [2002]) are employed for grouping similar motions. Principle Component Analysis and Scaled Gaussian Processes Latent Variable Model are applied to reduce the dimensionality of the data (Safonova et al. [2004] and Grochow et al. [2004]). Mukai and Kuriyama [2005] used Universal Kriging to decompose motions data into a trend component and its residual. The trend component is deterministic and it can be computed by Linear Regression. The residual is supposed to be a realization of a random function with some properties. In their work, given the computed trend model and assumptions on the residual, the parameters of kernel functions in Equation 2.4 were

estimated by optimising a variogram function.

For deterministic and statistical methods for motion blending, they both assume that the path between two blended postures lies in the C_{free} which is the free space of the configuration space. Usually, it can be guaranteed if enough samples are drawn. However, it is not very desirable because densely sampled data is not always available to animators, especially when postures to blended are arbitrary.

Motion streaming

Rather than modeling and controlling motion generation based on raw data, researchers organise motion capture data into units and link them based on similarity, so that they can focus on higher level goals. The granularity of motion units depends on the application. It can be a single frame, or a cyclic motion (for example, a walking cycle in Treuille et al. [2007]), or even an action (Shum et al. [2008b]). Once motion units are formed, one popular method for organising them is Finite State Machine (FSM). Motion units are treated as individual states and transitions between them are the edges of the FSM. Based on the FSM, a data structure called Motion Graph (Kovar et al. [2002]) (Figure 2.1) is widely used. In this data structure, the graph nodes represent motion units. Motion synthesis is conducted as a path-planning problem (Lo and Zwicker [2010]), or a reinforcement learning problem (Shum et al. [2010]). Edges in the graph are transitions between different postures or different actions. The kinematic similarities between nodes determine whether a connection should be established.

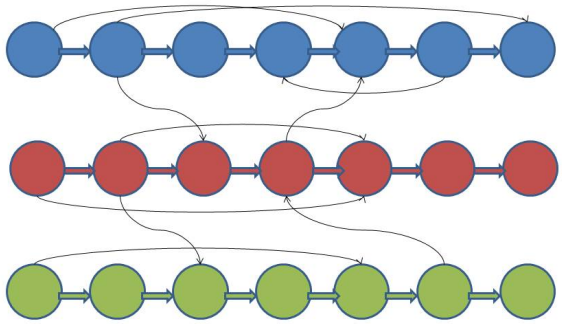

Figure 2.1: Motion Graph

Once the graph is built, different goals can be defined to find paths on this graph to stream the units. Usually, high-level control is the main focus, like navigation, collaboration, competition or user interaction. Kovar et al. [2002] made the character walk

along a path specified by the user. The resulting motion was the minimiser of the deviation of the generated trajectory from the user-defined path. Choi et al. [2003] sampled a complex environment to construct a roadmap, then built a global path planner working together with a local planner, and finally, it was used to guide a character through confined spaces. Lo and Zwicker [2010] first laid the motion graph onto a complex environmental setting with both static and moving obstacles. Then they started searching the graph from both the initial node and target node to shorten the searching time in the graph. Shum et al. [2008a] first built up a motion graph at the action level, then incrementally expanded a game tree for two fighting avatars so that they select appropriate actions to generate a continuously fighting scene. By patching individual fighting scenes in a spatial-tempo manner, long complicated fighting scenarios were produced. Later, Shum et al. [2010] extended the min-max search idea to make the character satisfy two possibly conflicting goals: user control and automatic action selection for fighting or collaboration.

Combining samples with physical-based models

Previous methods are either purely data-driven, or based on motion equations with limited usage of data as high-level control. It is hard for them to produce natural character animation which is also responsive to external impact. The fact that realistic human motions lie in a subspace of solution space (Wei et al. [2011]) drives people to introduce motion capture data as an "anchor" or "reference" where physically valid solutions can be filtered and the finally solution is close to natural motion region. In this section, I review some hybrid approaches that try to combine both of the previous methods. The reason I call them hybrid approaches because both data and physically-based models are equally involved. The final solution space is found out by enforcing constraints from both models.

Instead of directly utilising motion capture data, people strive for understanding the underlying representation of motions. This representation contains a large amount of information from basic dynamics to motion styles. Hence, machine learning techniques are combined with other techniques to generate realistic animation too. Wang et al. [2006, 2007] introduced Gaussian Process Dynamical Models to learn a low dimensional latent space associated with dynamics and a map from the latent space to the data space. They applied this model onto human motion sequences to construct a nonparametric model for dynamic systems with uncertainty. The system could build up a robust model based on a limited set of data samples. Ye and Liu [2010] used Scaled Gaussian Processes Latent Variable Models with back constraints (Lawrence

[2006a,b]) to learn a underlying dynamic model. The model was used for predicting the next ideal pose. If there were perturbations, a dynamic equation was solved with constraints added at the support foot. When the new pose was solved, the latent model was updated accordingly. By doing this, their system can control human body in a low dimensional space while taking external impact into account at the same time. Min et al. [2009] suggested a framework to learn a deformable model over given motion sequences, then synthesise motions by Maximum a Posterior (MAP). The system can handle user input like pre-defined end-effector trajectories. The system was extended in Wei et al. [2011] by incorporating a dynamic system into the framework. They first learned a Gaussian Process mapping between the generalized force and the kinematic data, what they called force field, then a probabilistic model was learnt from three components: the initial pose sampled from a Gaussian Mixture Model of different initial postures, the force field and a randomised dynamic system. The final result was solved by MAP where the final solution was the most likely motion under constraints.

For our problem, there are mainly two difficulties to use global planning approaches with pre-sampled data: (1) It is either hard to capture the motion of highly deformable objects such as rope and cloth, or meaningless to do that for our interpolation problem. Especially for the latter one, avoiding the data input and user input is part of the goal of the research. (2) The effectiveness of the motion planning is bounded by the pre-sampled data. Given any two arbitrary configurations, one has to find the same or at least similar configurations in the data. This is not convenient and sometimes even impossible.

2.1.3.3 Summary

Control methods based on sampling are not suitable for our problems mainly due to the high complexity of their sampling methods and path planning algorithms. For automatic sampling, the performance drops fast when the dimensionality of the problem goes high. Plus, stochastically sampled data and the path planning algorithms make it very difficult to control the quality of generated motions. On the other hand, existing technologies are not adequate for us to get quality motion data of highly deformable objects. Furthermore, the solution is limited by the pre-sampled data in this kind of approaches.

2.1.4 Spatial-relationship based models

There is a class of research that tries to model motions from a different perspective: the spatial relationship. A part of our research falls into this category too.

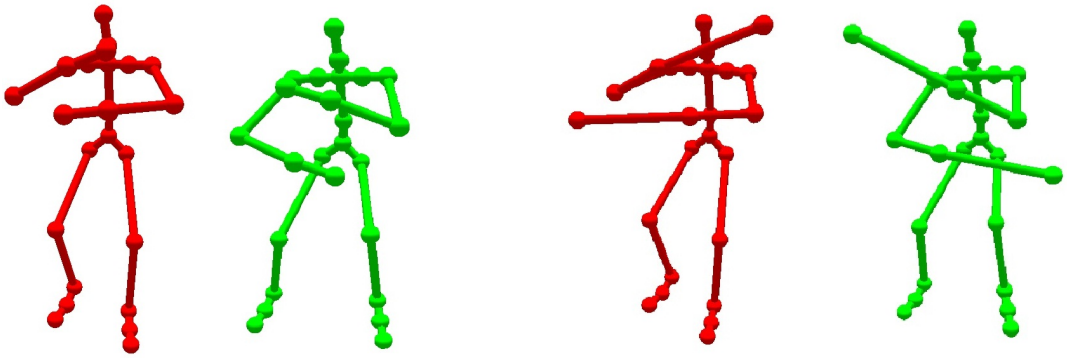

Figure 2.2: Left: an example of a character with arm crossed. Right: the same postures with lengthened forearms.

When people are in continuous and steady contacts, like one person piggy-backing another, the motion planning becomes difficult. The problem is also present in a single character animation (Figure 2.2). On the left of Figure 2.2, it is clear to see that when moving one configuration towards the other, any linear interpolation scheme will fail due to the foreseeable inter-penetrations of two arms. This can be seen more clearly on the right of Figure 2.2 with lengthened forearms. Similar motion planning problems like this have been raised by Ho and Komura [2009a,b]. A similar scenario was also shown in their papers (Figure 2.3)

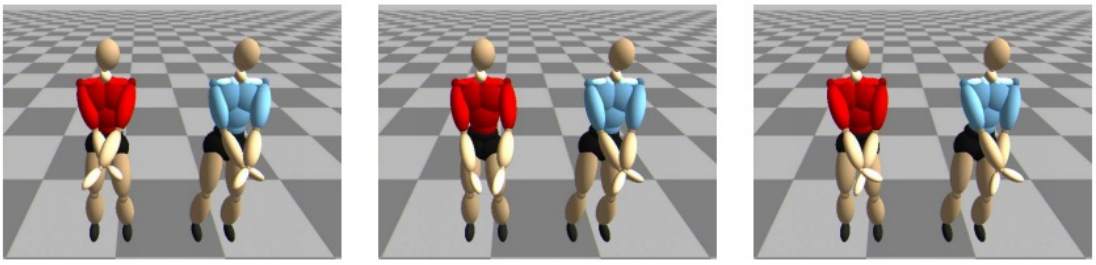

Figure 2.3: an example of path planning connecting two configurations with crossed arms(Ho and Komura [2009b])

In Ho and Komura [2009b], they introduced Topology Coordinates to encode the knots that widely exist in human postures. The path planning was done via a bi-mapping between the space of generalised coordinates and the knot space. First, they

represented postures via knots, then projected the configuration onto a matrix representation. By manipulating the matrix and mapping it back to the generalised coordinates of the character, they successfully un-knotted and re-knotted the character. Later, Ho et al. [2010] suggested another method, Interaction Mesh, for motion retargeting while preserving spatial relationship between characters. In this work, the spatial relationship between two characters was represented by a volumetric mesh computed by the tetrahetralisation of joint positions. When transferring and retargeting motions, the volumetric mesh was kept the same by preserving the Laplacian of the mesh.

This class of research sees the motion control problem from a different angle and inspires my research. However, for our problem, their methods are not desirable either. For arbitrary posture interpolation, although being aware of the spatial relationship between different body parts is crucial, Topology Coordinates more focused on manipulating a knot formed by two linkages, while interpolation has to consider all of them at the same time. In addition, Interaction Mesh is not suitable because it is for motion retargeting.

2.1.5 Summary

In this section, we reviewed existing planning methods in character animation. We investigated them from the local level to the global level. To solve our problem, local motion control methods do not have enough information. While global motion control methods require empty task spaces. In addition, control methods based on global path planning are either slow, hard to control the motion quality, or require pre-sampling data which is impractical for us. Lastly, although the spatial-relationship based models are inspiring, they are not directly applicable to our problems.

2.2 Planning in computational geometry

The research of deformable object control is also present in computational geometry. Finding out continuous motions to change the configurations of this kind of objects is an indispensable part of the research in computational geometry. A typical example is folding/unfolding problem. Our interpolation scheme is highly inspired by the research on this particular problem. Next, I will review existing work in this area and in the summary explain why there are still challenges to directly use them for character animation.

Folding and unfolding in computational geometry has been extensively studied in the past decade (Demaine and O'Rourke [2005]). However, the problem was implicitly raised as early as the 1500's (Durer [1977]). Generally, researchers in this area are interested in how objects, such as linkages, pieces of papers and polyhedra can be changed to certain target configurations subject to constraints. In this section, I will mainly focus on linkage folding/unfolding problem.

2.2.1 Problem definition

A *linkage* consists of a collection of rigid line segments connected with joints at their end points (vertices). The joints are usually universal joints, permitting the full angular range of motions (For example, from 0 to π for a universal joint in 2D). A linkage is regarded as *simple* if there is no crossing line segments. There is no limit on the topology of the linkage. It can be an open chain, a closed chain, or a tree (Demaine and O'Rourke [2005]). A linkage folding problem is moving the vertices in the R^d space, where d is degrees of freedom of the linkage, often under the length constraints of the line segments. Define a *configuration space* for linkages which contains all possible configurations of the given linkage. Paths in this space correspond to folding or unfolding sequences. The folding/unfolding problem is to find a path between one point and another. Here we only discuss unlocked configurations. Unlocked configurations are those configurations between any two of which the linkage is free to transit. For an unlocked configuration space, there is at least one path connecting any two configurations.

2.2.2 Simple configuration folding in 2D

For a lot of applications, self-crossing is not allowed. Hence researchers also study the folding/unfolding problem under the constraint that the linkage must stay *simple* all the time. In such research, a fundamental question is, given a linkage, whether it is possible to fold it between any two given simple configurations (Figure 2.4 Top). Since the folding process is reversible so it is actually asking whether there exists at least one intermediate configuration, or *canonical* configuration, that we can fold any linkage into. If such configuration exists, it means given any two configurations, A and B, we can fold them into this canonical configuration respectively (Figure 2.4 Middle). Then concatenating the folding motions of one configuration, *M(A)*, with the reversed one of the other, *Reverse(M(B))*,gives us a series of motions that folds A into B and vice

versa(Figure 2.4 Bottom). Based on this concept, simple linkage folding is studied for

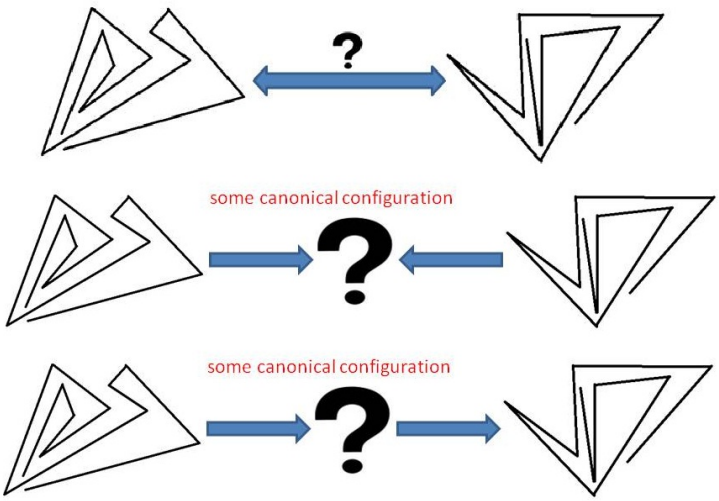

Figure 2.4: Top: Is there a canonical configuration?; Middle: If there exists a canonical configuration, then both linkages can be folded into it; Bottom: Concatenating one folding motion with the reversed folding motion of the other gives a continuous folding motion from one to the other.

three types of linkages: polygonal arcs, polygonal cycles and polygonal trees. Then

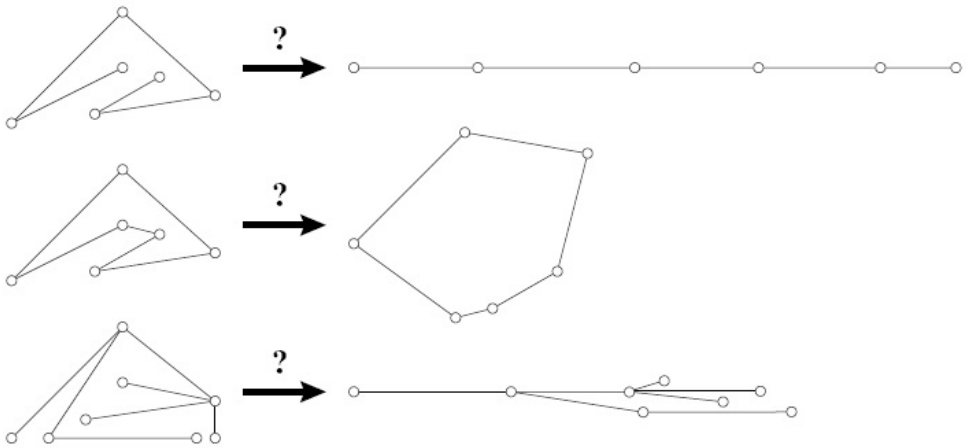

Figure 2.5: Three types of linkages and their respective canonical configurations. Demaine and O'Rourke [2005]

the fundamental question is decomposed into three: If any simple polygonal arc can be straightened, if any polygonal cycle can be convexified and if any polygonal tree can be nearly flattened as suggested in Figure 2.5.

Depending on the space in which linkages are embedded, different results have been shown so far. In 2D space, Connelly et al. [2000] have proven that all polygonal

arcs can be straightened and all polygonal cycles can be convexified. They defined a strict expansive motion over 2D linkages. More importantly, the existence of such local infinitesimal motions was also proven. For continuous global expansive motions, a non-linear minimisation problem was established to find out a unique direction $\mathbf{v}$ for every vertex. In such a direction, the lengths of line segments are preserved and the distances between vertices with any other non-incident line segments are kept or increased. Later, easier implementations were designed by Cantarella et al. [2004] and Iben et al. [2009]. Cantarella et al. [2004] parameterised arcs and cycles by the angles between any connected links. They defined a repulsive energy function, an elliptic distance between any two links that are not directly connected, so any proximity of two links would increase the energy. Under this setting, a series of expansive motions were calculated by following the negative gradient of the energy function. They also proved the existence of the gradient flow. Based on this idea, Iben et al. [2009] designed an interpolation for any two 2D arcs and cycles. The beauty of this interpolation scheme is that it is local and free from local minima, provided the existence of canonical configuration and gradient flow.

Although the unfolding motions of arcs and cycles are solved, polygonal trees in 2D space are shown not always flattenable by Biedl et al. [2002]. In their paper, they showed a n-pedal tree jointed at a center whose branches are inter-locked by each other (see Figure 2.6). The author also proved that there are at least two equivalent classes

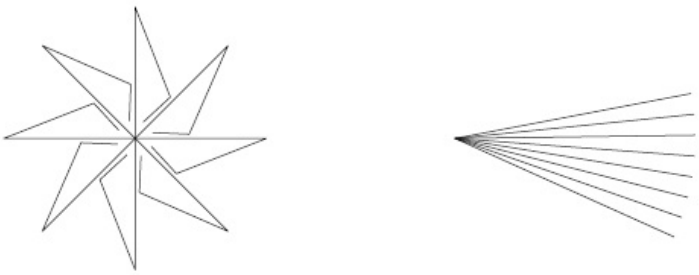

Figure 2.6: Left: a n-pedal tree, Right: a flat tree. The left one cannot be unfolded to the right one in 2D without crossing. Biedl et al. [2002]

of configurations that are also not flattenable. Connelly et al. [2002] found another locked tree configuration (Figure 2.7). They showed how to make many classes of locked planar linkages via linear programming by extending rigidity theory.

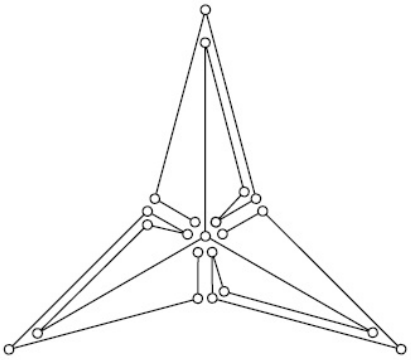

Figure 2.7: A locked tree.Connelly et al. [2002]

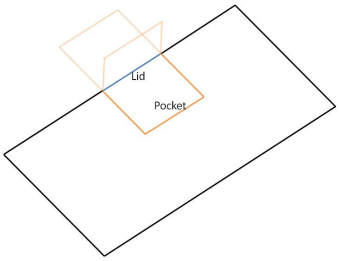

Figure 2.8: A non-convex polygon with a pocket and a flipping operation.

2.2.3 Simple configuration folding in 3D

Linkage folding/unfolding in 3D is a much more difficult problem. The question was initiated earlier by Erdos [1935] back in 1935. Starting from a polygon in a plane, he defined a *pocket*, a region bounded by a subchain of a given polygon edges, and a *lid* of the pocket which is an edge connecting the endpoints of the subchain. A rotation operation is a flip of the subchain around its lid by 180°. Such an operation flips a part of the polygon then land it back onto the same plane (See Figure 2.8). Under this framework, he asked the question: whether it is possible to convexify polygons by a finite number of simultaneous flips.

Soon, the problem that simultaneous flips can cause crossings was argued by Nagy. [1939] (Figure 2.9). However, Nagy proved that if one flip is done at one time, it is sufficient to convexify any such polygon by a finite number of flips. Even so, the number of flips needed increases arbitrarily with respect to the number of vertices. To figure out the boundary of the flips, people have been trying to find the shortest and longest sequence of flipping operations. The complexity of finding longest sequence is weakly NP hard (Aichholzer et al. [2002]).

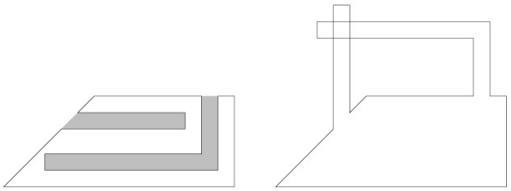

Figure 2.9: Simultaneous flips can cause self-crossings. Nagy. [1939]

Despite that, the flip method inspired other researchers. Different variations came out in the following several years. Biedl et al. [2001] suggested a modified version by sequentially lifting vertex by vertex to a plane that is orthogonal to the plane where the original polygon lies. While lifting, whenever a non-convex polygon is found in this orthogonal plane, it is convexfied. See Figure 2.10

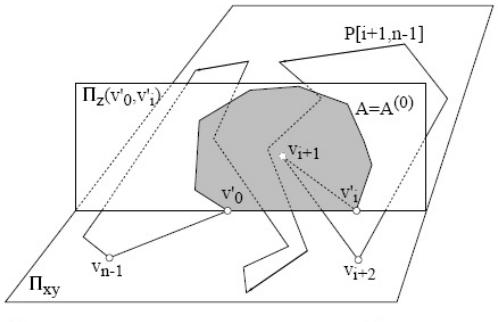

Figure 2.10: Vertices flipped sequentially onto a orthogonal plane while keeping the flipped part convex. Biedl et al. [2001]

Unlike convexifying polygons in 3D, straightening polygonal arcs lying in a plane or the surface of a convex polyhedron is relatively easier (Biedl et al. [2001]). A similar idea can be applied on planar arcs by lifting one link by another to a vertical line. For the surface of a convex polyhedron, when lifting bars, make sure the lifted prefix remains normal to the facet at all times.

If linkages cannot be embedded in a plane, the possibility of knotted configurations emerges. Under such circumstances, there might not be any motion to straighten the arcs or convexify the cycles. An example can be found in Figure 2.11.

Due to the existence of such complex configurations, finding a motion sequence to straighten arcs become more difficult. Actually, Alt et al. [2003] has proven that it is PSPACE hard to decide whether a 2D tree or 3D polygonal arc can be folded between two given configurations. However, for certain linkages, there is a quick solution. If arcs or cycles have *simple* orthogonal projections, where there is no self-crossing,

Figure 2.11: Two non-trivial locked chains. Cantarella and Johnston [1998]

they can be projected onto a plane where they remain *simple*, then straightened or convexified (Biedl et al. [2001] and Calvo et al. [2001]). Such a plane can be computed in polynomial time (Bose et al. [1999]).

2.2.4 Summary

Although the folding/unfolding problem of 2D polygonal arcs and cycles has been solved, to use it for 3D character animation, we still need to adapt the algorithm because: (1) Tree structures in 3D have been proven not always foldable. We need to test if a special tree structure like human skeleton is foldable. (2) For character animation, there are other constraints such as joint limits, or positional constraints for the purpose of motion design.

2.3 Planning for highly deformable objects in robotics

Researchers in robotics have been investigating how to actively control highly deformable objects for a long time. In terms of planning, some tasks are very similar to ours, such as making knots or folding paper. However, the methods they use are not directly useful. In this section, I will first review the control strategies for deformable objects in robotics, and then summarise the reasons that these strategies are not suitable in achieving our goal.

2.3.1 Local planning

Early work primarily focused on using force control to change the state of the target object. This was done by modelling and controlling of robots based on contact dynamics. It is a challenging topic because the system needs to evaluate parameters and adapt the controller. Katic and Vukobratovic [1998] proposed a PD controller to apply constant force onto a deformable 1-D surfaces without assuming any prior knowledge

of the environment characteristics. A neural network was used to learn the relationship between the force applied by the end-effector and the environment type. Then it was used to evaluate the gains of the PD controller. However, it worked only in one dimension. Tarokh and Bailey [1996] also focused on manipulating 1-D deformable surface. They used a fuzzy controller. Parameters of the controller were initialised and adapted later when the manipulator came into contact with the environment. Venkataraman et al. [1993] used neural networks to identify environment parameters. It was trained on nonlinear 1-D deformable surface, and then evaluated parameters like stiffness and damping coefficients were fed into a controller. People also tried to work with 2D surfaces. Al-Jarrah and Zheng [1998] used a nonlinear function to model a sheet's bending force and moment. When the manipulator tried to bend the sheet into a target configuration, the gains of the controllers were adjusted based on the deviation of the generated motion from the desired one. However, they could only handle one DoF deformation. Such research was mainly trying to find the right parameters for controllers to apply correct forces.

This kind of work focuses on parameter evaluation for force planning to change the configuration of the object by a limited amount. For our problem, they suffer from the similar problems as mentioned in section 2.1.1. They cannot handle the motion planning for severe deformations and resolve occlusions.

2.3.2 Global planning

Global planning algorithms are powerful tools for complicated control problems. They are usually equipped with models and representations that contain domain knowledge. In this section, I will focus on global planning algorithms based on sampling and their applications in motion planning for articulated bodies, manipulation of linear deformable objects and robotic origami. And finally I will explain why these methods are not desirable for our problems.

2.3.2.1 Global motion planning for articulated bodies

LaValle and Kuffner [2001] suggested a Rapidly-exploring Random Trees (RRTs) based approach for generic robotic system where the dimensionality of the configuration space is high and constrained regions caused by obstacles are present. Under the framework of RRTs, Bertram et al. [2006] defined heuristic functions to implicitly define goal configuration region to improve the performance of the search. It avoids

the problem of unreachable goals. Ho and Komura [2007] raised the motion planning problems for two humanoid robots to make postures where occlusions and tangles are present. They describe the tangles and occlusions by Gauss Linking Integral (GLI) which quantitatively measures how two links are tangled. Based on this, they employed RRTs to sample the configuration space.

2.3.2.2 Knotting and unknotting for deformable linear objects

More recently, people tried to find more complicated maneuvers to accomplish high-level tasks with objects of many DoFs, such as linear deformable objects. In such research, the highlight is the state abstraction and transitions between states. Depending on the goals, the state of the object along with path-planning algorithms varies from one to another. For linear deformable objects, many researchers have tried to propose robust control strategies for knotting. Henrich et al. [1999] and Acker et al. [2005] used contact information to classify the states of a linear object to describe assembly/dissembly tasks. Phillips et al. [2002] simulated the process of tying a knot given a loosed knotted rope as an initial state. Visual feedback information was incorporated by Matsuno et al. [2001] to tie a rope with a cylinder. However, in these studies, human input is vital because the motion planning given in advance was devised with human demonstrations. Another way to accomplish a knotting task is to give the user complete control in a virtual environment. Brown et al. [2004a] designed graphical interface for a rope simulator so that the control is totally delegated to the user. The highlight of this work is to build up a fast and robust rope simulator so that it works at an interactive rate.

In order to find a more systematic approach, Wakamatsu et al. [2005] decomposed a knotting/unknotting process into a series of crossing state transitions. The state is the topology of the object features by crossings. For each transition, they defined the grasping points and moving direction. Finally, basic operations for each transition were also defined. The automation of motion planning happened only at the local level. Later, Wakamatsu et al. [2006] extended the work by adding a method that automatically detected appropriate grasping points and moving directions to further automate the whole process. Although different operations were defined, there was not much research on the relationship between these operations and complex knots. Hence, following a similar decomposition of the knotting processes, Yamakawa et al. [2008] did analysis on the relationship between different manipulator skills and complicated knotting tasks. To determine necessary skills to make knots, they looked into

human demonstrations and generalized several basic operations like loop production, rope permutation, rope pulling and rope moving. Based on these elementary operations, two kinds of knots were realised. Topological state representation can also be found in other work. This work resembles ours in Chapter 4 in the sense that they also try to define basic modularised operations for knotting. However, we focus more on the quality of the motion for animation purpose and we also extend our control to strand-object scenarios. Saha and Isto [2007] also employed a topology based representation. The state was also featured by a sequence of signed crossings. In addition, they built up a topologically biased Probabilistic Roadmap (PRM) to do the motion planning. During the construction of the roadmap, they used Inverse Kinematics to update the state of robot configuration to test the feasibility. Moll and Kavraki [2006] proposed a path-planning algorithm by finding stable configurations between start and goal configurations. The intermediate states can be found based on the fact that they are minimal-energy curves. By restricting the path-planning to the minimal-energy curves, a planner was built to move the rope from one stable configuration to another. It can also be used as a local planner of a sampling-based PRM and makes it possible to compute the roadmap for the entire "shape space".

Although their problem resembles ours, these global planning methods are not desirable for our problem. When manipulating linear deformable objects, topology based states are employed and operations for transitions are defined. However, they either control one end point while fixing the other, or define grasping points for making crossings. All these require global planning. When the Degrees of Freedom are high, the performance of this kind of methods drops significantly. On the other hand, since most operations are pre-defined in terms of accomplishing atomic tasks, it is not easy to ensure the motion quality for animation purpose.

2.3.2.3 Robotic origami

Origami is a famous form of art creation out of paper sculpturing. It is interesting from the robotics perspective because of the range of the spectrum of problems that it poses, from the design of manipulation primitives, configuration-space analysis to the mathematical modeling of folding. The control problem in origami also resembles our problem in terms of actively controlling highly deformable objects. Given arbitrary crease patterns, through operations like folding, unfolding and flapping, a piece of paper can be folded into complex configurations (shown in Fig 2.12).

Origami has been extensively researched in computational geometry. A compre-

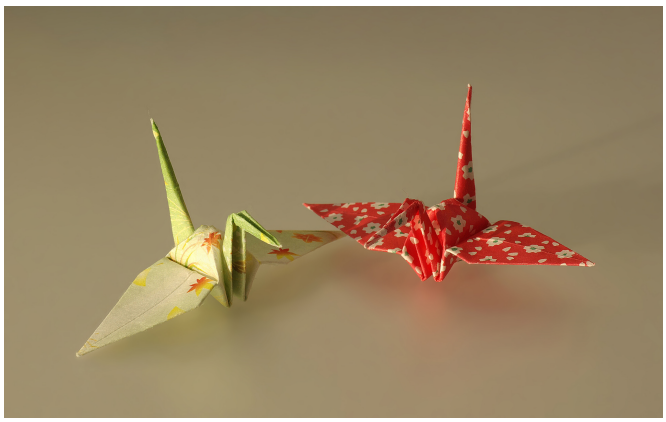

Figure 2.12: Origami Crane, courtesy of Laitche

hensive introduction, survey and articulation can be found in Demaine and O'Rourke [2008]. Mathematically, any arbitrary state can be achieved (Demaine and O'Rourke [2008]), and the creases do not have to be straight either (Demaine et al. [2009], Erik D. Demaine et al. [2011]). When it comes down to realising automated and programmable mechanisms, the current progress is good but still barely compete with humans (Fig 2.13). An interesting realisation combining origami techniques and robotics is Hawkes et al. [2010]. In this paper, they presented a sheet robot composed of interconnected triangular sections. Given desired configurations, they also proposed an end-to-end planning and fabrication process so that an optimal solution could be computed for the sheet and associated multiple controllers. However, since the self-configuration algorithm assumed connections can be dynamically built and destroyed between two units, when folding, establishing and breaking connections between neighbours happened very often. Later, An et al. [2011] improved this by considering fixed hinges. A continuous folding motion was planned so that the sheet still folded into pre-determined shapes.

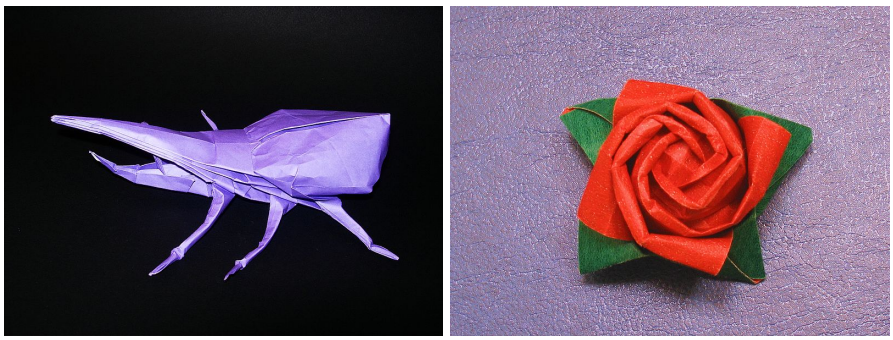

Figure 2.13: Origami examples, the state of art of human origami, by Hola soy Archivaldo

In robotic origami folding, the target object to manipulate is called *developable* surface which bends but does not stretch, such as a piece of paper. Balkcom and Mason [2008] gave a nice chart to intuitively describe the state of art. (Figure 2.14)

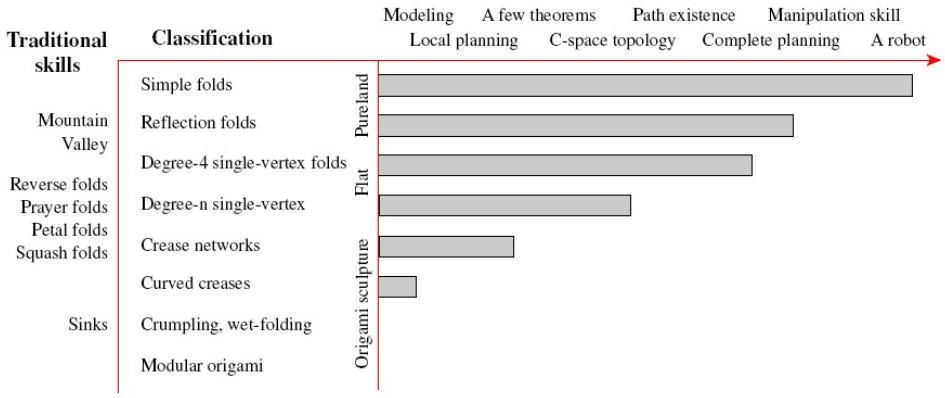

Figure 2.14: State of art skills, by Balkcom and Mason [2008]

A primitive form of robotic origami is box folding. Lu and Akella [2000] proposed an algorithm to fold a carton blank into a carton. A carton blank is essentially a piece of cardboard with pre-defined creases dividing the whole piece into smaller units that are panels of the carton. In their work, it started as a flat cardboard, then continuously evolved into a carton. In order to ensure the continuity of the motion, correlations between panels and folding operations were also considered. Pre-defined creases are regarded as one Dof joint and panels as rigid parts. Self-collisions and the desired configuration of the carton were formulated as constraints. By representing the configuration space as a recursive tree, the method enumerated all possible collision-free ways to fold the carton blank. Song and Amato [2004] solved a similar problem but their method can work in high-dimensional configuration spaces. They designed a Probabilistic Roadmap planner to explore the space. The application spans simple folding problems like multi-link object to protein folding with a large number of degrees of freedom. Gupta [1998] proposed a system for bending sheet metal. They had a two-level path planning algorithm. Low level planners defined a number of primitive operations while the high level planner determined the sequence of bending operations. The planner relied on kinematic models of the sheet metal and all bending operations are atomic. Much more recently, there is another work that catches attention. Maitin-Shepard et al. [2010] built a robot to fold randomly placed towels with a very high successful rate. However, although folding was only simply involved by aligning two corners with the other two. The work primarily focused on visual interpretation of the

towel.

Although as successful as robotic origami methods are, they are not directly applicable for automating Furoshiki wrapping, because the motion planning is based on given crease patterns. For our problems, the crease patterns can be extremely complex while we are trying to realise the natural results of wrapping movements, not something given in advance.

2.3.3 Summary

Global planning methods in robotics comprise an indispensable part of the state of art techniques for highly deformable object controlling. However, they are not desirable for making animation because: (1) In the reviewed work, the aesthetic requirement is not very important. Although a lot of control strategies were inspired by human demonstrations, the result behaviour does not have to be human-like. (2) These methods are general but most of them depend on sampling algorithms. For complicated tasks with high DoFs, the performance is low, thus not ideal for animation design which requires iterations of interactions between the animator and the computer. (3) For robotic origami, folding problems are solved with the help of pre-defined crease patterns. For our problems, it is impractical to give pre-defined crease patterns. One example is the creases in Furoshiki wraps. These creases are naturally generated by physical laws and should vary based on the control strategy and the geometry of the target object. Even if we manage to give pre-defined crease patterns, then every time we change the object, we have to change the pattern.

2.4 Physically based deformable models

Since part of the work is to control highly deformable objects such as ropes and cloth, we need a robust and efficient simulator. The simulator should be: (1) fast enough for objects with a large number of Degrees of Freedom. (2) Steady enough because the tasks can be complicated which requires small steps and long-term simulation. (3) Well designed for control purpose so that control signals can be easily incorporated into the framework.

Two prevalent models are continuum models and mass-spring models. After comparison, we chose the latter one. I will go through these models and explain the reason why we choose the mass-spring model as our simulator. Aside from the models, the

time integration is another issue. We chose an *explicit* Euler stepping scheme. I will also give the reason of our choice in this section.

2.4.1 Continuum models

Continuum elasticity modelling is one major approach for deformable object simulation. It assumes a deformable object occupies a continuous volume in 3D Euclidean space. In its rest pose, $q_{rest} \in R^3$, the volume it takes is a subset, M, of R^3. The coordinates of a point in the object, $q_m \in M$, are the *material coordinates* of that point. The discretisation of M is a point cloud that samples the rest shape of the object, $M(q_{rest})$.

The equilibrium state of the object under external impact can be described by:

$$\Pi = \Lambda - \Omega \tag{2.6}$$

where Π is the total potential energy of the system. Λ is the total strain energy and Ω is the work done by external forces. Some external forces act on points of the object, some forces like gravity has a distributive impact, and others only influence the object over the surface such as pressure.

Generally, solving this equation involves dealing with a Partial Differential Equation (PDE) of an infinite number of variables. Methods like Finite Element Method (FEM) can be used for discretising the system. They enable us to formulate a linear system for the equilibrium on sampled points. FEM is a well-researched topic. We only introduce the general concept here and different formulations later in this section. Details can be found in any recipe books such as Bathe [2007].

Overall, the object is discretised into basic primitives, called element, with nodes. Any physical property inside the element can be approximated as a finite sum of products of interpolation functions with their corresponding nodal values. For example, if the governing function of an element is a scalar function, F(q), then the value at a sample point q inside the element can be approximated as:

$$F(q) \approx \sum_i h_i(q_i) n_i \tag{2.7}$$

where $h_i(q)$ are the interpolation functions defined at the element containing q_i, and n_i is the value at node points of the element. Solving the system is equivalent to finding the node values n_i that minimise the total potential energy of the object.

Equation 2.7 is the general idea how we can convert the problem with infinite number of DoFs by an approximation with finite DoFs. After choosing elements for discretisation, *displacement* vector field can be used to derive the tensor.

The deformation of the object gives a *displacement* vector field:

$$\mathbf{D}(\mathbf{q}) = \mathbf{M}'(\mathbf{q}) - \mathbf{M}(\mathbf{q}) \tag{2.8}$$

where $\mathbf{M}(\mathbf{q})$ is the material coordinates of the object and $\mathbf{M}'(\mathbf{q})$ is the new coordinates under some deformation. If $\mathbf{D}(\mathbf{q})$ is constant everywhere, then the displacement is a translation of the object and there is no strain. So it is easy to see that the strain depends on the spatial variations of $\mathbf{D}(\mathbf{q})$. Popular ways of calculating the tensor from displacement field $\mathbf{D}$ are (Nealen et al. [2006]):

$$\varepsilon_G = \frac{1}{2}(\nabla D + [\nabla D]^T + [\nabla D]^T \nabla D) \tag{2.9}$$

$$\varepsilon_C = \frac{1}{2}(\nabla D + [\nabla D]^T) \tag{2.10}$$

where $\varepsilon_G \in R^{3\times 3}$ and $\varepsilon_C \in R^3$ are Green's nonlinear strain tensor and Cauchy's linear strain tensor which is the linearisation of ε_G.

In order to compute the strain energy of the object, one need to compute stress, σ, and strain, ε, tensors. Both of them are 3×3 matrices. The strain energy can be calculated by:

$$\Lambda = \int_V \sigma \cdot \varepsilon \, dv \tag{2.11}$$

where v is unit volume of the object. So far, given the displacement vector field, both strain and stress tensors can be calculated, then the strain energy, Λ, in Equation 2.6 is computed. There is another term on the right hand side of Equation 2.6, the work done by external force, Ω. It is computed by:

$$\Omega = \int_V D \cdot f \, dv \tag{2.12}$$

where f is the external force and $\mathbf{D}$ is the displacement vector field. Like mentioned before, external forces applied to the whole volume like gravity can be integrated over the whole volume and forces only applied on the surface like pressure will only accumulated over the surface of the volume.

When doing the simulation, deformable objects are embedded in a cell-decomposed 3D space. The governing equation of motions is defined as:

$$\rho \ddot{q} = \bigtriangledown \cdot \sigma + f \tag{2.13}$$

where ρ is the density and $\ddot{q}$ is the acceleration. $\bigtriangledown \cdot \sigma$ is the divergence of the tensor and f is the external force.

As mentioned in equation 2.7, n_i is the value of node i and is what is to be solved for. The interpolation functions are ideally Kronecker Delta functions where they get value 1 and their respective nodal point and zero anywhere else. Substituting equation 2.7 into equation 2.13 gives a linear system with unknowns n_i.

Solving for n_i was treated as an optimisation process by the Galerkin approach (Hunter [2005]). Many researchers (Debunne et al. [2001] Müller et al. [2002]) choose a simple FEM for simulation which is called *explicit* FEM where masses and forces are lumped to vertices. For each element, the displacement field **D** can be computed by equation 2.7 given positions of vertices of the element and basis functions. Then strain field, $\sigma(m)$, and stress field, $\varepsilon(m)$ can be computed. Finally, the strain energy can be calculated by Equation 2.11. Internal forces can be derived from the energy. And the equilibrium of the object under a deformation D_d is governed by Lagrange Mechanics:

$$M\ddot{D}_d + C\dot{D}_d + KD_d = f_{ext} \tag{2.14}$$

where **M** is the mass matrix, **C** is the damping matrix, **K** is the stiffness matrix and f_{ext} is the external force. This formulation is a bit different from Equation 2.13 due to the different discretisation of the object.

So far, we use linear algebraic equations to solve a linear PDE. If we use a linear strain and assume isotropicity of the material, Hook's law can be substituted into Equation 2.13 to get Lamé's linear PDE:

$$\rho\ddot{q} = \mu\Delta D + (\lambda+\mu)\bigtriangledown(\bigtriangledown\cdot D) \tag{2.15}$$

where Δ is the Laplacian of the mesh and $\bigtriangledown(\bigtriangledown\cdot)$ is the gradient of the divergence. λ and μ are coefficients. This formulation was used in Debunne et al. [1999]. It was broken down and solved in a multi-resolution discretisation scheme. Later, the operators were evolved in Debunne et al. [2000] based on the Gauss's Divergence Theorem. However, both of them used a linearised strain tensor. This was replaced by explicit finite elements and Green's nonlinear tensor by Debunne et al. [2001].

In practice, the general finite element method has different variations. If the object can be discretised by regular grids, then the Finite Difference (FD) can be used to solve Equation 2.13. If all computations can be carried out on the surface instead of the interior of the object, then a Boundary Element Method (BEM) is a good choice. We refer them as FEM-based models in this thesis. FEM-based models have been used to simulate various objects. Müller et al. [2001] and O'Brien et al. [2002] developed

techniques for simulating brittle and fragile materials. Picinbono et al. [2000, 2001] also use such models to do simulations for medical research.

Related to this thesis, FEM-based models are used for rod simulation. In mechanical engineering, Klapper [1996] discretised elastic rods in regular grids so that finite differences can be used to describe the energy. Yang et al. [1993] and Goyal et al. [2007] used finite element for the purpose of discretisation. The methodology has also been applied to cloth simulation. As early as the late 1980s, Terzopoulos and Witkin [1988]; Terzopoulos et al. [1987] proposed a general method for simulating various elastic objects including cloth. Terzopoulos et al. [1987] derived their models from elasticity theory and constructed differential equations for non-rigid curves, surfaces and solids. In Terzopoulos and Witkin [1988], they decomposed the formulation into a rigid reference term and a deformation term to solve the numerical problem existing in their previous work when the rigidity of the models increased. Later, Eischen et al. [1996] used methods from nonlinear shell theory to simulate 3D motions related to fabrics.

Comparing with Mass-spring model, FEMs provide more physical fidelity with fewer nodes, hence requiring solving a smaller linear system. However, any force must be converted into its vector form involving integrating forces over a volume for each step. On the other hand, since the mass and stiffness matrices are derived from a limited number of elements distributed in the volume, any topological or large geometrical changes lead to a re-evaluation of these matrices which could cause significant amount of pre-processing work. This fact makes it less favourable for our control problem of ropes and cloth because severe deformation is expected for tasks such as knotting and wrapping. It means more re-evaluations will be required very often and it will drag down the performance of the system. Meanwhile, although the continuum model provides more physical fidelity, this is something we can compromise to some extent. Next, I am going to review another model that we choose as our simulator for experiments involving ropes and cloth.

2.4.2 Mass-spring models

Another way to model a continuous volume is to see them as lumped masses connected with springs (Figure 2.15)

The states of the system can be derived by stacking all the states of individual particles. Given a particle with its position p_i and velocity v_i, the external forces are elastic

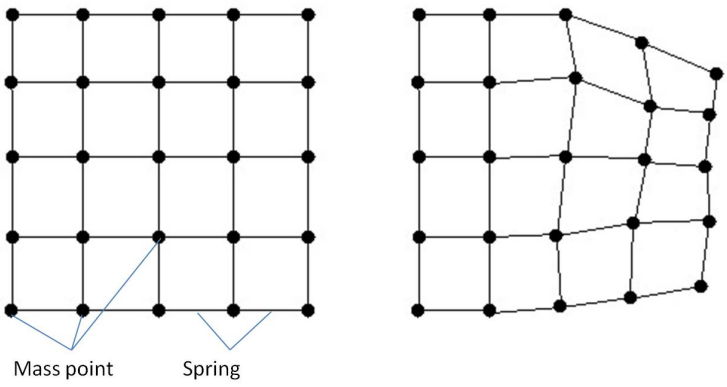

Figure 2.15: Left: A undeformed sheet represented by masses and springs. Right: A deformed one

forces by springs connected to it, and other forces like gravity and contact forces. The governing equation is Newton's second law:

$$m_i \ddot{p}_i = f(p_i, v_i) \tag{2.16}$$

where m_i is the lumped mass of the particle. Stacking all the particles:

$$M\ddot{p} = F(p, v) \tag{2.17}$$

where $\mathbf{M}$ is a $3n \times 3n$ diagonal mass matrix, n is the number of particles. Solving this large linear system gives the solution of the particle system. Springs are usually modelled as being linearly elastic:

$$f_i = k_s(dis(q_i, q_j) - l_{ij})dir(q_i, q_j) \tag{2.18}$$

where f_i is the elastic force between particle i and particle j. k_s is the stiffness of the spring. $dis(q_i, q_j)$ is the current distance between particle i and particle j, and l_{ij} is the rest length of these two particles. Finally $dir(q_i, q_j)$ is a unit vector pointing q_j from q_i. The formula obeys the conservation of energy perfectly which is rarely seen in reality due to the energy dissipation. So a damping term is applied (Nealen et al. [2006]):

$$f_i = k_d(\frac{v_{ij}^T q_{ij}}{q_{ij}^T q_{ij}})q_{ij} \tag{2.19}$$

This damping scheme makes sure that the wrinkles and rotations are not damped, which is very crucial to deformable object simulation. It only damps the part of the velocity difference that is projected along the vector $q_{ij} = q_j - q_i$.

Aside from searching for a general framework for cloth simulation, many researchers also focus on special behaviors of cloth. Bridson et al. [2003] separated the bending behavior from all other deformations. They derived a physically correct bending model so that non-zero rest angles can be used for cloth simulation to keep wrinkles and folds. A similar effort has been done by Wang et al. [2010a]. They used a semi-physical scheme where they predict some properties (like orientations) of wrinkles, then combine a low-resolution physical simulator with a high-resolution geometry database to provide visually convincing results for real-time applications. Goldenthal et al. [2007] suggested a velocity filter which projects unconstrained simulation back onto the manifold of stretchiness constraints, so that desired stretchiness limit can be kept while not compromising the performance.

Generally speaking, mass-spring systems are less accurate and usually do not converge (Nealen et al. [2006]). The result depends on the resolution and topology of the discretisation. Moreover, the coefficients are typically chosen arbitrarily which makes it harder to precisely model the cloth. To increase the accuracy, Kass [1995] proposed a one dimensional equivalence between a mass-spring system and a finite difference spatial discretisation. However, this kind of discretisation does not generalise to triangular meshes. Baraff and Witkin [1998] provided a cloth model for triangulated meshes and it also supports anisotropic behavior. But the convergence is not guaranteed. Teschner et al. [2004] established a framework to simulate tetrahedral and deformable triangular meshes. Their model is convergent and efficient.

Comparing to the continuum model, the mass-spring model is ideal for our case. It is easy to implement. Unlike the continuum model where the control force must be integrated over volumes, control signals can be easily introduced onto lumped mass points. Under severe deformation, the performance is generally better. Although it does not guarantee to converge, it still produces visually compelling results.

2.4.3 Data driven model

Aside from the previous two models that are based on the interpretation of the physical properties of deformable objects, researchers also try to find abstract models through data. Given a mass-spring model, there are still parameters like stiffness to be decided. It is difficult to examine how closely models with arbitrarily chosen parameters mimic real fabrics. Hence learning methods have been employed on different levels to improve the simulator. Bianchi et al. [2004] used the model by Baraff and Witkin [1998]

and a video of deforming cloth to learn the parameters. Feng et al. [2010] used kernel methods to learn wrinkle patterns on different levels, so that they could transform low-resolution simulation results into high-resolution deformation with dynamically introduced details. de Aguiar et al. [2010] learnt a model out of clothes motions. Their model consists of a mapping between body motions and cloth motions, a Linear Dynamical System that took care of internal dynamics and residue dynamics terms caused by local orientation changes. The result gave a stable space where cloth motion can be easily simulated while preserving folding details. However, it is mainly for simulating relatively low-resolution cloth motions that are attached to human bodies. For different kinds of fabrics, they usually display different anisotropic elastic behaviors. Wang et al. [2011] suggested a piece-wise linear elastic model for simulating the anisotropic behaviors of different fabrics. Their model can be incorporated into existing simulators to generate realistic elastic motions. They also proposed a framework to measure different parameters without having to isolate them.

However, the data driven model is not suitable for us. Because it is very difficult to capture the wrapping and knotting motion of ropes and cloth, especially when large distortions are present which happens quite often in this kind of tasks.

2.4.4 Time integration

After choosing the model, another important issue of physical simulation is the stepping scheme. Given a dynamic system, its trajectory along time dimension, q(t), evolves as time passes. The system is governed by an equation that relates q to its first and second order derivative with respect to time:

$$\ddot{q} = F(q, \dot{q}, t) \tag{2.20}$$

where $\ddot{q}$ is the acceleration and $\dot{q}$ is the velocity. A simple scheme to evolve the system is:

$$q(t + \nabla t) = q(t) + \Delta t \dot{q}(t) \tag{2.21}$$

$$\dot{q}(t + \nabla t) = \dot{q}(t) + \Delta t \ddot{q}(t) \tag{2.22}$$

This is also called *explicit* Euler integration. It is conditionally stable due to the fact that it extrapolates into the future. In order to improve the stability, Δt must be smaller than a threshold (Müller et al. [2005]). The effect can be mitigated by swapping the order of the above two equations which then becomes a forward-backward Euler scheme.

However, to completely eliminate the instability, an *implicit* method is used: substitute $\dot{q}(t+\Delta t)$ into the formula:

$$q(t+\nabla t) = q(t) + \Delta t \dot{q}(t+\Delta t) \tag{2.23}$$

$$\dot{q}(t+\nabla t) = \dot{q}(t) + \Delta t \ddot{q}(t+\Delta t) \tag{2.24}$$

The implicit or backward Euler integration is unconditionally stable at the cost of having to solve a linear system at every time step.

When simulating cloth, the governing Equation 2.17 is usually stiff (Kass [1995]). So tiny steps must be taken and it is unbearable for long-time simulation. Baraff and Witkin [1998] proposed an implicit scheme that allows large time steps to be used without numerical explosions. It has been proven to be efficient and robust and widely used in the field. Based on this work, people have been trying to improve the stability and efficiency (Desbrun et al. [1999] Volino and Thalmann [2000] Choi and Ko [2002] Bridson et al. [2003] Boxerman and Ascher [2004])

All the research mentioned above is *synchronous* stepping scheme. All particles are advanced all together after one time step. Synchrony has its innate shortcoming: the balance among safety, correctness and progress has to be carefully tuned (Harmon et al. [2009]). Its inability of exactly computing the sequence and timing of all impacts was discussed in Cirak and West [2005]. So the idea of *asynchronous variational integrators* (AVIs) was employed to handle different impacts at different timings. AVIs are proven of conserving energy over long time simulation. Their stability was investigated in Fong et al. [2007]. Harmon et al. [2009] presented a framework of an AVI with *Kinetic Data Structure* (KDS) (Basch et al. [1997]) to simulate deformable objects under severe deformations with high physical fidelity.

For our experiments, since we mainly focus on the control strategy, plus in order to do fine control, we take small steps, then the *implicit* stepping is not very desirable because it requires solving a linear system every time step. For a high-resolution rope or cloth, it is slow. Thus, we choose the *explicit* scheme as our stepping method. Meanwhile, asynchronous stepping can be used to improve the result too. But we choose not to use it for our experiments because it is generally much slower than synchronous stepping schemes.

2.4.5 Summary of physically-based models

Continuum models can achieve more physical fidelity with fewer elements. However, it is difficult to use them for large deformations without severely compromising the

performance. In addition, introducing control signals is not straightforward. Hence, we choose a mass-spring model. In order to avoid solving a huge linear system for simulation for every step, we choose explicit Euler integration. However, I would like to point out that the control methods we propose do not impose constraints on the simulator. They can work with any simulators with any time integration scheme.

2.5 Summary

Controlling highly deformable objects for animation purpose impose premises that prevent us from naively using existing techniques. These special premises vary at different levels from the controllability of the details for the animation, the aesthetic requirements of the motion to the performance and responsiveness of the system. All these conditions lead us to the point of finding out new models and control strategies.

Chapter 3

Interpolation of arbitrary postures of a human body

This chapter is primarily based on one of my publications (Wang and Komura [2011]). It shows results of controlling a 3D articulated human body model by using a repulsive energy function. The idea is based on the energy-based unfolding and interpolation, which are guaranteed to produce intersection-free movements for closed 2D linkages. Here, we apply those approaches for articulated characters in 3D space. We present the results of two experiments. In the initial experiment, starting from a posture that the body limbs are tangled with each other, the body is controlled to unfold tangles and straighten the limbs by moving the body in the gradient direction of an energy function based on the distance between two arbitrary linkages. In the second experiment, two different postures of limbs being tangled are interpolated by guiding the body using the energy function. We show that intersection free movements can be synthesised even when starting from complex postures that the limbs are intertwined with each other. At the end, we discuss about the limitations of the method and future possibilities of this approach.

3.1 Introduction

For human character animation, the relationship between body parts is important. A lot of postures are featured by tangles like Yoga exercises or martial arts. These phenomena are even more prevalent in interactions between characters and between characters and the environment. The motion planning under such circumstances are really hard due to occlusions. Controlling characters that are closely interacting with their own

bodies, other characters and the environment is a difficult problem. As the joint angle representation, which is the most prevalent way to express the posture of characters, does not consider the spatial relationship of the body limbs, the resultant movements suffer from collisions and penetrations. In this chapter, I will propose a method to control characters by using a repulsive energy function that has been successfully applied to control 2D linkages for unfolding Cantarella et al. [2004] and interpolating Iben et al. [2009] different configurations. We define a similar energy function used in Cantarella et al. [2004]; Iben et al. [2009] that is based on the Euclidian distance of the linkages of character skeletons in 3 dimensional space.

We show experiments that are extensions of Cantarella et al. [2004]; Iben et al. [2009] with additional constraints for generating human character animation. Two main experimental results will be shown below: one is convexification of human postures which resembles the unfolding problem defined in Cantarella et al. [2004]; the other one is a posture interpolation algorithm based on the 2D polygon interpolation scheme introduced in Iben et al. [2009].

The structure of the rest of the chapter is as follow. First I will review related work in different fields that are closely related to the problem we try to solve. Then I will present the energy function that plays the central role in this algorithm which leads to a straightforward algorithm of posture convexification. Finally, based on the convexification algorithm, an interpolation scheme is derived.

3.2 Contribution

1. We propose a repulsive energy function to model the potential collisions for character control.

2. We propose a fast interpolation algorithm on the manifold described by the energy function for interpolating arbitrary postures. The generated motion is guaranteed to be penetration-free.

3.3 Overview

For arbitrary posture interpolation, under the research diagram suggested in Figure 1.1, we suggest a framework where the object-centric manifold and the task-centric manifold are overlapped where the parameterisation of the object (the human body) is

consistent. The only additional parameter for the task-centric manifold is an energy value attached to every configuration. It is computed by an energy function we define to describe the task as: change the body configuration so that the energy gets lower and lower. This is a reinterpretation of the interpolation task based on the assumption: every posture can be guided into a low energy region and the interpolation in the low energy region can easily be penetration free.

3.4 Posture Descriptor-a Repulsive Energy Function

A posture is usually represented by joint angles. Since this representation does not capture the information about potential collisions between different parts of the body, we add another energy function to trace the potential collision information. Similar to the linkage unfolding problem in 2D, collisions and penetrations are concerns for human motions. So the energy function is a repulsive energy function which increases drastically when any proximity of two body parts happens.

Not any arbitrary repulsive energy function can be used in such cases. Four properties should be satisfied (Iben et al. [2009]) by the function. They are: (1) the energy is infinite when the linkage crosses itself; (2) repulsive motions decrease the energy; (3) the energy should be able to be divided into atomic terms, and can be calculated independently between paired elements; (4) the function should be at least C^1 continuous with bounded curvatures. These four properties are crucial for an expansive motion to always exist to convexify a linkage in 2D. The reader is referred to the paper (Iben et al. [2009]) for details.

For character animation in 3D, we cannot assume that the skeleton is infinitely slim as is the case for linkages in 2D. So we inflate the bone segments into capsules overlapped at joints as bounding volumes. Then we simply define the repulsive energy function as below:

$$G = \sum_{i,j,i\neq j} E(q_i, q_j) = \frac{1}{D(q_i, q_j) - (R_i + R_j)} \tag{3.1}$$

where q_i, q_j are the configurations of the two body segments, and $D(q_i, q_j)$ is the shortest distance between the two non-adjacent body segments, so distances between body segments connected by one common joint are ignored. R_i and R_j are the radii of the bounding capsules of link i and link j of the body respectively. We choose bounding capsules for following reasons: (1) the computation of the distance between two

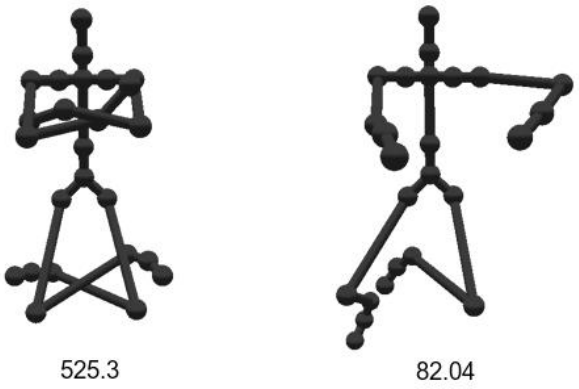

Figure 3.1: A folded posture (left) and an unfolded posture (right), and their repulsive energies shown at the bottom of each posture.

capsules is cheap; (2) they nicely form up a bounding geometry so it is easier to fit a character with arbitrary geometry into a capsule-based character model after the motion is planned. At any moment when there is no penetration, $D(q_i, q_j) - (R_i + R_j)$ will always be larger than zero. When any two body segments get in contact, it becomes zero and negative when penetrations happen. We deem whenever it becomes non-positive a failure of the interpolation.

It is easy to see that this function satisfies all four properties mentioned above. We use this function to guide the character to either unfold from a folded posture or interpolate two different postures. The energy becomes larger when the body is folded and will be smaller when all the limbs are stretched out, as shown in Figure 3.1.

3.5 Convexify Folded Postures

We use the energy function defined in Equation 3.1 to unfold the postures. In Cantarella et al. [2004], the author chose an angle-arc based representation, but special care must be taken when coming to closed chains. Hence, in Iben et al. [2009], the author chose a positional-based representation and used bone-length constraints to keep the lengths of all linkages, so that he did not have to handle the special cases separately. For us, there is no close loop in the kinematic structure of human body. So we choose joint angles over joint positions since it is easier to incorporate joint limits and other constraints into the system.

Starting from various folded postures in which the limbs are tangled with one an-

other, the Jacobian of the repulsive energy is computed, and the body posture is updated by moving the joint angles in the negative gradient direction. However, when other constraints are introduced, the final update might increase energy. Whenever we find it increasing the energy, it is updated by the following equation:

$$\Delta q = (-I + JJ^T)\frac{J}{\|J\|} \tag{3.2}$$

where q is the vector of joint angles and J is the Jacobian of the repulsive energy G: $J = \frac{\partial G}{\partial q}$. After solving Δq, it can be normalized and used to update q: $q := q + \Delta q$. However, we also need to consider other constraints, such as positional constraints and joint limits. Here we linearise the nonlinear constraints and turn inequality constraints into equality constraints by imposing them only when the original inequality constraints are violated (Ho et al. [2010]).

Let us represent all the linearised constraints by

$$K\Delta q = C. \tag{3.3}$$

Then, we project the original Δq onto the null space of these constraints by:

$$\Delta q' = \Delta q - K^T l \tag{3.4}$$

where

$$l = (KK^T)^{-1}(K\Delta q + \alpha\varepsilon), \tag{3.5}$$

α is constant whose value is set to 0.01 in our experiments and $\varepsilon = K\Delta q - C$.

Sometimes, after projection, $\Delta q'$ might be facing the gradient direction, i.e., $J \cdot \Delta q' > 0$. In such a case, we need to update $\Delta q'$ by a further projection:

$$\Delta q'' = \Delta q' - D^T l' \tag{3.6}$$

where

$$l' = (DD^T)^{-1}(D\Delta q' + \alpha\gamma) \tag{3.7}$$

$$D = \begin{bmatrix} K \\ G \end{bmatrix}, \gamma = \begin{bmatrix} \varepsilon \\ \sigma/\alpha \end{bmatrix}, \tag{3.8}$$

and σ is a small negative value which is set to -0.01 in our experiments. In every step, by initializing Δq to the normalised negative gradient of the energy function, the

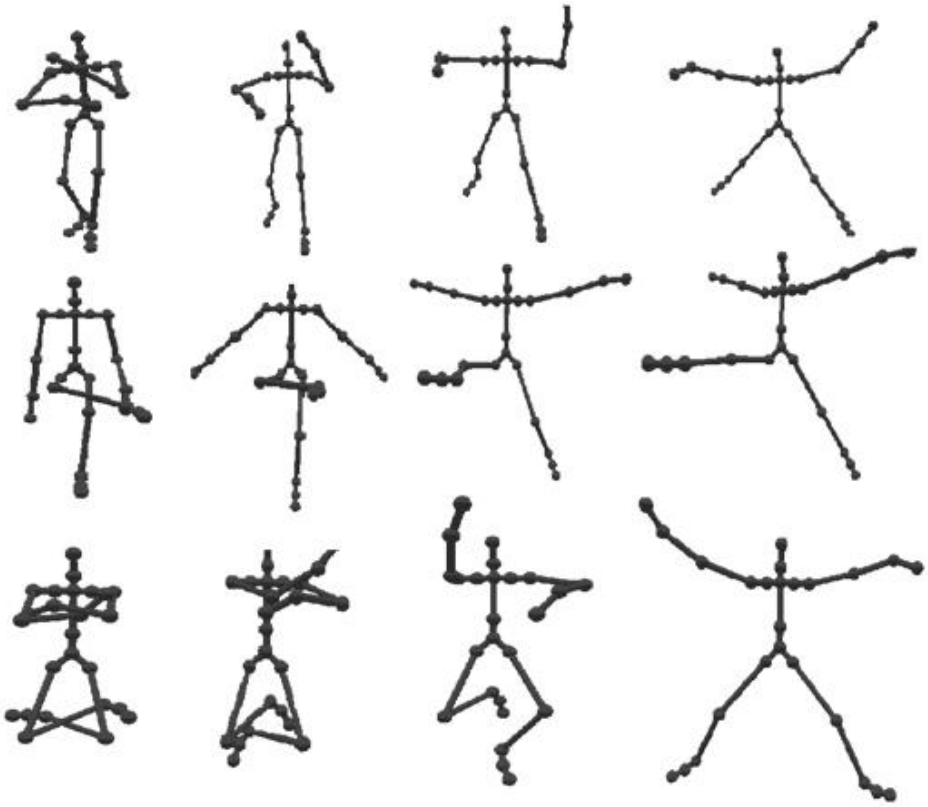

Figure 3.2: Examples of unfolding movements starting from postures in the left.

projecting it with respect to constraints by Equation 3.6, we can convexify different postures.

Starting from various configurations shown in the left-most column in Figure 3.2, the body can be unfolded into postures shown in the right-most column. It can be observed that the body can successfully unfold the limbs in these examples.

Although the unfolding is known to always converge for 2D linkages Cantarella et al. [2004], this is not always the case in 3D. This is simply because simply moving the body in the gradient of a repulsive energy can fall into local minima especially when the limbs are making knots with each other. However, we have not faced such situations in our experiments, even for a very complex yoga pose shown in the bottom of Figure 3.2.

3.6 Interpolating Postures by the Repulsive Energy

In this section, I will present the method and the experimental results of interpolating different postures using the repulsive energy defined in Section 3.4. We are applying the algorithm of interpolating arbitrary configurations by 2D linkages Iben et al. [2009] to 3D tree structures.

3.6.1 Methodology

Assume we are interpolating two different postures represented by q_1 and q_2. The algorithm proceeds by iteratively updating the two postures such that they approach towards each other. Again we assume there are linear constraints imposed to the system represented by Eq. 3.3. In each iteration, we first examine the energy of two postures, then move the posture with the higher energy, say q_h, towards the lower one, say q_l. We first calculate the update vector: $\Delta q = q_l - q_h$. Since moving along this direction could violate some linear constraints, we project the vector to the null space of the constraints by Eq. 3.4. Because moving in this direction might increase the repulsive energy, we apply another projection by Eq. 3.6. If such a direction cannot be found (if DD^T in Eq. 3.7 is not invertible), we project the gradient-descent direction vector of the energy to the null space of the linear constraints. Once the update vector is calculated, we move q_h along this direction by a small amount.

Below is the pseudo code of the interpolation algorithm:

1. Initialization: $q_1 \leftarrow q_1^0$ and $q_2 \leftarrow q_2^0$.
2. Compute the energy G_1 and G_2 at q_1 and q_2 by Eq.3.1
3. $\Delta q \leftarrow sign(G_2 - G_1)(q_1 - q_2)$
4. if$(G_1 > G_2)$ $\quad q \leftarrow q_1$ else $\quad q \leftarrow q_2$
5. Compute the Jacobian of G by partial difference : $J = \frac{\partial G}{\partial q}(q)$
6. Project Δq onto the null space of K by Eq. 3.4
7. if $(\Delta q \cdot J > 0)$
   if (DD^T in Eq.3.7 is invertible) update Δq by Eq. 3.6
   else $\Delta q \leftarrow -J$, project Δq to the null space of K by Eq. 3.4
8. $q \leftarrow q + \beta \frac{\Delta q}{\|\Delta q\|}$ where β is a small constant.

9. if $(G_1 > G_2)$ $q_1 \leftarrow q$ else $q_2 \leftarrow q$
10. If q_1 and q_2 are close enough, terminate.
11. Go to step 2.

Figure 3.3 shows how the interpolation schemes works. Once the algorithm is converged, a collision free path can be synthesized by connecting the trajectories of q_1 and q_2.

3.6.2 Experimental Results

Here we show results of interpolating two different postures explained in the previous subsection.

Waving arms. In this experiment, the two given postures are the leftmost and rightmost postures in Figure 3.4. The algorithm automatically plans the motion for the arms so that they slide over the sides of the body then reach their target configurations.

Warm-up motion. In this experiment, we try to generate warm-up motions based on two given postures shown by the first and last postures in Figure 3.5. The structure made by the hands and the leg is a knot. The knot is naturally unknotted and re-knotted.

Yoga exercise. We interpolate two complicated yoga postures (Figure 3.6). There are two knots made by the arms and legs. The algorithm successfully unknots them and re-knots them in the reversed direction.

Contortion. Contortion is an unusual form of physical display which involves the dramatic bending and flexing of the human body. In this experiment, we assume the character first lies on the ground facing down, and finally folds himself into a very distorted posture, see Figure 3.7.

It can be observed that the postures can be interpolated without any collisions or penetrations. In our experiments, all different combinations of interpolations succeeded without getting stuck at local minima. It is to be noted that the interpolation of these postures using joint angles or joint positions can easily result in penetrations of the limbs in these examples.

However, there are undesirable effects generated by this algorithm, as shown in the leg crossing experiment.

Leg crossing. Although the postures are successfully interpolated, the unnecessary expansive motions of the arms make the animation unnatural (Figure 3.8). This is because when interpolating, some body segments might tend to get closer. However, in

order to decrease the energy, when legs get a bit closer, other parts have to expand to counter-balance the increment of the energy caused by the proximity of the legs. This is an innate feature of this algorithm. To solve this problem, several methods are tried. We tried grouping body segments into groups and manipulating their respective energies and the cross energies. It did work because when dealing with two kinds of energy, the monotonicity of the energy function is lost thus the problem cannot be handled by local planning methods due to local minima. A static weighting scheme on different body parts did not work either because the weights on different body segments do not reflect the constantly changing relationships between different body parts. Actually any method that breaks the integrity of the energy function fails to work. The energy for the whole body is the key indicator of the posture and the interpolation scheme is tightly coupled with this value. In order to keep the integrity while reducing the unnecessary expansions, we changed Equation 3.1 to:

$$G = \sum_{i,j,i\neq j} E(q_i, q_j) = \frac{1}{(D(q_i, q_j) - (R_i + R_j))^d} \tag{3.9}$$

where d is an extra parameter defined by the user.

This new formula protects the integrity of the energy function by still satisfying the four properties mentioned in Section 3.4. At the same time, we can impose different weights on to different body segments by changing d. Increasing d means the energy is more dominated by the closest pair of body segments; decreasing d is downplaying the energy differences between different pairs of segments. Although d is a constant, it is still like a dynamic weighting scheme which amplifies the importance differences of body parts according to their current contributions to the energy. So to reduce unnecessary expansions, one can simply increase d then any energy increment caused by the proximity of the closest segments cannot be counter-balanced by the expansion of other segments. Hence, it will force the closest pair to get away from each other. However, d cannot be too large because the energy can easily go beyond the numerical limit. We find it works well in all of our experiments when d varies between 2 and 8. A new motion of the leg crossing experiment can be seen in Figure 3.9.

3.7 Discussions and Conclusion

In this chapter, we have shown experimental results of unfolding and interpolating tangled body postures using a repulsive energy function. Although intersection-free

movements are assured for linkages in 2D, this is not the case for 3D skeletons due to locks and possible knotted postures. Despite the concern, valid movements can be synthesized in all of our experiments. This can be because the body limbs are only composed of three to four short linkages, which is not enough to compose complex knots. In order to unfold or interpolate configurations with complex knots, it will be necessary to analyse the postures and find out the knots, and control the limbs to dissolve them.

One problem of the energy-based approach is that even postures can be unfolded / interpolated in most of the cases, the motion synthesized is usually very dynamic and not realistic in terms of human movements. Producing natural smooth motions using the repulsive energy function can be an interesting research direction to follow. Spacetime optimisation can be a good candidate for this problem. Since a penetration-free solution is provided, all the requirements for natural motions can be incorporated into the objective function while collisions can be formed as constraints.

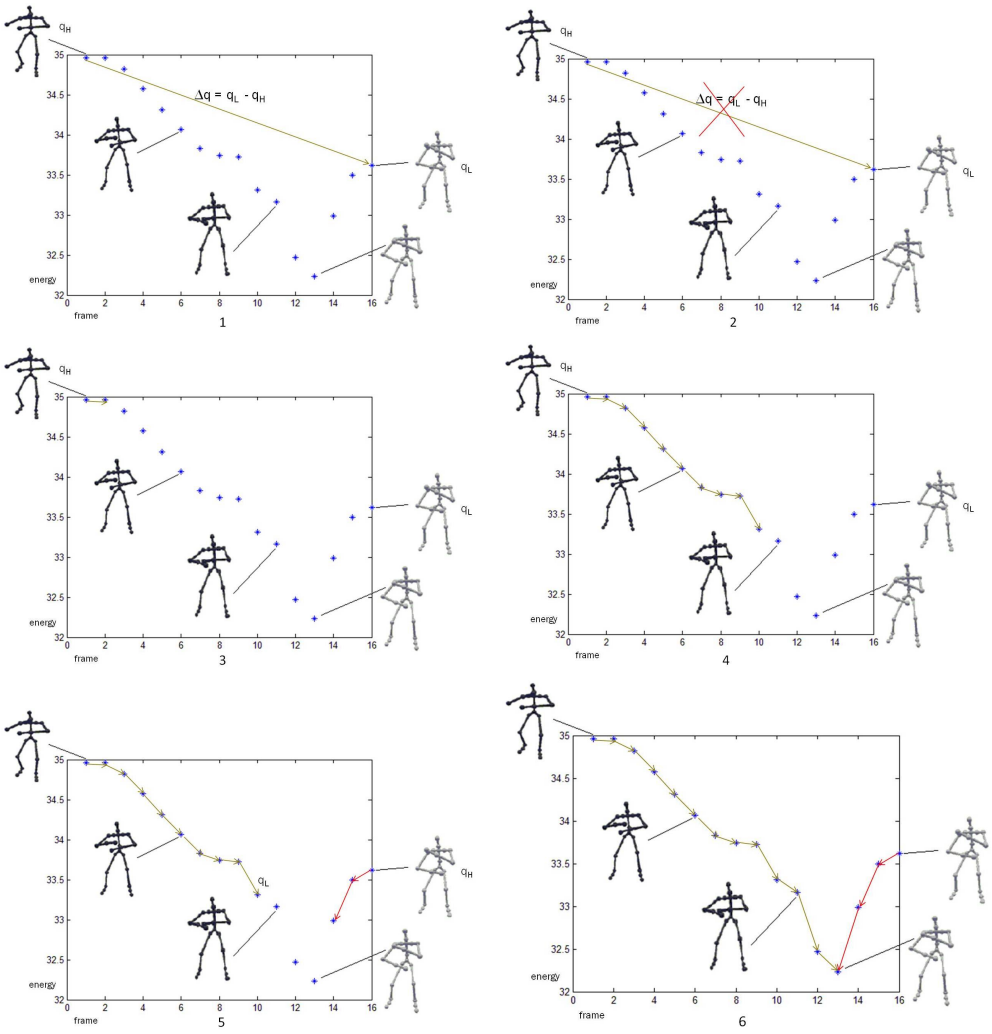

Figure 3.3: Assume two given postures are the leftmost and rightmost postures. And vertical axis is the energy and the horizontal axis is the frame number. The left posture initially has a higher energy. 1: try to move the left posture towards the right one by linear interpolation; 2 if it increases the energy then project it; 3 get a solution which reduces the energy while trying not to deviate from the linear interpolation direction too much; 4 keep going until we find the right posture has a higher energy; 5 do the same planning for the right posture; 6 do the planning alternatively for both postures until the algorithm converges.

Figure 3.4: In the first posture, the left forearm is behind the torso and the right forearm is in front of the torso while in the last posture their positions are exchanged.

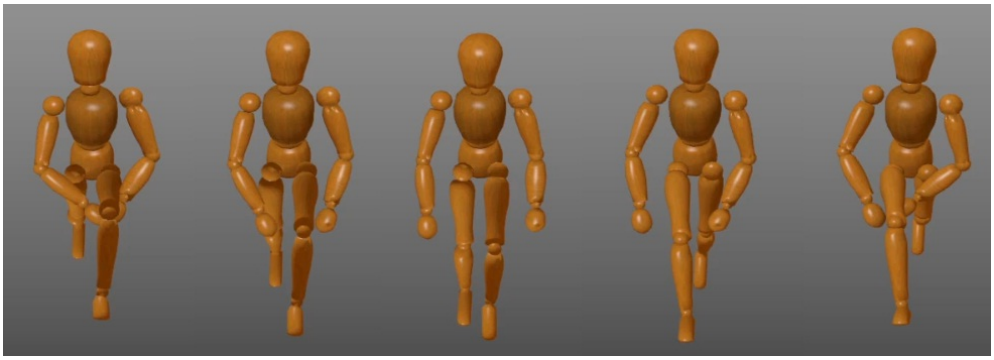

Figure 3.5: In the first posture, the hands are under the left leg while in the last one they are under the right leg.

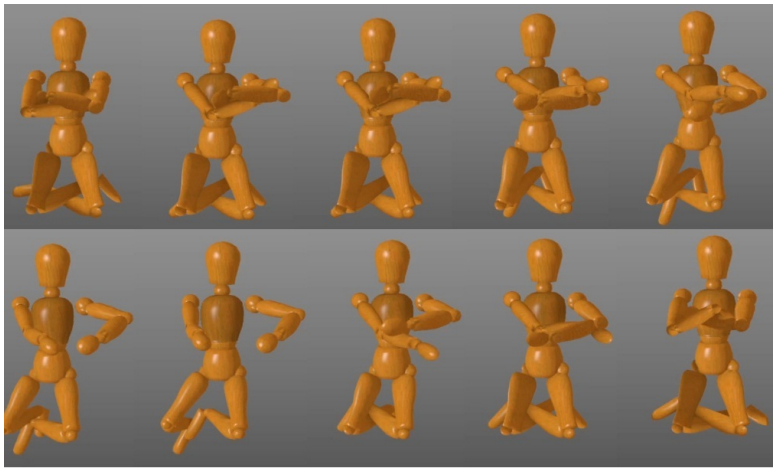

Figure 3.6: The first and last postures are yoga postures with arms and legs knotted with each other in two opposite ways.

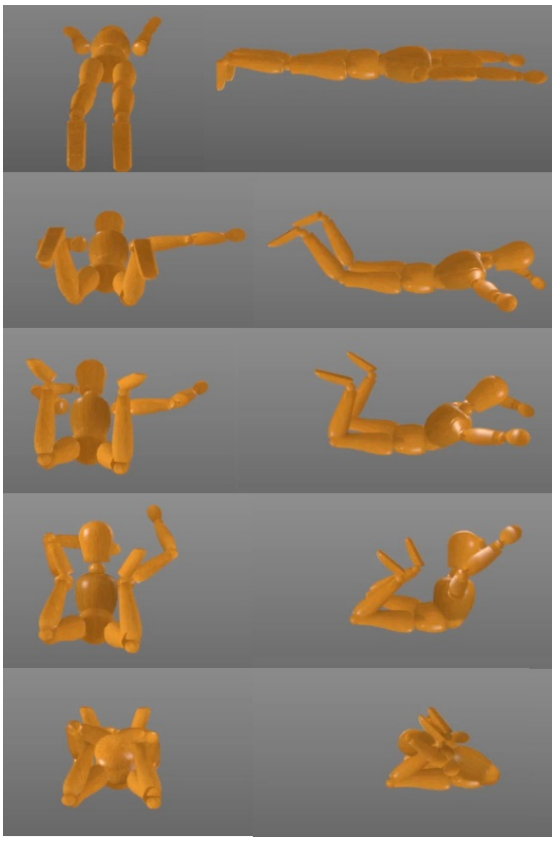

Figure 3.7: A back view (left column) and a side view (right column) of a contortion motion. The character starts with flat configuration facing down and finally folds himself.

Figure 3.8: The legs are first crossed, then untangled and crossed in the opposite way.

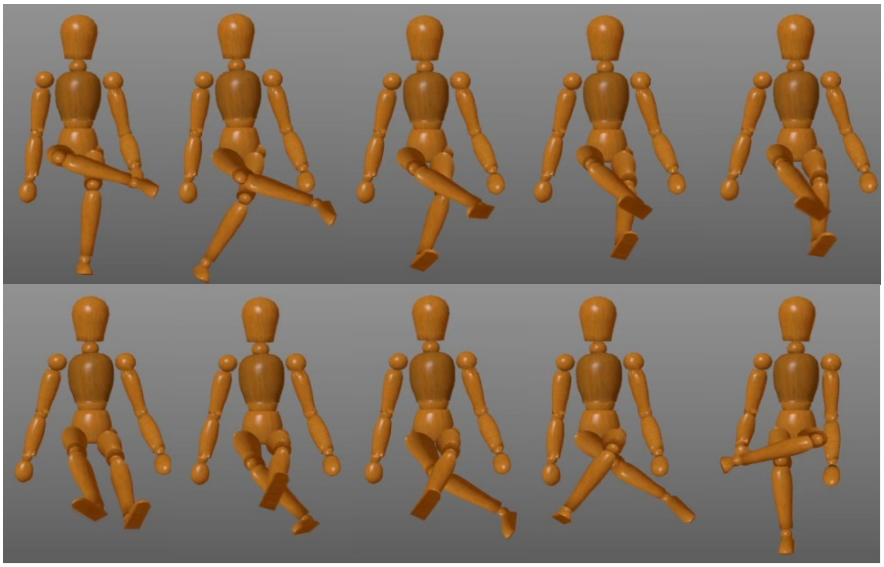

Figure 3.9: The leg crossing experiment with d = 8

Chapter 4

Manipulation of Flexible Objects by Geodesic Control

A large portion of the content of this chapter comes from my publication (Wang and Komura [2012]). We propose an effective and intuitive method for controlling flexible models such as ropes and cloth. Automating manipulation of such flexible objects is not an easy task due to the high dimensionality of the objects and the low dimensionality of the control. In order to cope with this problem, we set our goal as controlling a piece of cloth for wrapping in specific styles and introduce a method called Geodesic Control. The core idea is to decrease the degrees of freedom of the flexible object by moving it along some geodesic lines of the target object which are computed as guidance of wrapping control. By repeatedly applying this control, users can easily synthesize animations of twisting and knotting a piece of rope or wrapping a cloth around an object. We show examples of "furoshiki wrapping", in which an object is wrapped by a piece of cloth controlled by a series of manoeuvres based on Geodesic Control. As our representation can abstract such manoeuvres well, the procedure designed by a user can be re-applied for different combinations of cloth and an object. The method is applicable not only for computer animation but also for 3D computer games and virtual reality systems.

4.1 Introduction

Simulation of flexible objects such as ropes and cloth is a rapidly growing area with many applications in computer games, movies and design. However, there have not been many solutions proposed for controlling these objects during manoeuvres such as

Figure 4.1: Snapshots of conducting a basket wrapping using Geodesic Control.

winding, knotting and wrapping.

For example, a wrapping task called basket wrapping involves many steps as shown in Figure 4.1: putting the object at the centre of the cloth, bringing its two corners and knotting it under the handle of the basket, then winding the other two corners around the handle and knotting them at the top. Although humans can easily demonstrate and explain such processes to one another, automating such manoeuvres and robustly controlling the cloth to reach the target configuration is not an easy task.

The main source of the problem is the high dimensionality of the flexible object. Traditionally, rope and cloth manipulations are guided through the 3D trajectories of the end points or the corners. The rest of the object is only passively affected by the movement of the controlled area. However, the number of degrees of freedom of a piece of cloth is very large. Controlling an object with a high dimensional state space using low dimensional control signals is difficult. In addition, the trajectories of the end-points must be planned by global path-planning approaches to make sure that the flexible object does not get hooked with obstacles or with itself.

In order to cope with this problem, we propose a method to abstract the control of the flexible object. More specifically, it is a control method called Geodesic Control that can significantly decrease the degrees of freedom of the flexible object while also ensuring the resultant movement is always predictable.

This approach is based on the observation of the method that humans use to control flexible objects. For example, when winding one rope around another, generally people first cross them, and then gradually increase the contact area between the two ropes along the tangent surface of the ropes. This strategy is robust from the control point of view, as there is little redundancy at the region where the knot is produced. A similar idea can be used when wrapping a cloth around an object; the surface contact area can be gradually increased until the cloth covers the whole object. Using Geodesic

Control, a significant reduction in the complexity of the control can be achieved. At the level of winding and wrapping, global path planning is not needed as the flexible object simply moves either along the geodesic line defined on the surface of the object or that of its convex hull.

Using Geodesic Control, the process of making "furoshiki wraps", a style of wrapping and knotting cloths to carry objects, can be abstracted to a simple Finite State Machine (FSM). The same FSM can be used to wrap objects of different shapes. The wrapping manoeuvres can be synthesized in interactive time, which means the method is not only applicable for offline animation for films but also for interactive applications such as computer games and virtual reality systems.

Contributions

1. We propose a control approach called Geodesic Control, which significantly reduces the complexity and increases the robustness of winding and wrapping manoeuvres.

2. We propose to abstract complex wrapping styles by a FSM composed of a series of Geodesic Control manoeuvres. Using this FSM, we can automatically wrap objects of different geometries.

4.2 Overview

The general design of the system is shown in Figure 4.2. Our system produces animations where ropes and cloths are manipulated by following a Finite State Machine (FSM) whose states are connected by discrete manoeuvres listed in Table 4.1 and Table 4.2.

Table 4.1: The maneuvers for knotting two ropes

name	***attributes***
Crossing	sign, crossing position, control points
Winding	sign, orientation angle
Tightening	winding/knots

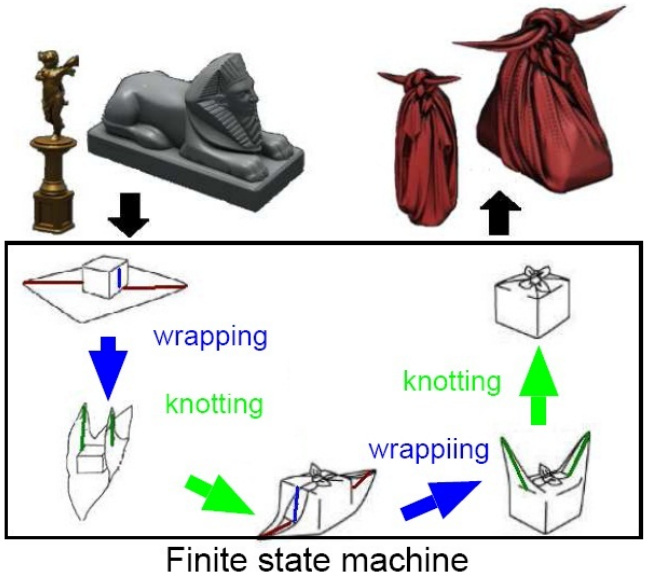

Figure 4.2: Workflow of our system. The arrows in the finite state machine represent the transitions between the states.

Referring to the framework suggested in Figure 1.1, the task-centric manifold is modelled by Finite State Machines. In each transition, the task-centric manifold, which represents tasks such as knotting and wrapping, is described by Geodesic Control. The Geodesic Control requires defining the control line and the target line, which are both geodesic lines defined either over a strand, cloth or an object. The basic idea of Geodesic Control is to gradually increase the contact area of the control line and the target line. The target lines are geodesic lines defined over the surface of the target rope for winding, and the surface of the target object for wrapping.

In 2D manifolds, geodesic lines are known as the locally shortest paths between points and are parameterized with constant velocity. They can be computed by extending a straight line on the flattened shape. They may not necessarily be the shortest curve

Table 4.2: The maneuvers for manipulating cloth

name	***attributes***
Anchoring	position of cloth and object
Wrapping	control line, geodesic line (target object)
Crossing	control lines, location
Winding	control lines (active/target), direction
Tightening	winding, direction

connecting two points, as a curve connecting two points on a sphere by the greater arc is also a geodesic line.

For knotting ropes, the user specifies the part of the rope or cloth that should be used to make the knot, and selects what kind of knots should be made (green transitions in Figure 4.2). Knotting is done by conducting a series of winding processes on the specified region of the cloth, which is guided by Geodesic Control. Although arbitrary knots can be synthesized by combining simple winding operations, we have prepared a number of template knots to ease the knotting process. The basics of knotting and how it can be guided by Geodesic Control is explained in Section 4.3.

Geodesic control can also be applied for manipulating cloth. For wrapping, the user specifies the area of the cloth and the object that is to wrap / be wrapped through our user interface (blue transitions in Figure 4.2). Wrapping is guided by gradually increasing the contact area of the control line of the cloth and the target geodesic line defined over the object or its convex hull. We can also knot two ends of cloth as we do with ropes. This process is explained in Section 4.4.

Using these techniques, examples of controlling ropes and producing wrappings using Geodesic Control are presented in Section 4.5.

4.3 Knotting Ropes by Geodesic Control

Most knots in daily life such as self-knots and granny knots can be produced by repeating the process of crossing and winding the ropes. A self-knot can be produced by crossing the strand with itself and conducting a winding (see Figure 4.3, top). For a granny knot, after one winding is done, the two ends are raised again, crossed at the middle point, and another winding is conducted (see Figure 4.3, bottom).

We automate the knotting process by dividing it into the steps listed below. Let us assume that we wind an active rope denoted by C around a target rope denoted by T. Both ropes are composed of particles connected with rigid rods of constant length.

1. C and T are moved towards each other until a cross is produced at point X.
2. A geodesic line G is computed over T's bounding tube. Starting from X, C is wound around T by gradually increasing the contact area of C and G.
3. The configurations of G and T are updated by optimization, taking into account the physical properties of the ropes, collisions and external forces.

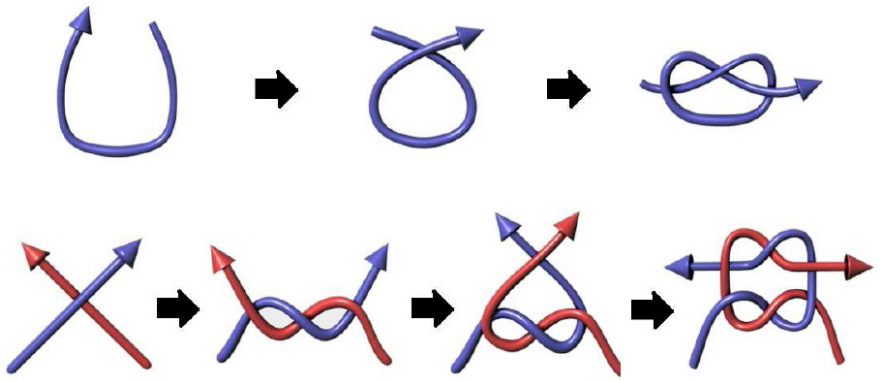

Figure 4.3: (top) The process of producing a self-knot by crossing the strand and winding around it. (bottom) The process of producing a granny knot by repeating the crossing and winding.

4. Pull the two end of the rope to tighten the winding / knot.

Step 3 is repeated until C and T are wound as desired. Knots are produced by repeating 1-4 for different windings. The individual processes are described below.

4.3.1 Crossing two ropes

The initial process of winding two ropes is to cross the two ends. Although there can be many ways to produce a cross, our strategy is to lift the two ropes C and T, set a target point X in the middle and bring the ropes towards it. We take this approach because of the following reasons: (1) we want to make a symmetric knot in the middle of the space between the two ends for aesthetic reasons, and (2) more open space will be available to manipulate the rope after lifting because gravity will pull down the rest of the rope.

Let us explain the procedure of crossing the two ropes. A coordinate system is produced as shown in Figure 4.4(a), by using the constrained bottom points of the two ropes and the vertical direction. The two ropes are crossed at a point X, whose x,z coordinates are at the middle of the two bottoms while the height (y coordinate) is set higher than the two bottoms. The two ropes are moved towards X: their z coordinates are slightly adjusted such that they do not collide with each other. A positive cross is produced in the case where a positive wind is to be made, and vice versa for a negative cross (see Figure 4.4(b), (c)).

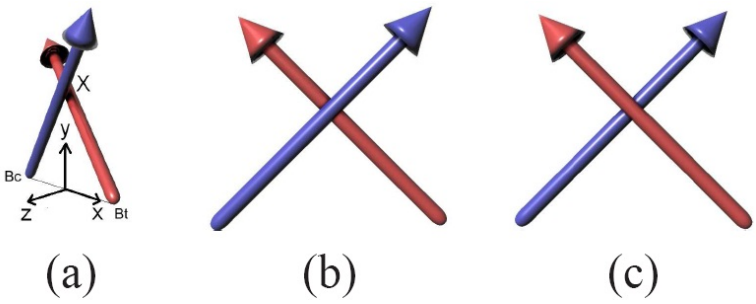

Figure 4.4: (a)The state of the two ropes when they are crossed. (b) Positive and (c) negative crosses.

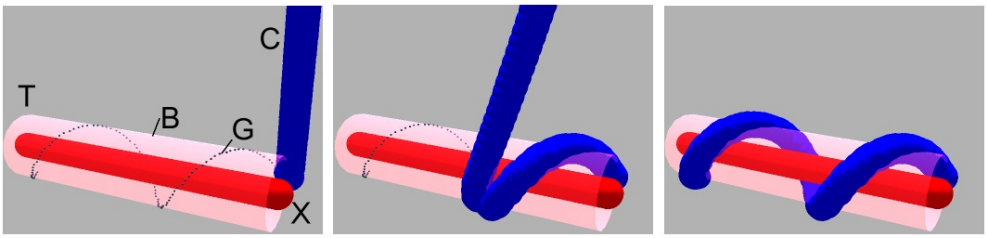

Figure 4.5: Snapshots of a blue rope wound around a red rope using Geodesic Control. An bounding tube is produced around the red tube, and a geodesic line is defined on its surface.

4.3.2 Winding two ropes

Here we explain the process of winding the rope C around the rope T. In order to ease the winding process and make the wind/knot appear horizontal and symmetric, T is straightened for a short distance along the x axis before the winding is started.

The winding starts by first computing the bounding tube of T, which is denoted by B, and then gradually attaching C onto the geodesic line G defined on B. (see Figure 4.5). The radius of B is set to $r_C + r_T + \varepsilon$, where r_C, r_T are the radius of C and T, respectively, and ε is a small value that is set to $0.1 \times (r_C + r_T)$, such that the two ropes are tightly wound with each other when the particles of T are located on B.

The geodesic line G starts from X_G, the point where the crossing point X is projected onto the bounding tube B. Let us also define the point that is closest to X on curve T as X_T. At X_G, G proceeds in the direction of $\mathbf{n}_X^C$, which is computed by orienting the direction of the rope T at X_T around $\overrightarrow{X_T X_G}$ for θ $(0 < \theta < \frac{\pi}{2})$ (see Figure 4.6, left). θ is decided according to how much the user wants to spread the winding across T: a smaller value will make it wider and a larger value will make it tighter (see Figure 4.6, right).

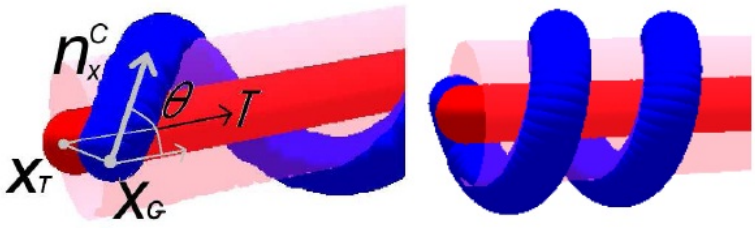

Figure 4.6: The orientation of the normal of G with respect to the negative normal of T at X_C (here denoted by θ) determines how much the winding spreads over T (left). A tighter winding produced by a larger θ (right).

G is extended along T until the winding integral (Berger and Prior [2006]) reaches the target value w_0. The winding integral can be computed by

$$\int_{x_0}^{x_1} \frac{\hat{\mathbf{x}} \cdot \mathbf{r}_{GT}(x) \times \mathbf{r}'_{GT}(x)}{|\mathbf{r}_{GT}(x)|^2} dx \tag{4.1}$$

where x is the direction of winding which is set to the direction of T at X_T, $\hat{\mathbf{x}}$ is the normal vector in the x direction, $[x_0, x_1]$ is the range of computing the winding integral along the x axis, and $\mathbf{r}_{GT}(x)$ is the vector connecting G and T in the $y-z$ plane at each x value (see Figure 4.7, left). w_0 is the number of times we want C to wind around T, which is mostly 1 for knots that we make in daily life, such as self-knots and granny knots.

The winding integral is equivalent to the Gauss Linking Integral used by Ho and Komura (Ho and Komura [2009b]) but requires only a single integration. It is only applicable when the axis of winding does not turn more than 90 degrees: if that is the case, we need to use the Gauss Linking Integral.

The winding proceeds by iteratively moving each particle p_i^C of C to its target location t_i sampled over G. Starting from X_C, t_i is sampled at every distance r, where r is the distance between every adjacent particle in C. Every time p_i^C reaches its target location t_i, the control switches to the next particle p_{i+1}^C (see Figure 4.7, right). We use a constant distance for r, although it is also possible to adaptively subsample more particles in the middle according to the existence of obstacles, and curvature of the other rope, as done in Spillmann and Teschner [2008].

This control strategy for winding is simple and stable. It requires no global path planning as each particle is already in the vicinity of the target location after the previous particle is in contact with G. It also works under a condition that T is dynamically moving. In that case, G is dynamically updated according to the movement of T, and

the target locations of the particles are updated.

Using the geodesic line as the target line stabilizes the control and the state of the rope as it is the locally shortest path. We will only need to constrain the last particle of p_i^C to preserve the contact of C and T. This is an important feature if it is to be used for controlling robots to wind ropes.

4.3.3 Computing the movements of the control lines

For computing the movements of the ropes, we use an optimization framework that takes into account external force, inextensibility and bending stiffness of the ropes, kinematic constraints and collisions. Technically, we employ an idea similar to the Fast Projection (Goldenthal et al. [2007]): we first simulate the movements of the particles composing the rope by only taking into account the gravity, and then, project the updated position of the particles to a manifold that satisfies the constraints by solving an optimization problem. The individual constraints and the optimization process is explained below.

Inextensibility constraints: A control line or a rope is modelled by particles whose positions are $\mathbf{p}_i$ where $i \in [0 : n-1]$ and links between them whose length are r. The inextensibility constraints can be formed as:

$$C_e = \| \mathbf{p}_{i+1} - \mathbf{p}_i \|^2 - r^2 = 0. \tag{4.2}$$

Bending stiffness: For bending stiffness, we reformulate the idea used in Jakobsen [2001] as below:

$$C_b = \| \mathbf{p}_{i+1} - \mathbf{p}_{i-1} \| - 2r = 0. \tag{4.3}$$

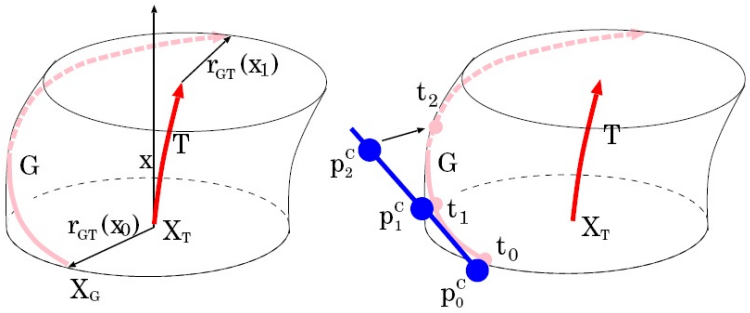

Figure 4.7: (left) The winding is computed by integration along the normal vector of T at X_T. (right) C winded around T by gradually attaching particles p_i^C from X_T.

Kinematic constraints: The active rope is controlled by moving one of its particles p^c towards its target location $\mathbf{p}_t$. As $\mathbf{p}_t$ can be too far away from p_c to be reached by one step, a series of intermediate target points $\mathbf{p}_k = \frac{k}{n_c-k}\mathbf{p}_c + \frac{1}{n_c}\mathbf{p}_t (k \in [1 : n_c - 1]$ are defined, where n_c is a constant integer defined by the initial distance between $\mathbf{p}_c$ and $\mathbf{p}_t$. In every iteration, the following kinematic constraint is imposed:

$$C_k = \| \mathbf{p}_c - \mathbf{p}_k \| = 0. \tag{4.4}$$

Collision constraints: We model ropes by linked capsules. The collisions between two capsules s_i and s_j can be resolved by imposing the following constraint:

$$C_c^{s_i,s_j} = D(Capsule_{s_i}, Capsule_{s_j}) - (r_{s_i} + r_{s_j}) >= 0 \tag{4.5}$$

where $D()$ returns the shortest distance between the two capsules, and r_{s_i}, r_{s_j} are the radii of the two capsules, whose values are same as r_C and r_T. We do not apply this constraint to adjacent capsules in each rope as their ends are already overlapping to compose the rope.

Projecting to the manifold of constraints: After adjusting the particle locations of the control ropes based on the gravity, their positions are mapped on to the manifold of the constraints defined above. This is done by computing the minimum particle displacement $\Delta\mathbf{p} = (\Delta\mathbf{p}_1^T, ..., \Delta\mathbf{p}_n^T)^T$ needed for satisfying all the constraints. As the inextensibility constraints, bending stiffness and collision constraints are nonlinear, we linearise them by the finite difference method. The collision constraints are handled as hard constraints as they are essential for preventing penetrations, which are visually influential. The rest are handled as soft constraints to increase the adaptiveness of the ropes. $\Delta\mathbf{p}$ is computed by solving the following optimisation problem:

$$\begin{aligned} \arg\min_{\Delta\mathbf{p}} \frac{1}{2}\Delta\mathbf{p}^T M^T M \Delta\mathbf{p} + \frac{1}{2}a\|\mathbf{J}_e\Delta\mathbf{p} - \alpha\|^2 \\ + \frac{1}{2}b\|\mathbf{J}_b\Delta\mathbf{p} - \beta\|^2 + \frac{1}{2}c\|\mathbf{J}_k\Delta\mathbf{p} - \gamma\|^2 \end{aligned} \tag{4.6}$$

subject to

$$\mathbf{J}_c\Delta\mathbf{p} + \sigma = \mathbf{O} \qquad \textit{only imposed if } \sigma < 0 \tag{4.7}$$

where $(\alpha, \beta, \gamma, \sigma)$ and $(\mathbf{J}_e, \mathbf{J}_b, \mathbf{J}_k, \mathbf{J}_c)$ are the values and Jacobians of (C_e, C_b, C_k, C_c), respectively. And a, b and c are coefficients where $a = exp(abs(\alpha) \times 10), b = 2000, c = 3000$.

A solution can be obtained by solving the following sparse linear equation

$$\begin{bmatrix} \mathbf{I}+\mathbf{J}_e^T\mathbf{J}_e+\mathbf{J}_b^T\mathbf{J}_b+\mathbf{J}_k^T\mathbf{J}_k & \mathbf{J}_c^T \\ \mathbf{J}_c & \mathbf{O} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{p} \\ \lambda \end{bmatrix} = \begin{bmatrix} a\mathbf{J}_e^T\alpha+b\mathbf{J}_b^T\beta+c\mathbf{J}_k^T\gamma \\ -\sigma \end{bmatrix}$$

After $\Delta\mathbf{p}$ is computed, the particle locations are updated by adding the corresponding elements to $\mathbf{p}_i$. This procedure can be iterated until all the constraints are satisfied, although we find the first iteration usually produces satisfactory results.

4.4 Controlling Cloth by Geodesic Control

The idea of Geodesic Control explained in the previous section can be applied for controlling cloth to wrap it around an object or knotting two ends of cloth. We explain about these processes in this section. Knotting is done in the same way as ropes, but with an additional procedure based on medial axes to reduce the self-penetrations between the cloth.

4.4.1 Wrapping by Geodesic Control

Each wrapping manoeuvre is achieved by gradually increasing the contact area of the control line of the cloth with the target curve defined over the object or its convex hull. The procedure to compute these lines and control the cloth using them is first explained. The movement of the rest of the cloth is computed passively using a generic simulator based on particle physics. This process is explained next.

Guiding wrapping by control lines: We now define a control line and a target line on the cloth and object, respectively to guide the cloth to wrap around the object by Geodesic Control. The control lines are straight lines over the cloth defined by the user. In most cases, they are lines connecting the centre and the corners of the cloth. The target lines are geodesic lines defined either over the wrapped object or its convex hull, depending on the wrapping style. In the initial configuration, the control line must be in contact with the target line (see Figure 4.8, left). Either the object must be shifted over the cloth or the target curve needs to be redefined if this condition is not met.

As the degrees of freedom of a cloth is much greater than that of a rope, it is necessary to further constrain it to make it robustly wrap around the object. This is achieved by straining the cloth in the direction of the control line, which eases the process of overlapping the control and the target line.

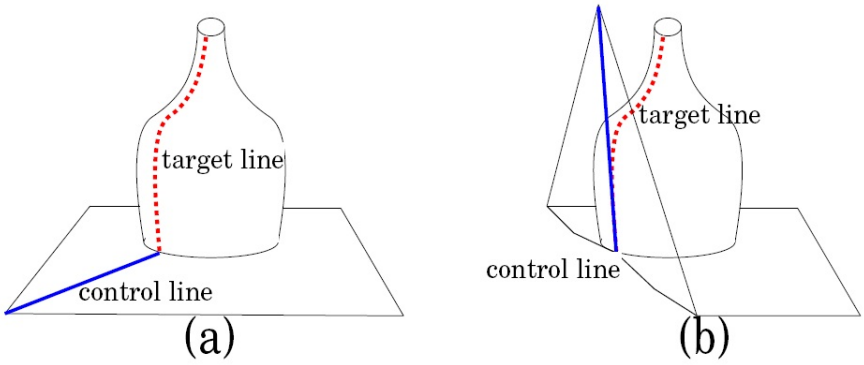

Figure 4.8: (a) The control line on the cloth and the target line on the object. (b) The cloth wrapped around the target object by overlapping the control line on the target line.

After the cloth is strained, the control line is overlapped with the target line by bending it at the edge of the object that the control line is in contact with (see Figure 4.8, right). The control line is rotated until it fully contacts the target curve on the polygon that includes the last contact point. This process is repeated until the control line fully overlaps with the target line.

In practice, the cloth is manipulated by the series of particles defined over the control line. Their movements are updated by solving the optimization problem described in Section 4.3.3 (Equation 4.6 and Equation 4.7).

Cloth movements by particle physics: Once the movement of the control lines are computed, we constrain the position of the particles that compose the control lines and compute the movements of the rest of the cloth particles by a generic simulator based on particle physics. Regarding the physical properties, we take into account the stretchiness and the bending stiffness between the particles, whose values are set to 0.9 and 0.011. The distance between the particles in the grid are set to 0.2 and the thickness of the cloth is set to 0.3, which produces a bounding sphere whose radius is 0.15 for collision detection.

We use PhysX 2.84 (NVidia [2008]) for the physical simulator.

4.4.2 Knotting Cloth

For knotting cloth, we apply the Geodesic Control to the control lines that are defined on the surface of the cloth.

Similar to the process of winding ropes, we define the active line and the target line. A bounding tube of a constant radius is defined around the target curve, and

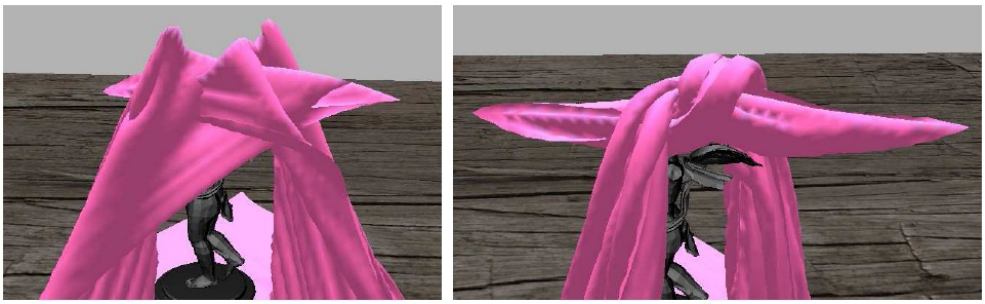

Figure 4.9: Self-penetrations happening during knotting two ends of a cloth by a generic simulator (left). They can be greatly eliminated by separating the ends by the medial axis (right).

a geodesic line is computed on the surface of the tube. Finally, the active curve is wound around the target curve by gradually locating the particles of the active curve around the bounding tube. The tube is defined only for a short distance around the area where the winding is going to be made. This is to avoid a large area of the cloth to be squeezed. The radius of the tube is computed based on the amount of cloth involved in the knotting. We also add an extra post-processing stage to minimize the amount of cloth-cloth penetration. These processes are explained below.

Reducing self-penetration using the medial axis: Self-collision is a serious issue for particle-based cloth simulators. The artifacts are evident when simulating complex interactions such as knotting using generic simulators that use naive collision detection approaches (see Figure 4.9, left). Although its effect can be reduced using asynchronous stepping as done in Harmon et al. [2009], this requires a huge amount of computation, which may not be acceptable for real-time applications. Here we eliminate such artifacts by inserting an additional medial axis layer (medial sheet) between the two control lines (see Figure 4.10,(a)) and forcing the associated particles to stay on the same side of the medial sheet.

Each free particle f is associated to a region of the control line composed of M line segments, which is denoted here by $\mathbf{L}_f$. $\mathbf{L}_f$ is the closest set of line segments from f along the surface of the cloth that is composed of M line segments (M is set to 3 in our examples). Figure 4.10,(b) shows an example of free particles and their associated line segments.

After computing $\mathbf{p}_f$, which is the position of particle f, using the physical simulator while constraining the control lines, we examine the distance between f and all the line segments; if the closest particle is not included in $\mathbf{L}_f$, it is moved towards the closest point on $\mathbf{L}_f$. More specifically, let us denote by d_f^i the distance between f

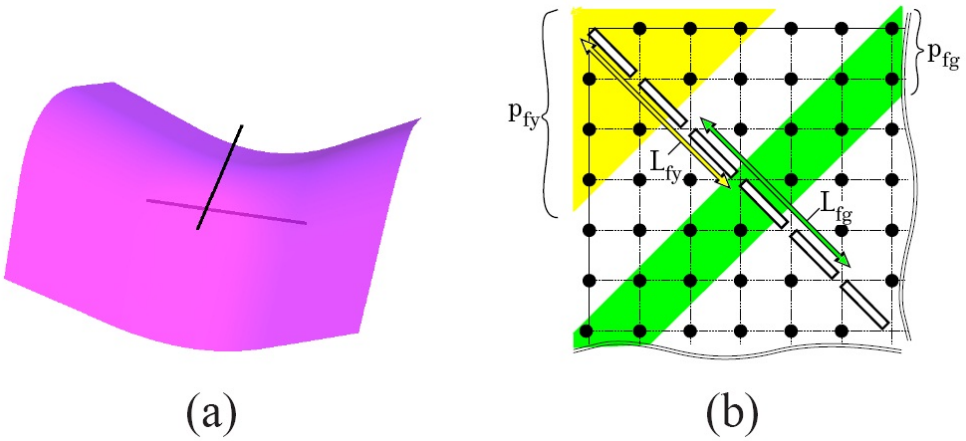

Figure 4.10: (a) A visualization of the medial sheet computed from the configuration of the two control lines. It is to be noted that we do not explicitly compute the medial sheet. (b) The enlarged top-left corner of the cloth : The free particles in the yellow (p_{fy}) and green (p_{fg}) region are associated to the line segments specified by the yellow (L_{fy}) and green arrows (L_{fg}

and all the line segments $l_i (i \in [1:m])$. If $\arg\min_i d_f^i (1 \leq i \leq m) \notin \mathbf{L}_f$, we move $\mathbf{p}_f$ towards its projection point on $\mathbf{L}_f$, which is defined here by $\mathbf{p}_f^p$. Namely, $\mathbf{p}_f$ is updated by $\mathbf{p}_f \leftarrow \mathbf{p}_f + a\frac{\mathbf{p}_f^p - \mathbf{p}_f}{\|\mathbf{p}_f^p - \mathbf{p}_f\|}$ where a = 0.01. This process is applied for all the free particles. Then, we run the simulation again while constraining all the $\mathbf{p}_f$'s that have been updated. This process is repeated until either each free particle is closest to its associated line segment, i.e., located within the Voronoi regions of its associated line segment, or the maximum number of iterations is reached, which is set to 3 in our experiments.

Our method can significantly reduce the penetrations between the two cloth-piece during the knotting procedure (see Figure 4.9, right). The twisted medial surface also helps to produce a natural effect of a cloth being squeezed.

4.5 Experimental Results

In this section, we first show an example of rope winding. Next, we show examples where various furoshiki wraps are generated by sequentially applying the winding and knotting techniques. Finally, we discuss about the computational costs.

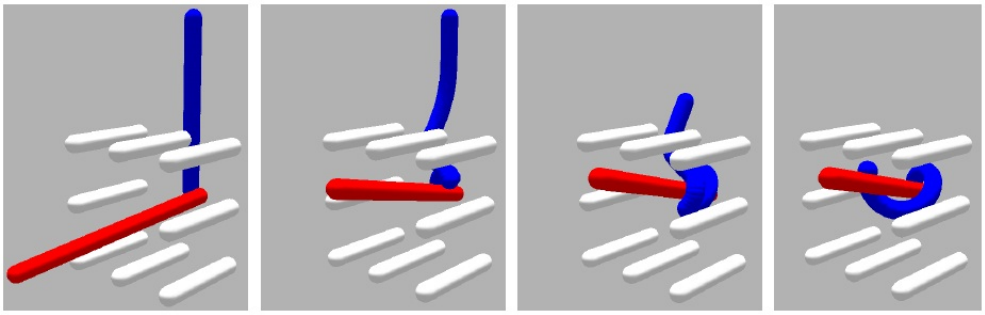

Figure 4.11: Winding one rope around another dynamically moving rope using Geodesic Control in an environment with obstacles.

4.5.1 Rope winding

We first show an example of winding one rope around another moving rope in an environment with multiple obstacles. Snapshots of the scene are shown in Figure 4.11. This is an example that is difficult even using manual control as it is necessary to avoid the obstacles and unnecessary winding while following and winding around the moving rope. Geodesic control can successfully achieve the task without global path planning.

4.5.2 Furoshiki wrapping

Figure 4.12: Target objects used in experiments

We wrap objects shown in Figure 4.12 by a two granny-knot box wrapping, a wine-bottle wrapping and a basket wrapping.

Two granny-knot box wrapping:

A two granny-knot box wrapping is produced by repeating the process of wrapping and making a granny-knot at the top of the object using the opposite corners of the

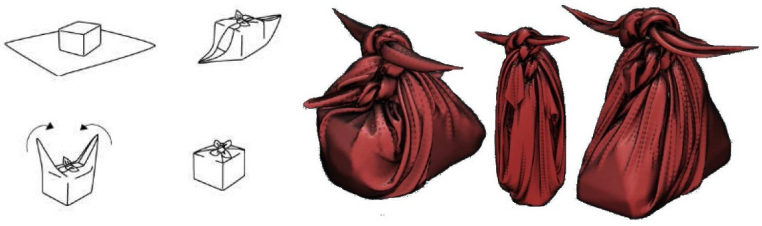

Figure 4.13: The FSM of box wrapping (left) and the jeep, cupid and sphinx wrapped by this style (right)

cloth. The procedure of the wrapping can be found in Figure 4.13, left and Table A.2. We show results of wrapping a jeep model, a sphinx sculpture and a cupid sculpture, which all have different structures. Despite such variations, the objects are successfully wrapped as shown in Figure 4.13, right. A sequence of snapshots of the sphinx model are in Figure 4.14.

Figure 4.14: The snapshots of the two granny knot wrapping of a sphinx model.

Wine bottle wrapping:

The wine bottle wrapping is suitable for wrapping cylindrical objects, such as a wine bottle. The procedure of the wrapping can be found in Figure 4.15, left and Table A.3. This wrapping involves winding the cloth around the bottle using Geodesic Control. We show examples of wrapping a wine bottle and the cupid structure in

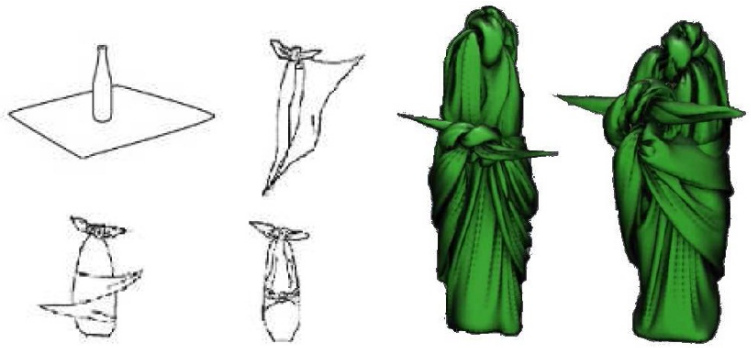

Figure 4.15: The FSM of wine bottle wrapping (left) and a bottle and cupid wrapped by this style(right)

Figure 4.15, right. A sequence of snapshots of the wine model are in Figure 4.16.

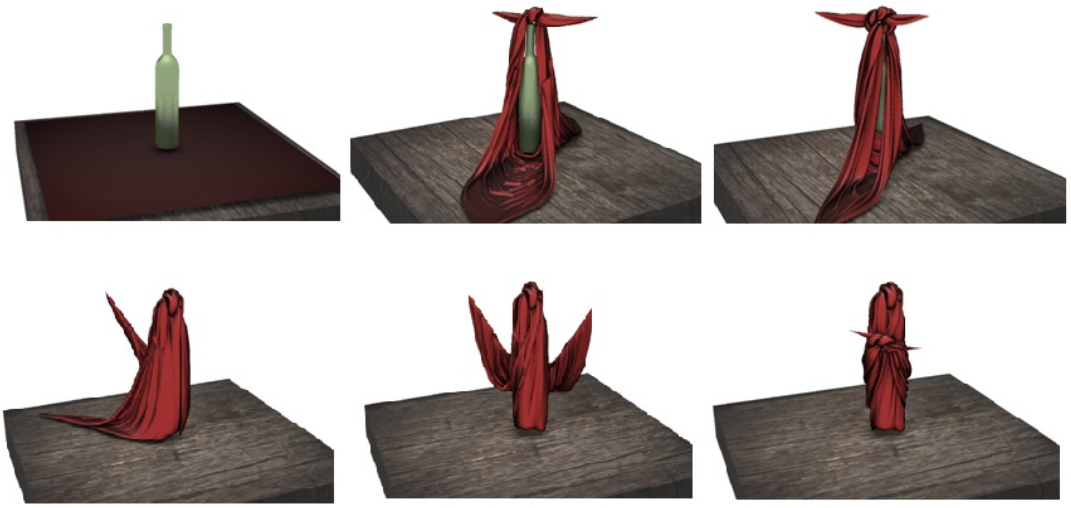

Figure 4.16: The snapshots of the wine bottle wrapping of a wine model.

Each control strategy is robust enough to be applied on different models. So we can apply different wrapping strategies onto the same object as well. See Figure 4.17.

Basket Wrapping

Finally, a basket wrapping was done to a basket. The procedure of the wrapping can be found in Figure 4.18, left and Table A.4. In this example, the cloth winds around the handle, which is also guided by Geodesic Control. The result is shown in Figure 4.18, right. A sequence of snapshots of the basket model are in Figure 4.1.

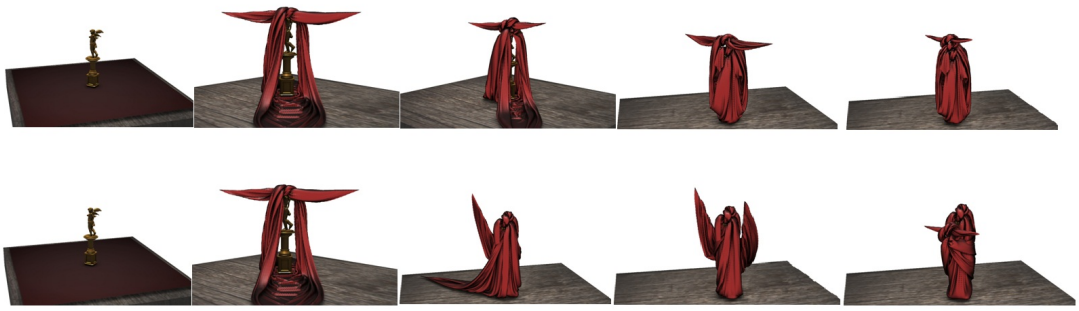

Figure 4.17: The snapshots of two wrapping styles for the same cupid model. Top: two granny knot wrapping. Bottom: wine bottle wrapping.

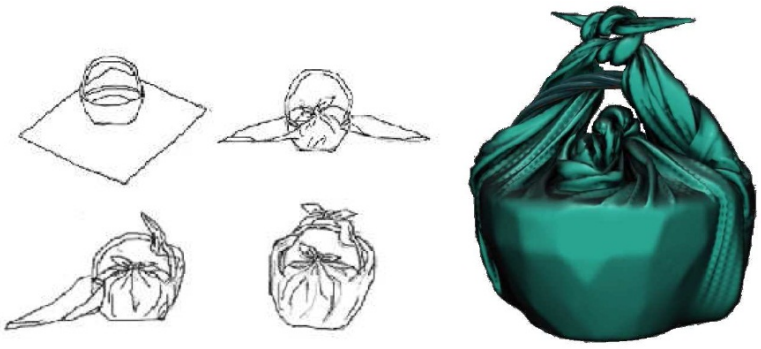

Figure 4.18: The FSM of basket wrapping (left) and a basket wrapped by this style(right)

4.5.3 Computational Costs

Regarding the complexity of our method, the set-up and solving of Equation 4.6 is $O(m)$, computation of the medial axis and updating the particle positions is $O(m \times n)$, and the particle simulation of the cloth based on a generic simulator is $O(n)$, where m is the number of rods on the control line and n is the number of particles composing the cloth.

The breakdown of the computation of each step in the descending order is collision resolving by the medial axis, (0.2 second per frame), the set-up and solving of Equation 4.6 (0.016spf) and particle simulation (0.001spf). Note that we only apply a naive approach that computes the distance of every pair of particles and line segments for the collision resolving. An acceleration can be achieved by applying methods such as oct-trees.

The cloth used in the system is composed of 26569 (163×163) particles, and we

can generate an animation of 1 second physical simulation in roughly 3 seconds, using a system with a Intel Core i7 2.93GHz CPU, 6GB RAM and NVidia GeForce GTS 240. The computation for one frame is completed in around 0.22 seconds. We used UMFPACK for solving the optimization problem and PhysX for the cloth simulation.

4.6 Discussions

One of the advantages of our method is that the complex manoeuvres of wrapping and knotting are highly abstracted. Therefore, the description of the manoeuvres is applicable to different combinations of objects and pieces of cloth, as far as the size of the cloth affords it.

Successful winding can be easily achieved by Geodesic Control, even under conditions where there are multiple obstacles or the target is dynamically moving, which can be difficult for other methods such as Topology Coordinates (Ho and Komura [2009b]) or "follow the leader" (Brown et al. [2004a]). The Topology Coordinates is another local control method that can be applied for winding motion. If there are obstacles that the rope should not wind around, the user needs to specify that as additional constraints. When there are multiple obstacles, such extra constraints can cause local minima. The follow the leader approach requires global path planning of the rope tips and there is a chance that the loose area gets hooked with obstacles in the environment. Geodesic Control, however, can robustly wind around another rope or object without any costly computation. The initial crossing is the only condition for starting the control, but that is not difficult, especially for tasks in which the ropes are open ended.

Geodesic Control is a local control solution for winding and wrapping procedures, and it is not a solution that can replace global path planning. In our examples, the FSM takes on the role of global path planning. For manoeuvres such as passing the rope between a series of ropes or a narrow area, a global path planner that provides the target geodesic lines to follow is needed. One possible extension is to set up a FSM that is based on homotopy groups (Bhattacharya et al. [2011]). The homotopy groups categorize paths based on the topology of the paths. Once a homotopy group is given, we can compute the geodesic path within that homotopy group.

There are several ad-hoc steps in our system. Some parameters of the each manoeuvre are manually specified or set by simple ad-hoc rules. For example, the geodesic lines on the objects (the target lines) are manually defined by the user. The user spec-

ifies the position of the end point and then a geodesic line is computed by finding the shortest route connecting the two end points. One problem of this approach is that there are cases where the geodesic line takes a different route from that which is desired (such as taking the shorter arc on a sphere instead of the desired longer arc). One way to solve this problem is to let the user also specify the tangent vector at one of the end points, compute the geodesic curve based on this, and finally let the system adjust it such that it passes through the other end point. Automatically deciding parameter values such as the position of the knot based on the geometry of the target object, the length of the control line and the style of the wrap is a challenging but interesting topic to work on.

Our approach can also be applied to overlap control lines to non-geodesic lines defined on objects. As mentioned earlier, the advantage of overlapping the control curve with geodesic lines of the object or its convex hull is in its physical stability; only the last control point needs to be constrained to keep the same configuration. For animation purposes, this condition can be dropped; an interface to letting users draw curves on surfaces and allowing flexible objects follow them for modelling objects and creating animation would be an interesting extension.

4.7 Conclusion and Future Work

In this chapter, we have proposed a method to synthesize animations of wrapping and knotting cloths by introducing discrete representations and control methods to interpolate such representations. Currently, the movement of the flexible object is guided by the control curves. For animation and control purposes, it is necessary to realize such movements through the interaction of the robot/character with the flexible object. This is an interesting future direction for research.

Chapter 5

Free wrapping

This chapter is mainly based on my paper submitted to ACM Transaction on Graphics that is currently under revision. In this chapter, we proposes a new method for synthesizing animations such as characters putting on or taking off clothes, wrapping objects with a piece of cloth or a bag, and controlling characters such as fish to catch and eat prey. Such scenes are difficult to handle due to the high dimensionality of the state parameters, the low dimensionality of the control parameters and the lack of a representation that describes the relationship of the deformable objects and the other scene components.

We model these problems as free wrapping tasks. We first propose an efficient and general approach to solve such problems using winding numbers as the guidance of the control strategy, which is suitable for convex target objects. And then we propose a new body-centred parameterisation called Electric Coordinates which encodes the outer space of the target characters or objects in the scene. Using the Electric Coordinates, we can easily control the characters to conduct complex manoeuvres such as wrapping and clothing using a small set of abstract parameters.

The abstract nature of our methods enables the same strategy of wrapping and clothing to be applied for deformable and target objects of different sizes, geometry and topology, which greatly automates the control of such manoeuvres. We demonstrate their effectiveness and versatility in synthesizing a wide variety of motions with close interactions.

5.1 Introduction

Controlling deformable objects with many degrees of freedom, such as a cloth, in wrapping around a reference object of either convex or concave shape, such as a character, is a challenging problem in computer animation and robotics. Such manoeuvres are required when *e.g.* controlling a snake or a fish to swallow prey, covering an object with a piece of cloth, or putting on or taking off a character's garments. Such animations are difficult to produce due to the high number of degrees of freedom in the deformable object (such as a cloth composed of many particles), the low number of control parameters, and the lack of a representation that describes the relationship of the deformable objects and the characters (or other reference objects) in the scene.

Automatically manipulating high dimensional objects by low dimensional control signals is a difficult problem. Typically, many-DoF objects are controlled by specifying the movements of a small number of control particles. In such cases, a good abstraction of the state space is needed to describe the state of a high dimensional object in terms of a small number of parameters.

For example, the meaningful actions that a cloth (an object with very many DoFs) can perform can be typically described parsimoniously, *e.g.* "wrap around an object", "put on a glove". We aim to quantify such intuitive concepts *e.g.* "the degree to which a glove is put on", and so simplify the problem of control. This allows us to express the problem of control and animation of many-DoF objects in terms of an optimization problem in which a small number of parameters (parsimoniously characterising the semantically important relationship between objects) is optimized for in order to achieve the desired goal configuration.

In this chapter, we concentrate on guiding wrapping movements of a deformable 2D open surface (referred to as the **deformable object**, *e.g.*, a piece of cloth or a bag) around a closed 2D manifold surface (referred to as the **reference object**, *e.g.*, a rigid object, an articulated character or a deformable surface), through a limited number of control parameters such as the trajectories of the opening of a garment.

In the first half of this chapter, I will start with a simple task described in Figure 1.1, and introduce Winding Number as the representation of the wrapping task. On top of it, a framework is presented to show how motion planning can be done via the mappings between the object-centric manifold and task-centric manifold. Although mainly for convex objects, it is faster and does not require pre-computation.

In the second half, based on a similar framework, a further generalised parameteri-

sation of the task-centric manifold involving arbitrarily deformable objects is proposed, based on the concepts from electrostatics, to guide complex manoeuvres such as wrapping and dressing. In addition, a generalised measure is introduced which quantifies the extent to which an object is wrapped. This can be modelled as an electric flux around a charged conductor. Finally, a body-centric coordinate system, called Electric Coordinates, is defined which parameterises the outer space of a reference object in a way analogous to polar coordinates. Using this framework, one can easily control characters conducting complex manoeuvres, such as wrapping and dressing, through a small number of abstract parameters. As a result, the same strategy for wrapping and dressing can be applied to objects of different sizes, geometries, and topologies, which can greatly automate the control of such manoeuvres. I will demonstrate the effectiveness and versatility of the framework in synthesising a wide variety of motions with close interactions.

5.2 Contributions

1. We propose new parameterisations of the space, based on the concepts of winding number and electrostatics, which are suitable for controlling deformable objects for wrapping and dressing.

2. We propose abstract, intuitive, shape insensitive measures of coverage of one object by another.

3. We propose methods to synthesize complex wrapping movements by linearly interpolating the state of the deformable object in the state space of parameters, which does not require a global path planner.

5.3 Related Work

Cloth animation is an important topic in computer graphics. Most of the recent work focused on stable simulation (Baraff and Witkin [1998]; Curtis et al. [2008]; Harmon et al. [2009]; Teschner et al. [2003]), interactive design (Emmanuel Turquin et al. [2004]; Igarashi et al. [2009]; Mori and Igarashi [2007]; Turquin et al. [2007]), and adding high resolution detail to low resolution cloth (Bergou et al. [2007]; de Aguiar et al. [2010]; Wang et al. [2010a]).

Our focus in this chapter is cloth *control*, which is a relatively young area where comparatively little work has been done. Manipulation of cloths can be classified into three different categories: (1) controlling the raw geometry of the cloth, (2) using the contact information for guiding the movements, and (3) defining a state space based on the spatial relationship between the objects involved. We now review previous work in each of these categories.

Control of raw geometry in Cartesian space: A traditional way to control a cloth is to specify 3D trajectories of some control particles. However, it is difficult to devise a natural way to extrapolate the motion of the control particles to the rest of the cloth. Kondo et al. [2005] propose a keyframe animation approach to control deformable characters using elastic force. Wojtan et al. [2006] propose a spacetime optimization approach to let artists control the movement of a cloth through keyframes. For synthesizing animations such as characters putting on garments, the keyframing process itself can be very time consuming as it is necessary to specify a lot of keyframes to define the desired motion. More importantly, the keyframes cannot be recycled when the geometry of the characters or the deformable objects involved are altered.

To summarize, the drawbacks of using the raw geometry is that excessive manual control is required, and the synthesized animation cannot be recycled for later use.

Control by contact information: Specifying contact information is an intuitive way to control a cloth. Igarashi and Hughes [2002] put garments on characters by overlapping marks on the garment and the character. Wang and Komura [2012] propose to control deformable objects such as ropes and cloths by gradually increasing the contact area between the deformable object and the reference object.

The problem of using contact information as a representation is that it may not be consistent or may not even exist during some wrapping and dressing processes. For example, when passing limbs through trouser legs, the limbs may be in contact with the trousers in an unpredictable fashion, making it difficult to use such information for representing the motion.

Control based on spatial relationship: Modelling of spatial relationships has been applied for deformation transfer (Zhou et al. [2010]) and motion retargeting (Ho et al. [2010]). Zhou et al. [2010] apply the minimal spanning tree to preserve the relationship between a dress and a character while dancing. Ho et al. [2010] use the Delaunay tetrahedralisation to encode the spatial relationship between interacting characters. The structures used in these works are not suitable for parameterising the continuous

configuration space due to their discrete nature. A continuous parameterization is more suitable for synthesizing novel motions.

Ho and Komura [2009b] use the Gauss Linking Integral to guide movements of a garment when dressing. A character successfully passes its arms through the sleeves of a shirt in their demo. This approach is successfully applied for controlling robots to put a shirt on humans (Tamei et al. [2011]). In these approaches, the body is represented by articulated 1D links and the surface-surface relationship is not considered. However, modelling surface-surface relationships would be advantageous for animating close interactions between a garment and a character. For example, in the case of putting on a sock, the goal is achieved when the sock fits tightly around the foot and the lower part of the sock is covering the heel. We are going to seek an approach to quantify such relationships in this chapter.

Yuki Igarashi et al. [2009] propose to unwrap a cover by dragging it in the opposite direction of the opening. The coverage is evaluated by checking if rays casted in the normal direction at each vertex intersect the cover. Their approach is only applicable for objects of convex shapes as the rays may intersect the cover wrapping other parts of the surface in the case of concave objects. In this chapter, we propose a more mathematically thorough measure of coverage applicable to both concave and convex objects, and propose a control strategy for wrapping based on this measure.

In summary, a continuous parameter space based on spatial relationship is most suitable for guiding wrapping-like movements. However, there has been little work that quantifies the surface-surface relationship of a deformable object wrapping around a reference object. This work achieves this.

5.4 Wrapping control

In this section, a simple representation for free wrapping is given, and a control method, *wrapping control*, is equipped with the representation to control a system with many DoFs by solving a local optimisation problem per frame. The controlled system is represented by a polyline in 2D cases and by a polygon mesh in 3D cases. The target object is represented by a point or a 2D / 3D convex hull. The winding number is computed by projecting the controlled system onto the unit hemisphere whose centre is inside the target object.

5.4.1 Winding Numbers

For a simple task that resembles Figure 1.1, the winding number is a good parameterisation of the task-centric manifold. The winding number (Wikipedia contributors [2012]) computes how many times a point is surrounded by a closed 2D curve in the counter-clockwise direction, or whether an object is within a closed manifold surface in 3D case.

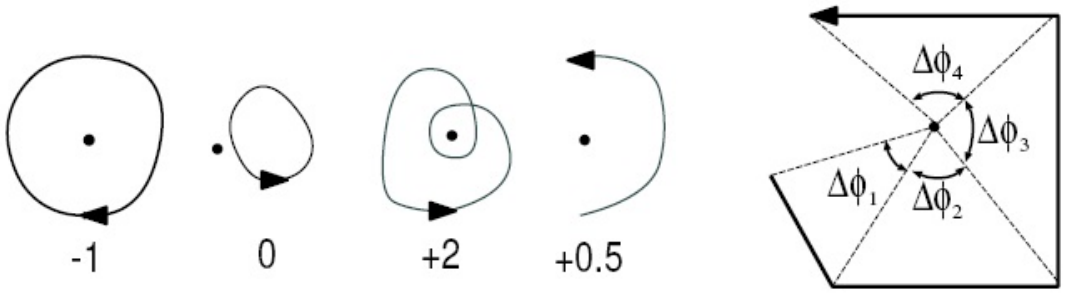

Figure 5.1: Example of 2D curves and the winding numbers (left). Winding numbers can be computed for closed curves as well as open curves. We compute the number as cumulative angles between radial rods (right).

Figure 5.1, left, shows the winding numbers between different curves and a target point. It can be observed that the more the point is surrounded by the polyline, the higher the winding number is. In computer graphics, the method has been applied for judging whether a point is within a 3D shadow volume (Crow [1977]). It is a topological entity, which means the value is invariant under geometric deformation of the curve or translation of the point, unless the curve / surface crosses over the point. For closed manifolds, it can be easily computed by drawing a line from the point outwards and checking how many times it intersects with the surrounding curve or surface.

Although winding numbers are usually computed for closed manifolds, they can also be computed for open manifolds. When they are computed for open curves and surfaces, the value varies according to how much a curve (in 2D cases) or a surface (in 3D cases) is surrounding a point. We make use of this feature to guide the controlled system to surround other objects.

In 2D, the winding numbers for parametric curves can be computed by:

$$w = \frac{1}{2\pi}\int_0^1 \frac{x(t)-x_0}{r^2}dy - \frac{y(t)-y_0}{r^2}dx \tag{5.1}$$

where $(x(t), y(t))$ is the trajectory of the curve, (x_0, y_0) is the point to be surrounded and $r^2 = (x(t)-x_0)^2 + (y(t)-y_0)^2$. In practice, we represent the winding curve by radial rods and accumulate the turning angle between the rods (see Figure 5.1, right):

$$w = \sum_{i=1}^{N_k} \Delta\phi_i \tag{5.2}$$

where N_k is the number of rods and $\Delta\phi_i$ is the angle that rod i winds around the target point.

The winding of a surface around a point can be computed in a similar way as to the form factor in radiosity (Cohen et al. [1993]). We define an unit hemisphere centred at the target object, project the deformable object onto the hemisphere, and sum the covered area (see Figure 5.2, left):

$$w = \int_S \cos\theta dS \tag{5.3}$$

where S is the wrapping surface, dS is an infinitesimal area of the wrapping surface and θ is the angle between the line from the centre of the object towards dS and the normal vector of dS. Assuming the surface of the surrounding object is decomposed into triangles, Equation 5.3 can be computed by projecting each triangle onto the hemisphere, and summing the signed projected area :

$$w = \sum_{i\in T} -sgn(\cos(\theta_i))(\pi - \alpha_i - \beta_i - \gamma_i) \tag{5.4}$$

where T is the set of triangles composing the surface of the surrounding object, $\alpha_i, \beta_i, \gamma_i$ are the three angles of the projected i-th triangle and θ_i is the angle between the vector connecting the centre of the triangle and the centre of the object, and the normal vector of the triangle (see Figure 5.2, right).

5.4.2 Wrapping Control

We use the winding number to guide the wrapping movements. Let us assume the system state is represented by generalised coordinates $\mathbf{v} = (v_1, ..., v_N)$. We make the system wrap around a target object by solving a least-squares problem. This is done by minimising the weighted squared updates of $\mathbf{v}$ subject to constraints including the

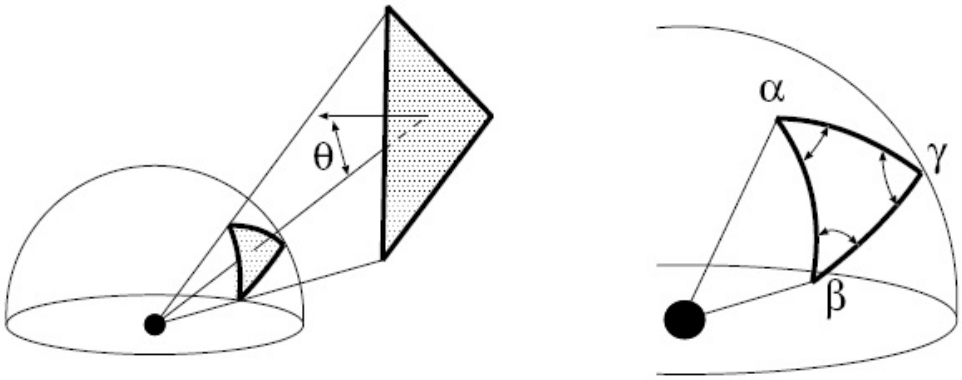

Figure 5.2: The 3D winding number between a point and a deformable surface can be computed by projecting the surface onto the hemisphere surrounding the point (left) and summing the area. The area of a projected triangle onto the unit hemisphere can be computed by subtracting the sum of the three angles of the spherical triangle from π.

increment of the winding numbers and the additional linear constraints such as distance constraints. Next we explain each of these terms and then show how to compute the updates of the system per frame.

Minimizing the updates: Minimizing the updates of the system is an approach widely used in character animation and inverse kinematics, which results in smooth and realistic movements. The weighted squared updates can be evaluated by $\Delta\mathbf{v}^{\mathbf{T}}\mathbf{M}\Delta\mathbf{v}$ where $\Delta\mathbf{v}$ is the updates of the system, $\mathbf{M}$ is a matrix that weighs the movements of each DOF and Δw is the step increment in the winding number. $\mathbf{M}$ is set to an identity matrix for the particle based systems. For the articulated systems, the entries in $\mathbf{M}$ for the root translation and rotation are set to a small constants. We use 0.001 and 0.01 respectively in our experiments although a small range near these two numbers is valid depending on the user's requirements of the smoothness of the generated movements. The rest of the entries are all set to 1.

Constraining the winding numbers: We linearise the relationship between the winding numbers and the generalized coordinates and use them as constraints when controlling the system. More specifically, the linear constraint of the winding number can be written as $\Delta w = \mathbf{J}\Delta\mathbf{v}$ where Δw is the step size of the winding number and $\mathbf{J}$ is the Jacobian of the winding number computed using Equation 5.2 for 2D cases and

Equation 5.4 for 3D cases by finite difference. Regarding the step size, the values can be controlled by the user; a smaller value will produce a smoother animation. In our experiments, we set the value to 0.001.

Additional constraints: Distance constraints between adjacent particles and any other fine tuning of the control can be handled through additional constraints. Such constraints can be summarized by $\mathbf{d} = \mathbf{C}\Delta\mathbf{v}$, where $\mathbf{d}, \mathbf{C}$ are the constants and Jacobian of the constraints.

Distance constraints are used for cloth and crowd control. We compute the linear relationship between the particle movements and the distances between the adjacent particles by finite difference to compute the Jacobian. Then a constraint to resolve the error of the distance between the particles is composed: $r - r_o = \mathbf{B}\Delta\mathbf{v}$, where $\mathbf{B}$ is the Jacobian, r is the current length between the two particles, and r_o is the desired length between the particles.

Computing the movements: Using the above-mentioned terms, we compute the updates of the system by solving the following least-squares problem:

$$\arg\min_{\Delta\mathbf{v}} \Delta\mathbf{v}^{\mathbf{T}}\mathbf{M}\Delta\mathbf{v} + a(\Delta w - \mathbf{J}\Delta\mathbf{v})^T(\Delta w - \mathbf{J}\Delta\mathbf{v}) + (\mathbf{d} - \mathbf{C}\Delta\mathbf{v})^T\mathbf{W}(\mathbf{d} - \mathbf{C}\Delta\mathbf{v}) \quad (5.5)$$

where a and , $\mathbf{W}$ are weight / weight matrix for the winding number and the additional constraints, respectively, which are needed to handle the constraints as soft constraints. Δw is target winding number increment/decrement. After some experimenting, we found proper values for these coefficients. We set them to $a = 0.01 \times w$, and for the entries of the distance constraints in $\mathbf{W}$, they are set to $\exp(abs(r_0 - r) \times 10)$.

$\Delta\mathbf{v}$ can be computed by solving the following linear problem:

$$\left(\mathbf{M} + \alpha\mathbf{J}^T\mathbf{J} + \mathbf{C}^T\mathbf{W}\mathbf{C}\right)\mathbf{v} = \alpha\Delta w\mathbf{J} + \mathbf{d}^T\mathbf{C}\mathbf{W}. \quad (5.6)$$

In order to keep the system stable and the updates uniform, we normalize the computed $\Delta\mathbf{v}$ and update the system using a fixed norm: $\mathbf{v} = \mathbf{v} + \beta\frac{\Delta\mathbf{v}}{\|\Delta\mathbf{v}\|}$, where β is set to 0.1.

Finally, the computed $\mathbf{v}$ is passed to the physical simulator for collision detection and handling. The positions of the particles are computed such that they are penetration free and the momentums of the particles are preserved. We use NVidia PhysX (NVidia [2008]) for the simulation.

5.4.3 Coverage Ratio

When the target object's shape is very different from a sphere, i.e., a long box, the winding number may guide the cloth in a direction such that it collides with the target object. This is because it only guides the wrapping surface to surround the centre point of the object (as the brown arrow in Figure 5.3, right). To cope with this problem, instead of a sphere, any geometry for which the winding number can be easily calculated is good for the control. Convex shapes are good candidates.

The coverage ratio is an amount that describes how much the target object is surrounded by the wrapping curve or surface. It can be computed by projecting the wrapping curve or surface onto the target object (see Figure 5.3, left), and dividing the projected area by the total surface area of the target object. In practice, we compute the coverage ratio by projecting each of the vertices of the wrapper's triangles onto the target surface then summing the area of the projected triangles. This is only an approximation, but we have found that it works satisfactorily for high resolution meshes such as a cloth.

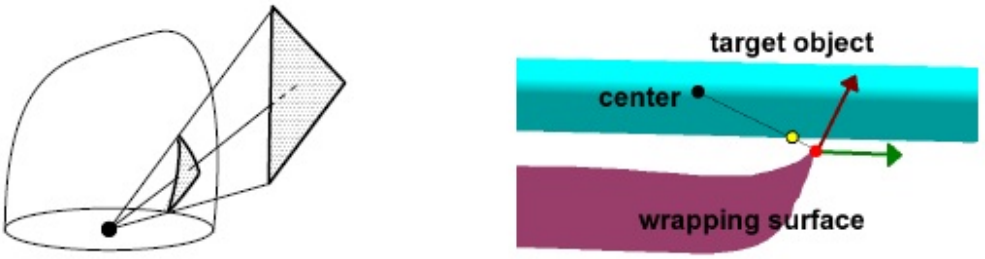

Figure 5.3: (Left) The coverage ratio can be computed by projecting the wrapping surface onto the surface of the target object and dividing its area by the whole surface area. (Right) Winding number guides the particle on the boundary (red point) to wrap around the object centre (in the direction of the brown arrow) resulting in collision. The coverage ratio solves the problem by guiding the particle to move in the direction parallel to the tangent plane of the target object (green arrow).

The coverage ratio guides the wrapping curve or surface to surround the target object by moving the boundary particles parallel to the tangent plane of the target object (as the green arrow in Figure 5.3, right). Assume the position of a boundary particle on the wrapping surface is represented by $\mathbf{x}$ (the red point in Figure 5.3, right), and its projection onto the surface is $\mathbf{x}'$ (the yellow point in Figure 5.3, right). The

most efficient way to increase the projected surface area (and the coverage ratio) is to move $\mathbf{x}$ parallel to the tangent plane at $\mathbf{x}'$ because the elements out of the tangent plane will be null after the projection.

As a result, moving $\mathbf{x}$ along the gradient vector of the coverage ratio will produce an effect of fully stretching the wrapping surface to envelope the entire surface of the target object.

The coverage ratio can be used exactly in the same way as winding number in Wrapping Control. We simply need to replace $\Delta w, \mathbf{J}$ in Equation 5.14 and Equation 5.15 with the increment and Jacobian of the coverage ratio, respectively.

5.4.4 Experimental Results

We demonstrate how the Wrapping Control can be used for controlling various systems, including cloth, articulated bodies and a crowd of characters. We first explain the setup of the individual systems and then present the results of controlling them with our method. Finally, we discuss the complexity and computational costs.

Cloth wrapping an object

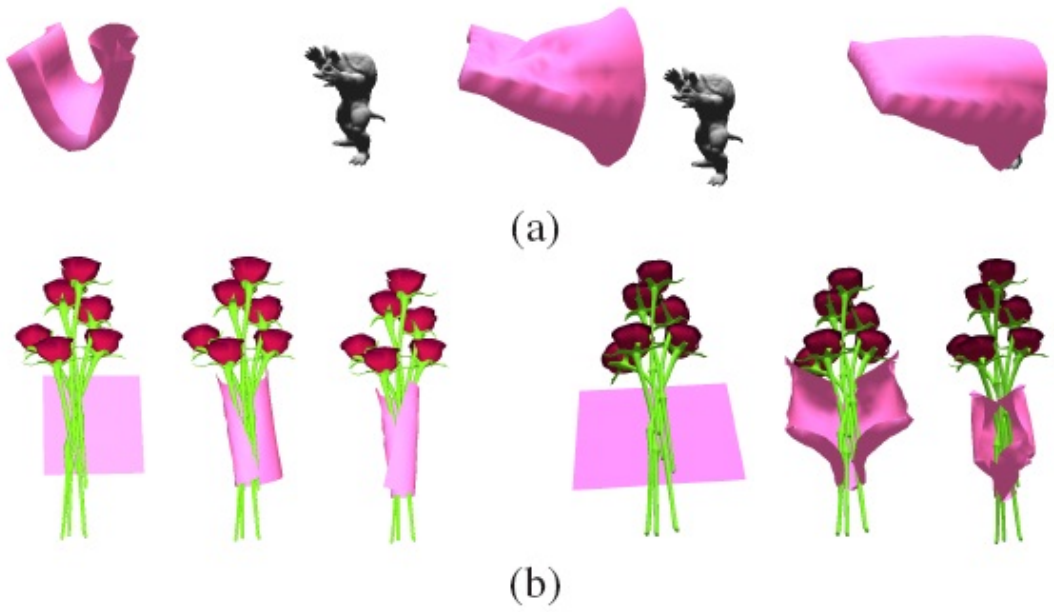

Figure 5.4: (a) A folded cloth unfolds and winds around the target object by simply gradually increasing the winding number. (b) A cloth enveloping a bundle of flowers using coverage ratio (left) and the winding number (right) as the criteria.

We model deformable models including a bag and a planar cloth using a network of particles, and control all the particles simultaneously to make them wrap around the target objects. In this case, $\mathbf{v}$ in Equation 5.14 includes all the position of the particles composing the deformable model: $\mathbf{v} = (x_1, y_1, z_1, ..., x_n, y_n, z_n)$ where (x_i, y_i, z_i) are the 3D coordinates of each particle and n is the number of particles.

We first show an example where we automatically control a bag wrapping around a mesh character. A series of snapshots in which the bag starts from a folded state, unfolds, and finally wraps around a target object is shown in Figure 5.4 (a). This motion is synthesized without any global path-planning, but only by specifying the increment of the winding number and minimizing the movements of the particles. Path planning an unfolding motion at the particle level can be difficult due to the large number of particles and the huge amount of state evaluation. This example shows the strength of the abstraction of our approach. Next, we show an example in which a planar cloth wraps around a long object. This example shows that the coverage ratio works much better for wrapping around objects which have a shape that is very different from a sphere. The result of wrapping a sheet of paper around the stem of the flowers using the coverage ratio is shown in Figure 5.4(b), left. In contrast, the control by the winding number can result in states shown in Figure 5.4(b), right. The cloth tries to approach the centre of the flower bundles, rather than enveloping the flower stems. We fitted a cylinder to the stems of the flower and computed the coverage ratio of the cloth with respect to this cylinder.

Crowd Control

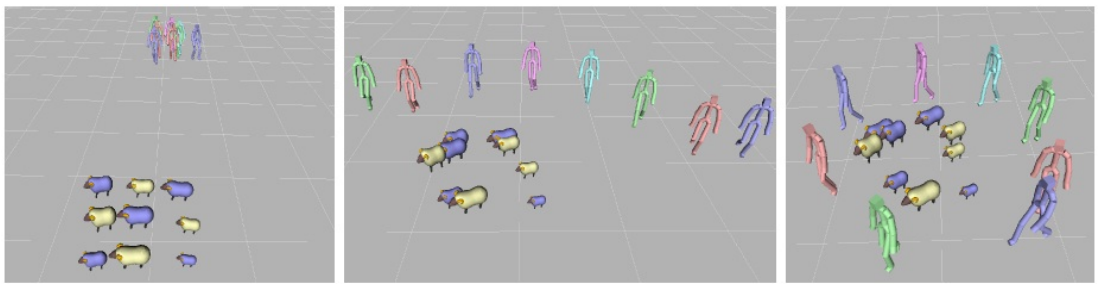

Figure 5.5: The crowd following and surrounding a herd of sheep. The characters can automatically adapt their formation to the shape produced by the sheep.

Finally, we tried to control a crowd of characters in surrounding an object. In this case, $\mathbf{v}$ in Equation 5.14 correspond to the positions of the characters in the 2D plane:

$\mathbf{v} = (x_1, y_1, x_2, y_2, ..., x_n, y_n)$ where (x_i, y_i) are the 2D coordinates of each character in the 2D plane and n is the number of controlled characters. We assume there is a virtual connection between the characters and the crowd can be represented by a 1D linkage. As our objective is to let the characters surround the target object, we first compute the polar coordinates of the characters using the centre of the target object as the origin, and then connect the adjacent characters along the angle axis by linkages. This approach will minimize the movements of each character in surrounding the object along the angle axis. In order to make the distance between the characters uniform after surrounding the target object, $\mathbf{l_0}$, the desired distance between the characters is set to L/n, where L is the circumference of the target object.

The experimental results are shown in Figure 5.5. Figure 5.5 shows snapshots of a crowd of characters following and surrounding a herd of sheep that are dynamically moving.

One problem that we face here is that the Wrapping Control guides the crowd to stretch out as much as possible before the distance constraints stop them. This movement is different from human crowds as usually humans start to surround the target only when they are close enough to the target. Therefore, we apply a hybrid approach, in which we make the crowd approach the target in a fixed formation first, and then gradually start to blend the movements computed by the Wrapping Control once they approach close enough to the target. It can be observed that the characters can dynamically adapt their formation to successfully surround the sheep.

Complexity and Computational Costs The complexity for solving Equation 5.15 is $O(N)$ for particle-based systems such as cloth and crowd, and is $O(N^3)$ for articulated systems such as a hand and a full body of a character, where N is the DOFs of the system. This is because the matrix is sparse for particle-based systems. The computation takes 0.01sec and 0.01msec for the cloth and crowd control, respectively, using one core of a Core i7 2.67GHz CPU. We use UMFPACK Davis [2004a] + GotoBLAS Goto and Van De Geijn [2008] for solving the linear problem. The resolution of the bag is $15 \times 15 \times 2$, and that of the cloth is 15×15.

5.4.5 Discussions

Here we explain the advantages and drawbacks of Wrapping Control.

The features of Wrapping Control that make it suitable for controlling systems with many DoFs include:

(1) **Abstractness**: The criteria we use are abstract values which are rotationally invariant, and are affected little by the local geometry of the curves and surfaces.

(2) **Simplicity**: The computation is straightforward and requires little adjustment of parameters. Unlike radiosity, we do not need to compute the occluded area, which requires more elaborate evaluation of the projected area.

These features allow the method to successfully guide the system to surround target objects without using a global path planning algorithm. They also make the system adaptive in the sense that the same principle can be used for controlling systems of different topology and DoFs. The system is also adaptive to the size and geometry of the target object.

On the other hand, shortcomings are present too. First, the core part of the representation is the evaluation of the progress of the wrapping task. Although it is been extended to objects with convex shapes, for non-convex objects, it is not applicable because the coverage is hard to calculate under current representation. Second, the starting configuration of the wrapping needs to be given by the user. To give an ideal initial configuration is crucial for wrapping tasks like putting on a sock where wrapping is supposed to begin with the toes not the heel. In this situation, if the initial position of the wrapper is randomly given, moving the wrapper to an ideal initial position requires global path-planning.

5.5 Electrostatics-guided Wrapping and Dressing

To target the remaining problems by Wrapping Control, we give a new and more generalised parameterisation of the task-centric manifold for free wrapping.

Figure 5.6: Applications of electrostatics-based motion synthesis: guiding a deformable object (a piece of cloth) to controlling a character to dress and undress.

5.5.1 Introduction

In this section, we concentrate on guiding wrapping movements of a deformable 2D open surface (referred to as the **deformable object**, *e.g.*, a piece of cloth or a bag) around a closed 2D manifold surface (referred to as the **reference object**, *e.g.*, a rigid object, an articulated character or a deformable surface), through a limited number of control parameters such as the trajectories of the opening of a garment.

We also address the problem of the lack of a representation that describes the spatial relationship between the deformable object and the reference objects composing the scene. In many cases, animators want the movements of a deformable object to be associated with the movements of other characters or objects involved. Contrary to this, the state of the deformable object is usually described by raw 3D Cartesian coordinates. Such a representation is agnostic of how the deformable object is moving with respect to the other (reference) objects in the scene. As a result, when the sizes, the shapes, or the movements of reference objects are changed, the motion of the deformable object is not updated accordingly, until the animator edits it or the motion is forced to adapt through physical constraints such as external forces.

We propose a new framework for handling interactions between a deformable object and a reference object in the scene. By using shape invariant properties of Gauss's law in integral form, we can compute a parameter called flux, which quantifies the degree to which a character or an object is surrounded by another deformable object (like a cloth). The flux can be used to guide wrapping-like movements.

We also define an object-centric curvilinear coordinate system, called Electric Coordinates. The latter relies on the equipotential surfaces and gradient lines of a harmonic scalar field (electric potential around a charged conductor) and parameterises the space outside the scene components in a way analogous to polar coordinates. As harmonic fields do not have local extrema, the above representation is suitable for path planning.

Using the flux and the Electric Coordinates as state values, the relationship between the deformable object and the reference object can be concisely represented by a small set of abstract parameters, which eases the guidance of the deformable object in wrapping around the reference object.

We demonstrate that our method is applicable for synthesizing a wide range of animations including wrapping flowers, a fish eating prey, and a character putting on or taking off garments. (convex or concave) and topology (can handle multiple objects

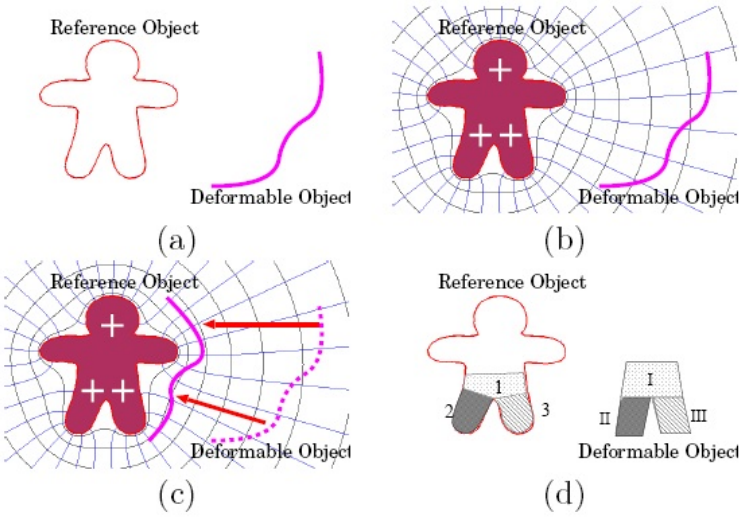

Figure 5.7: The overview of the method: (a) The deformable and the reference object, (b) charge simulation, (c) controlling motion, and (d) subtasking complex maneuvers.

as well as objects with holes), which makes the method suitable for arbitrary scenes in which characters closely interact with deformable objects.

5.5.2 Overview

Our method can be summarized as follows. Given the current state of the deformable and reference object (see Figure 5.7 (a)), we virtually charge the reference object and compute electrostatic parameters that represent the relationship between the deformable and the reference object (see Figure 5.7 (b) and Section 5.5.3). The deformable object is then controlled by optimisation over electrostatic parameters (see Figure 5.7 (c) and Section 5.5.4.1,Section 5.5.4.2). For more complex manoeuvres such as putting on trousers, the deformable object is divided into smaller regions and the whole procedure of wrapping is divided into multiple subtasks (see Figure 5.7(d) and Section 5.5.4.3).

5.5.3 Electrostatic Parameterization

The idea behind our method is to simulate the reference object as a charged conductor, and use its physical properties to define abstract and intuitive parameters that represent the spatial relationship between the deformable object and the reference object.

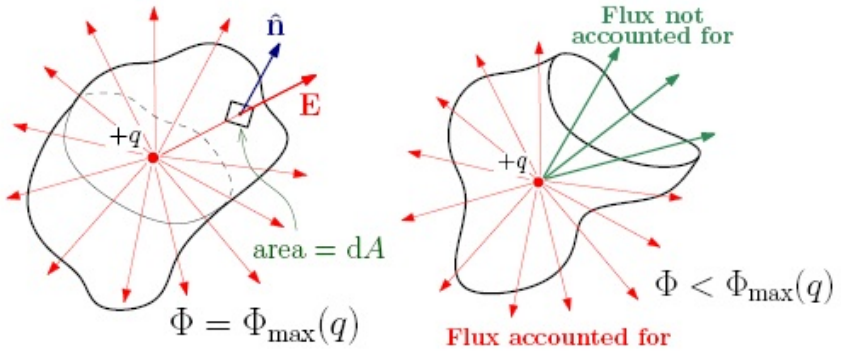

Figure 5.8: Illustration of the Gauss's Law.

More specifically, electrostatics provides us with the following two important concepts:

1. Coverage: A physical attribute called flux, which can be computed by Gauss's law (see Section 5.5.3.1), represents the coverage of the reference object by the deformable object. This can be used to guide wrapping movements.

2. An object-centric curvilinear coordinate system: A curvilinear coordinate system, which we call Electric Coordinates, can be defined using the electric potential and field lines induced by the charged reference object (see Section 5.5.3.2). The Electric Coordinates (EC) can be used to guide the deformable object to reach arbitrary parts of the reference object without using global path planning.

Our parameterization can greatly simplify the control of a deformable object to wrap around the reference object.

5.5.3.1 Computing Coverage by Gauss's Law

We now explain how to quantify how much the reference object is surrounded by the deformable object. Gauss's law, which states that the total amount of electric flux through any closed surface is proportional *only* to the enclosed electric charge, provides a good hint to quantify such coverage. Gauss's law in integral form can be formulated as follows:

$$\Phi = \oint_S \vec{E} \cdot \mathrm{d}\vec{A} = \oint_S \vec{E} \cdot \vec{n} \mathrm{d}A = \frac{Q}{\varepsilon_0} = \text{const}, \tag{5.7}$$

where $\vec{E}$ is the electric field being integrated over the surface S that is surrounding the charged object (with charge Q), $\mathrm{d}\vec{A}$ is an infinitesimal region of S (a vector with area $\mathrm{d}A$ pointing in the normal direction $\vec{n}$), and ε_0 is the electric constant (see Figure 5.8,

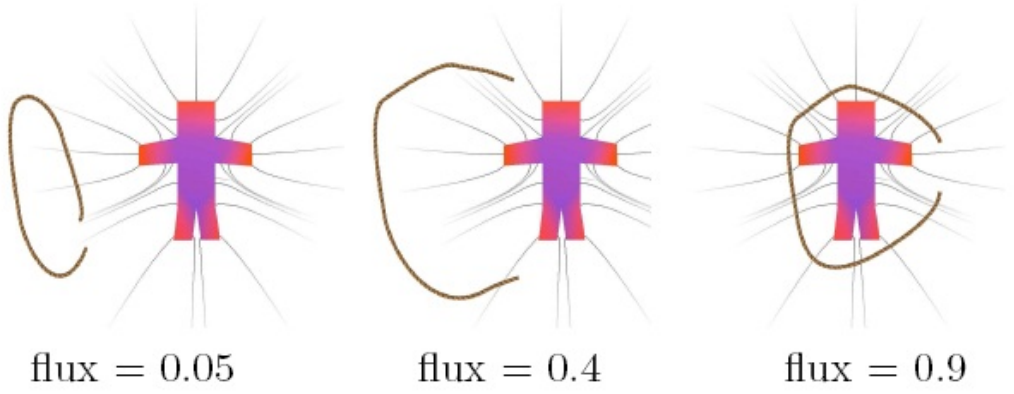

Figure 5.9: Different configurations of the deformable object and the reference object, and the corresponding flux values.

left). This means the total electric flux Φ through a closed surface S depends only on the charge Q enclosed by that surface, and does *not* depend on the *shape* of the surface. If the surface is not closed, the integral in Equation 5.7 will not account for all of the flux and the value of Φ will be smaller. Since the integral in Equation 5.7 can also be computed for open surfaces, it becomes a good measure of how much a surface wraps around a charge. This is illustrated in Figure 5.8 showing the flux through a closed and an open surface. Note also, that the dot product with the surface normal in Equation 5.7 accounts for the correct net flux even when a field line penetrates the surface multiple times.

Examples of different configurations in which a deformable object is surrounding the reference object are shown together with the flux value in Figure 5.9. It can be observed that the more the object is surrounded, the larger the flux is. For polygonal meshes, the integral in Equation 5.7 can be computed by summing the flux through all the triangles in the mesh for which an analytical expression exists (Van Oosterom and Strackee [1983]).

We can also compute the electric fields (and corresponding potentials) due to charged objects of arbitrary geometry and topology as long as they are represented by polygonal meshes. Since the analytical expressions for the field and potential around a charged triangle (or rectangle) are known (see Goto et al. [1992]) this can be done using the superposition principle, by summation over all polygons in a mesh:

$$\vec{E}(\vec{x})_{\text{total}} = \sum_{i=1}^{n} \vec{E}_i(\vec{x}, q_i), \text{ and } V(\vec{x})_{\text{total}} = \sum_{i=1}^{n} V_i(\vec{x}, q_i), \tag{5.8}$$

where $\vec{E}_i(\vec{x}, q_i), V_i(\vec{x}, q_i)$ are the analytical forms of the electric field and potential due to the i-th triangle carrying charge q_i, $\vec{x}$ is any desired point in space, and n is the number of elements composing the charged object.

5.5.3.2 Electric Coordinates

Assume there is a *harmonic* potential field around the reference object with a potential of 1 volt everywhere on the surface of the object and vanishing at infinity. Then every point in the 3D space can be characterized by its potential and the unique gradient line on which it lies.

In computer graphics, harmonic fields have been widely used for applications such as quadrilateral remeshing (Dong et al. [2005]), mesh editing(Au et al. [2007]), and generalized barycentric coordinates (Joshi et al. [2007]).In these works, the fields are computed by solving the Laplace equation on a tessellated domain. This approach is not suitable for our purposes as it requires the tessellation of the entire space (dividing the space into small elements) and solving a large-scale linear system whose dimension is the number of elements. As we need to be able to compute the field everywhere in the 3D open space, the size of the grid can be very large in some cases, depending on the working volume and the geometry of the reference object.

The charge simulation approach (described below) allows us to achieve the same goal by performing computations only on the tessellated boundary of the reference object, which then allows us to compute the electric field and potential *everywhere* in the space analytically using the superposition principle presented in Equation 5.8.

The idea is to compute a distribution of charges on the surface of the object such that the resulting electric potential has the above properties. This is analogous to simulating a charged conductor in which all the charge resides only on the surface, external electric field lines are perpendicular to the surface at the surface, there is no electric field inside, and the conductor surface is an equipotential. By the Uniqueness Theorem, the simulated charges will then completely define the electric field everywhere in space. Without loss of generality, assume that we want the potential to be 1 volt on the surface. Following the approach of Maxwell (Malik [1989]) and using the principle of superposition, and the fact that the potential around a charged object is proportional to the charge, we can construct a dense linear system of n equations, each representing the notion that the potential must be 1 volt at some probe point on the surface of the

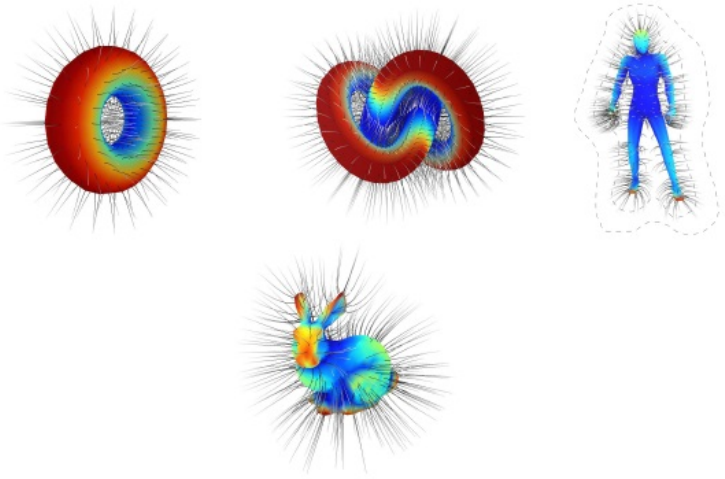

Figure 5.10: The charge distribution (red/blue = more positive/negative charge) and the resulting field lines.

object, with n variables denoting the unknown charges q_i for each of the elements:

$$\begin{cases} V_1(\vec{x}_1)q_1 + V_2(\vec{x}_1)q_2 + \cdots + V_n(\vec{x}_1)q_n = 1 \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \cdots \\ V_1(\vec{x}_n)q_1 + V_2(\vec{x}_n)q_2 + \cdots + V_n(\vec{x}_n)q_n = 1 \end{cases} \tag{5.9}$$

where $V_i(\vec{x}_j)q_i$ denotes the potential V_i at point $\vec{x}_j$ due to the i-th surface element carrying charge q_i in the analytical form Goto et al. [1992]. For the probe points $\vec{x}_j$ we select the barycentres of the mesh triangles (even though any n different points inside or on the surface of the object would do). This is a dense linear system $\mathbf{P}\vec{q} = \vec{1}$, which is typically ill-conditioned, and therefore we solve it using pseudo-inverse of $\mathbf{P}$ in the least squares sense.

Figure 5.10 visualizes the distribution of charges over the surface of different objects and the resulting electric fields around them by plotting field lines $\frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = \vec{E}(\vec{x})$.

When performing the simulation on low-resolution meshes, it is prudent to increase the density of the mesh to more accurately represent the charge distribution, especially in sharp protruding areas.

After the charge simulation, the resulting electric field can be used to parameterize the space. Indeed, the field lines emanating from each point on the surface never intersect (or else the conservation of energy would be violated), and since the potential harmonically decreases with distance from the object, each point along a particular field line has a unique potential (see Figure 5.20 (right)). Each field line is defined

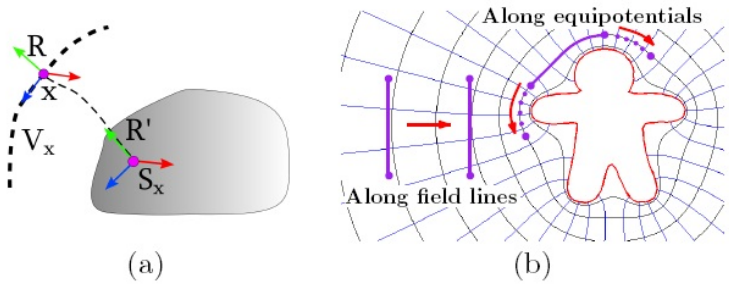

Figure 5.11: (a) A point x in the open space is projected to a point S_x on the surface of the reference object by moving along the field lines. Orientation can also be defined by gradually rotating the coordinate system R along the field lines by SLERP. (b) Driving a wrapping motion by increasing the flux.

by its starting point on the surface (in some parametric (u, v) coordinates). Therefore, each point in the 3D space can be represented by two orthogonal elements: its electric potential (V_x) and the parametric coordinates (u, v) of the origin of the corresponding field line on the surface of the reference object (see Figure 5.11 (a)). This mapping is continuous and invertible. The electric potential V_x is computed using the analytical form (Goto et al. [1992]) in Equation 5.9. The projected point on the reference object $\mathbf{S_x}$ is computed by tracing the field lines from $\mathbf{x}$ back onto the reference object. Field lines, a common way of depicting the electric fields, are the integral curves of

$$\frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = \vec{E}(\vec{x}) \tag{5.10}$$

and so tracing a field line from some starting point $\mathbf{x}$ is a simple initial value problem.

We can also define an orientation at each point in the space with respect to the object using the EC. When projecting an orientable point in the space to the surface of the object, the point can be rotated as it is moved along the field line using SLERP, and the orientation with respect to the surface can be computed when reaching the surface (see $\mathbf{R}$ and $\mathbf{R}'$ in Figure 5.11 (a)).

The EC have several nice features which make them suitable for controlling objects with respect to a reference object. For example, when a sock is to wrap around a foot, we need to make sure that the sock's opening approaches the foot from the toes with the heel pointing downwards. The EC can help in guiding the deformable object in such cases. Although the flux well represents the coverage by the deformable object, in many cases we need to specify the direction from which the deformable object approaches the reference object as well as its orientation at that time.

The parameterization in EC is continuous and invertible, and does not result in any penetrations of the particles of the deformable object into the reference object, as the electric field is zero inside the reference object. Such features make the EC suitable for planning movements for the deformable object to approach and wrap around the reference object, which is explained in Section 5.5.4.

5.5.4 Cloth Manipulation by the Electrostatic Parameters

In this section, we explain how we use the electrostatic parameters to guide wrapping and dressing movements. We first explain the control parameters for wrapping (see Section 5.5.4.1). We next explain the control method for wrapping and dressing manoeuvres (see Section 5.5.4.2). We then illustrate our approach with an example of putting on a sock (see Section 5.5.4.3).

5.5.4.1 Control Parameters for Wrapping

Let us first define the state and control parameters for wrapping. The state of the deformable object is represented by $\mathbf{V} = (\mathbf{v}_1, ..., \mathbf{v}_N)$, where $\mathbf{v}_i$ is the 3D coordinate of the i-th particle and N is the total number of particles. The state of the reference object is represented by $\mathbf{S}$. The reference object can be rigid, articulated or a deformable object controlled by a skeleton. The deformable object and/or the reference object are controlled by the control parameters $\mathbf{c} = (c_1, ..., c_k)$, where k is the dimensionality of the control. These can be positions of some particles or the joint angles of the character putting on a garment.

Given $\mathbf{S}$ and $\mathbf{V}$, we can compute the electrostatic parameters that represent the spatial relationship between the deformable object and the reference object:

$$(\Phi^T, \mathbf{P^T}, \Theta^T)^T = \mathbf{E_{sv}}(\mathbf{S}, \mathbf{V}) \tag{5.11}$$

where Φ is the flux between the deformable object and the reference object, $\mathbf{P}, \Theta$ are the position and the orientation of a reference point of the deformable object in EC, and $\mathbf{E_{sv}}(*,*)$ is the function to compute the electrostatic parameters. The reference point needs to be a point of the deformable object that the system has a good control of. This can be one of the control particles or the barycentre of the opening. As $\mathbf{S}$ and $\mathbf{V}$ are dependent on the control parameters $\mathbf{c}$, Equation 5.11 can be rewritten as

$$(\Phi^T, \mathbf{P^T}, \Theta^T)^T = \mathbf{E_{sv}}(\mathbf{S}(\mathbf{c}), \mathbf{V}(\mathbf{c})) = \mathbf{E_c}(\mathbf{c}) \tag{5.12}$$

We assume Equation 5.12 can be linearised for small updates of $\mathbf{c}$:

$$\begin{pmatrix} \Delta\Phi \\ \Delta\boldsymbol{P} \\ \Delta\Theta \end{pmatrix} = \begin{pmatrix} \boldsymbol{J}_\Phi \\ \boldsymbol{J}_{\boldsymbol{P}} \\ \boldsymbol{J}_\Theta \end{pmatrix} \Delta\mathbf{c} \tag{5.13}$$

where $\boldsymbol{J}_\Phi$, $\boldsymbol{J}_{\boldsymbol{P}}$, $\boldsymbol{J}_\Theta$ are the Jacobians of Φ, $\boldsymbol{P}$, Θ with respect to $\boldsymbol{c}$.

5.5.4.2 Guiding Wrapping Movements

Given the current and goal state of the deformable and reference objects, the system provides intermediate reference values of the electrostatic parameters, and then the system updates the state configuration through $\mathbf{c}$ by solving a least squares problem. The intermediate reference values for the EC are computed by interpolation (see Section 5.5.4.3) and the values for the flux are forced to increment or decrement depending on the manoeuvre.

Given the reference electrostatic parameters at each frame, which are represented by $(\Phi_d, \mathbf{P_d}, \Theta_d)$, the corresponding update in the control parameters are computed by solving the following least square problem:

$$\arg\min_{\Delta\boldsymbol{c}} \|\Delta\boldsymbol{c}\|^2 + \alpha\|\Delta\Phi_d - \boldsymbol{J}_\Phi\Delta\mathbf{c}\|^2 + \beta\|\Delta\boldsymbol{P}_d - \boldsymbol{J}_{\boldsymbol{p}}\Delta\boldsymbol{c}\|^2 + \gamma\|\Delta\Theta_d - \boldsymbol{J}_\Theta\Delta\boldsymbol{c}\|^2, \tag{5.14}$$

where the first term serves to minimize the updates of the control parameters for increasing the stability, the next three terms serve to minimize the difference of the desired electrostatic parameters and their actual values, and α, β, γ are constant weights. Solving Equation 5.14 for $\Delta\mathbf{c}$ can be achieved by solving the following linear problem:

$$\left(\mathbf{I} + \alpha\boldsymbol{J}_\Phi^T\boldsymbol{J}_\Phi + \beta\boldsymbol{J}_{\boldsymbol{p}}^T\boldsymbol{J}_{\boldsymbol{p}} + \gamma\boldsymbol{J}_\Theta^T\boldsymbol{J}_\Theta\right)\Delta\mathbf{c} = \alpha\boldsymbol{J}_\Phi^T\Delta\Phi_d + \beta\boldsymbol{J}_{\boldsymbol{p}}^T\Delta\boldsymbol{P}_d + \gamma\boldsymbol{J}_\Theta^T\Delta\Theta_d \tag{5.15}$$

After computing $\Delta\mathbf{c}$, the control parameters $\mathbf{c}$ are updated by $\mathbf{c} := \mathbf{c} + \Delta\mathbf{c}$. Then, a physical simulation is run using a generic simulator for a single step while constraining $\mathbf{c}$, and the state of the whole system is updated. Our approach here is assuming static physics: we found this approach works fine in our examples in which the deformable objects are not controlled very fast.

By iteratively solving this problem, the deformable object is guided to orient itself and wrap or unwrap around the reference object. If the position and orientation of the deformable object do not have to be controlled, such as when simply putting an object into a bag, we can simply use the flux term for reference and neglect the direction and orientation terms.

Note that the flux depends only on the boundary of the surface, and not on the rest of the surface. This is in accordance with Stokes' theorem. This allows us to drastically reduce the dimensionality of the control as we only need to actively control the points on the boundary of the surface. The rest of the cloth is controlled by the physical simulator.

To intuitively understand driving a cloth in a wrapping manoeuvre by increasing the flux through the cloth, decompose the motion into two orthogonal components: (1) moving the cloth perpendicular to the field lines closer to the object, in which case increasingly more field lines will penetrate the same area of the cloth; (2) expanding the cloth along the equipotential surfaces perpendicular to the field lines. This is illustrated in Figure 5.11 (b).

5.5.4.3 Example of Control in Electric Coordinates

Here we provide an example of putting on a sock using electrostatic parameters. The sock needs to be put on from the tip of the toes, and the heel part must be facing downwards. This is guided by specifying the EC of the barycentre of the sock opening, whose position and orientation is defined by x_s and θ_s respectively. At the initial configuration, the barycentre of the sock opening, x_s^0, is projected onto the surface of the foot at point S, using the computed electric fields (see Figure 5.12). A geodesic curve is then found from S to a point T, which lies in the centre of the toes. Let us parameterize the geodesic curve by $g(t)$, where $g(0) = S$ and $g(1) = T$. We also compute the electric potential of the barycentre of the opening, which is defined by V_0 here. The potential at the toes is assumed to be 1 volt.

In each iteration, we compute the desired EC of the barycentre of the sock opening by linearly interpolating the EC at the initial configuration and the configuration at the toes when the sock starts to wrap the foot. We first compute the electric potential value by linear interpolation: $V(t) = V_0 + (1 - V_0)t$. Next, starting from the interpolated point on the geodesic curve $g(t)$, we move along the gradient of the electric field outwards in the open space, until we reach the equipotential surface of $V(t)$. This point P_d becomes the desired position of barycentre of the sock opening.

We also compute the desired orientation around the field line at P_d by linear interpolation in the EC. We use two representative particles of the opening and the opening's barycentre to define the orientation. We compute the orientation at $t = 0$ and $t = 1$, and then compute the desired orientation Θ_d at each frame by linearly interpolating them.

We then proceed to increase the desired flux to guide the complex motion of the

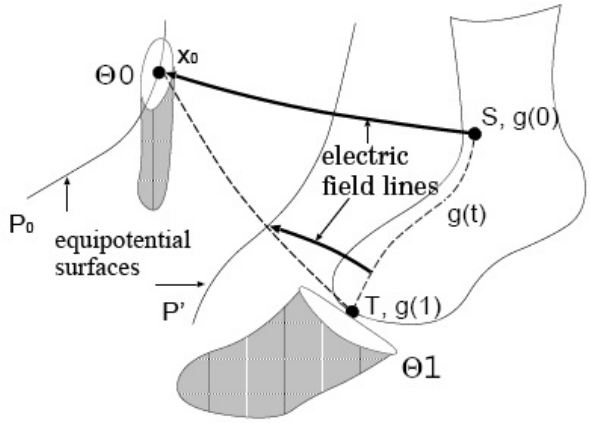

Figure 5.12: The sock's opening is brought to the tip of the toes by interpolating the generalized polar coordinates.

sock in a dressing manoeuvre.

When putting on garments, sometimes it is necessary to specify the part of the garment which is to wrap around each part of the body. For example, when putting on trousers, each trouser leg must be covering the corresponding limb. We first split the garment into different regions based on the branches. Then, the whole dressing process is subdivided into a sequence of subtasks, such as moving the leg towards the left trouser leg opening, increasing the flux between the two while stepping down, doing the same for the right leg, and finally pulling up the trousers by increasing the flux between the whole garment and the body.

5.5.5 Experimental Results

We now explain how to apply our approach to synthesize animations of wrapping a bouquet of flowers in a sheet of paper, a fish character following and eating prey, and a human character dressing or undressing. The readers are referred to the supplementary video for further details. At the end, we discuss the complexity and computational costs.

Wrapping a bouquet of flowers: A sheet is automatically controlled to wrap it around a bouquet of flowers. Starting from a configuration in which the stems are on the sheet, the sheet is wrapped around the stems by gradually increasing the flux value. The sheet nicely wraps around the stems (see Figure 5.13, 1st row).

Wrapping an object by a bag: A bag is automatically controlled to wrap around

Figure 5.13: Wrapping a bouquet.

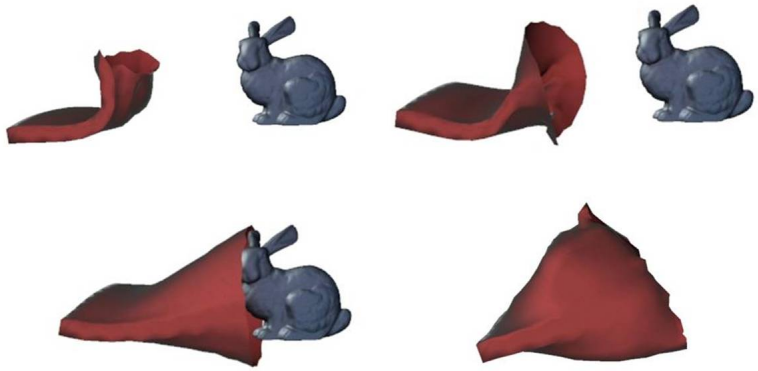

Figure 5.14: An initially folded bag wrapping a standford bunny.

different polygonal models including the Stanford Bunny (Figure 5.14), a trefoil knot (Figure 5.15) and the Armadillo Man (Figure 5.17),. Even in the case where the bag starts from a folded state, it automatically unfolds, directs its opening towards the reference object, and wraps around it while adapting the shape of the opening to the geometry of the object. Such control is difficult by existing methods without global path planning.

We also produced a similar example in which a fish character is controlled such that the flux due to the charged prey through the fish is increased (see Figure 5.17, 3rd row).

Taking off garments: We let the character grasp some specific part of the garments and then use the grasped part as the control particles. The location of these particles correspond to $\mathbf{c}$ that is used to guide the manipulation of the garments. The controlled particles are manually specified. The character then moves its hands in the direction that decreases the flux. This guides the movements to take off the garments. We show examples of a mother character taking the sweater off a child character. In these exam-

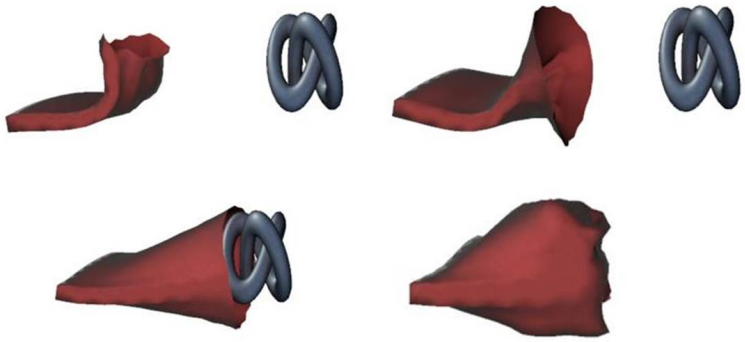

Figure 5.15: An initially folded bag wrapping a trefoil knot.

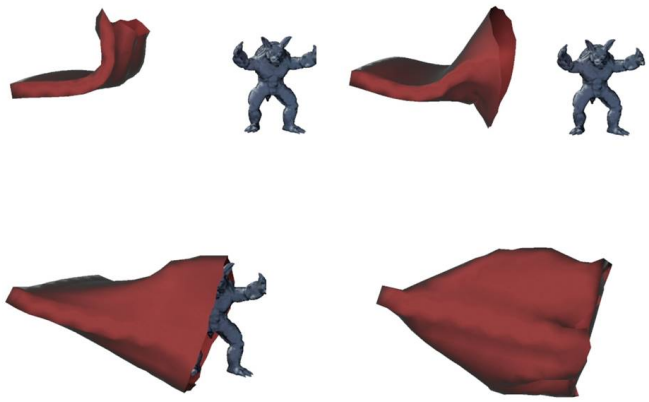

Figure 5.16: An initially folded bag wrapping an armadillo.

ples, *only the flux values* are used to guide the movements. Three different examples in which the child character raises its arms in different directions are produced (see Figure 5.18, 4th row). The mother character successfully takes off the shirt in all these examples by moving the shirt along the tangent surface of the body, which is the most efficient way to decrease the flux.

Putting on garments: Putting on garments requires more complicated control as the direction and orientation also have to be controlled in addition to the flux in order to correctly guide the character to conduct the motion.

In the first example, we show an example of putting on a sock using the approach explained in Section 5.5.4.3. We specify the sock's initial configuration and the position/orientation of the opening's barycentre when the toes are about to pass the opening. Two sets of particles of the sock's opening, which correspond to the area that is

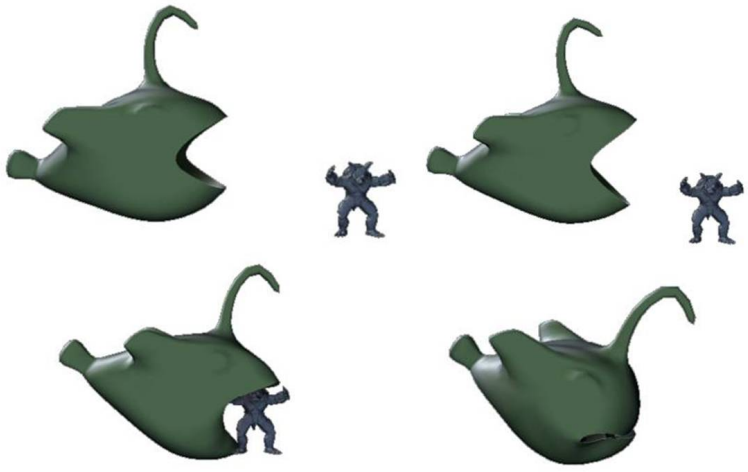

Figure 5.17: A fish wrapping an armadillo.

grasped by the fingers, are defined as control particles. The movement of the opening's barycentre is computed by linearly interpolating its initial position/orientation and those when the toes passing the opening, in the EC. Then, the movements of the control particles are computed at each frame by solving Equation 5.15. Finally, the arm movements of the character controlling the sock are computed by inverse kinematics (see Figure 5.19 and Figure 5.6).

Next, we show an example of putting on shorts also as explained in Section 5.5.4.3. The procedure is split into three steps: stepping the right foot into the shorts, stepping the left foot into the shorts, and finally pulling them up by increasing the flux with respect to the whole body (see Figure 5.6). As the body is moving in this example, the electric field is recomputed per frame.

Complexity and Computational Costs The complexity for the charge simulation (Equation 5.9), flux computation (Equation 5.7) are state update(Equation 5.15) are is $O(n^3)$, $O(n \times m)$ and $O(k)$, respectively, where n is the number of triangles composing the reference object, m is that of the deformable object, and k is the number of control parameters. The charge simulation is costly as the coefficient matrix in Equation 5.9 is dense. However, this needs to be done only once for static reference objects, and for deforming objects, its computation can be significantly accelerated by using the charges in the previous step as the initial guess for the conjugate gradient solver in the next step. The computation for time required for the sweater, pants and bag (with the Armadillo Man) demo using one core of a Core i7 2.67GHz CPU are shown in Table 5.1. We use UMFPACK (Davis [2004b]) with GotoBLAS (Goto and Van De Geijn

[2008]) for solving the sparse linear problem in Equation 5.15.

We use PhysX 2.84 (NVidia [2008]) for the physical simulation.

5.5.6 Discussion

The features of our electrostatic parameterization include: (1) **Abstractness**: The flux is an abstract parameter which well describes the spatial relationship of the deformable object and the reference object. Using flux allows us to disregard unimportant geometric detail and summarise the semantically important degree of wrapping around in a single quantity. Flux is insensitive to the local geometry of the curves and surfaces. As a result, the same control strategy can be applied for reference objects of different geometries and postures. (2) **Object-centric parameterization**: The space is parameterized with respect to the characters and objects by a harmonic field that does not include any local minima.

These features allow the method to successfully guide the movements of deformable object without using any global path planning algorithm.

One alternative to compute the coverage and parameterize the outer space is to use the distance field. Igarashi et al. [2009] evaluate the coverage by testing if the normal vectors intersect the deformable objects, which is similar to using a distance field. Schmid et al. [2011] propose a modelling tool for designing on the outer surface of closed surfaces. The distance field is not suitable for our purpose as it cannot handle concave objects and multiple objects well. Singular points/lines/surfaces are going to be created, which can result in non-smooth movements when the particles cross the singular area (see Figure 5.20). As the field is not harmonic, we cannot benefit from the local path planning to guide the deformable object.

Limitations As our method is a local path planning method, there are cases that the motion cannot be synthesized by linear interpolation. For example, in case that the

Table 5.1: Data of each example. *def*, *ref*: the triangle number of the deformable and reference object, *charge*, *jac*, *lin*: time for charge simulation, calculating the Jacobians, and solving the linear problem (in *ms*). *The charge simulation for the pants after the first frame is 0.005 *ms*.

Scene	*def*	*ref*	*charge*	*jac*	*lin*
sweater	2176	288	0.149	0.076	0.007
pants	798	288	0.138*	0.078	0.003
bag	868	303	0.165	0.021	0.003

deformable object is knotted at the opening (or at other parts), it is not possible to unknot the configurations; a knot-based energy (H.Freedman et al. [1994]) is required for guiding the unknotting process in such a case. As the dressing/undressing process sometimes results in squeezing the cloth, it will require a precise collision detection and handling for improving the visual quality of the results. In such case, the progress of the simulation can become slow. Although the abstract, shape insensitive nature of coverage increases its applicability, sometimes it can be a disadvantage as the details cannot be controlled.

In such a case, additional positional constraints can be given by the user for finer control.

5.5.7 Conclusion and Future Work

In this chapter, we have proposed a new approach based on electrostatics to synthesize scenes in which deformable objects closely interact with other characters and objects. We have defined a new state space using a value called flux and Electric Coordinates. The flux is an abstract parameter that evaluates how much the reference object is surrounded by the deformable object. The EC provide a continuous parameterization of the open space around the reference object, which helps to guide movements of the deformable object.

The body-centric EC have the potential to be applied for other purposes, such as layered representation of deformable objects. Layered representations (Igarashi and Mitani [2010]; McCann and Pollard [2009]), which are used to discretise the relationship of multiple, deformable objects, are currently only applicable for 2D instances as the layer order is defined by the height vector. The electric potential can serve as the distance value to determine the order of the layers in 3D with respect to the reference object. Such a usage can be useful for collision resolving in cloth animation.

We have shown that the approach is useful for animations of wrapping and putting on garments. One problem of the method is that for complex behaviours such as putting on garments, the user needs to provide a sequence of different manoeuvres; for example, for putting on trousers, the left and right limbs need to be passed through the corresponding trouser legs first, and then pulled up to the waist. Automatically computing such sequences from the final configuration would be an interesting research direction.

Another interesting direction is to apply the method for robotics. Given models

of the reference and deformable object, it is possible to estimate their state based on the vision data. We can then compute the electrostatic parameters and estimate the relationship of the two. In order to robustly estimate the state of the relationships, it will be necessary to use statistical approaches, which is an interesting direction to pursue.

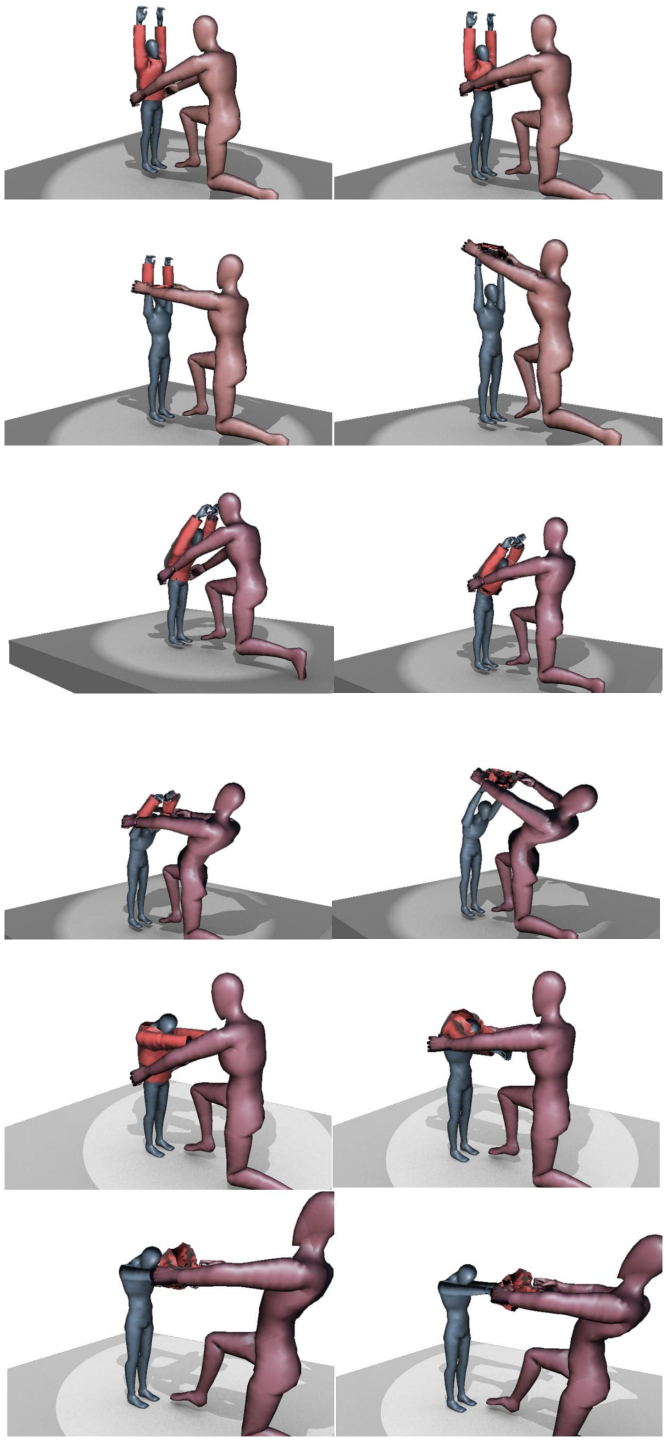

Figure 5.18: A mom taking off the t-shirt from a child. Different postures show that the algorithm automatically adapts to the geometry of the target object.

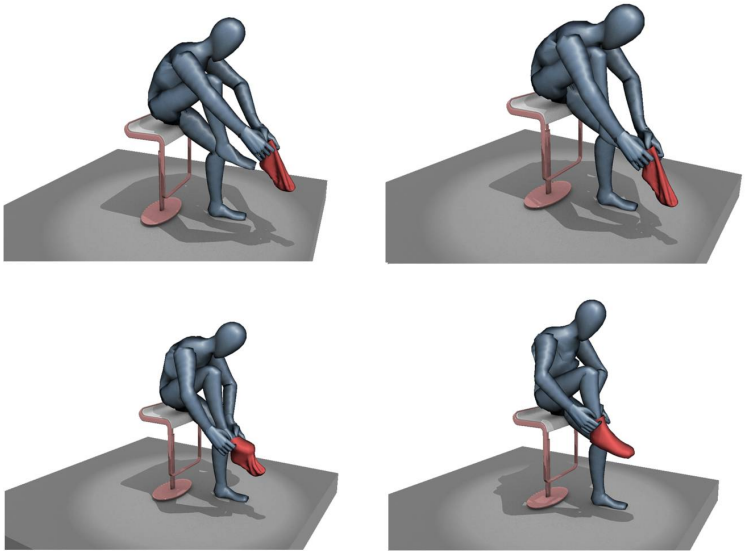

Figure 5.19: Putting on a sock. The sock is first positioned near the toes by control illustrated in Figure 5.12

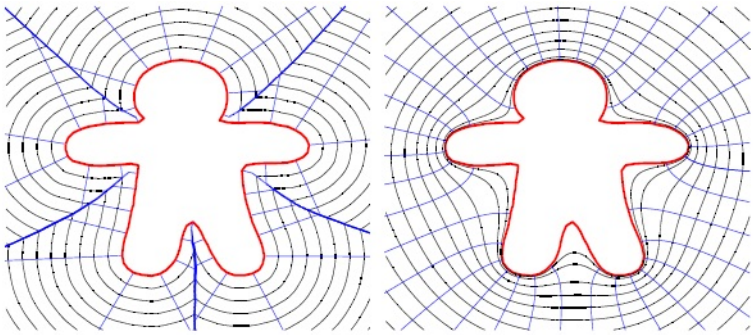

Figure 5.20: *Left:* the isosurfaces and the field lines of the distance field. *Right:* the grid (field lines and equipotentials) of our curvilinear Electric Coordinates system.

Chapter 6

Conclusions, discussions and future work

6.1 Contributions

In this thesis, first I showed an energy-based representation equipped with a local planning algorithm for arbitrary posture interpolation. It represents postures by capturing the spatial relationship and potential penetrations between body parts. The formulation of the energy function guarantees there is a low energy region (the region near the canonical configuration) where penetrations are less likely to happen so that linear interpolation can be employed. By moving given postures towards the low energy region, the postures can always be interpolated. In addition, since the global structure of the energy manifold is concisely described by the function, the motion planning can be done by a local method.

Second, I proposed a simple control strategy called Geodesic Control and a model that uses Finite State Machines for tasks such as wrapping and knotting. Based on that, different furoshiki wraps were produced. The essence of Geodesic Control is gradually increasing the contact area of two geodesic lines in a pre-defined direction by user. The robustness of the Geodesic Control and the easiness of control make the method ideal for generating natural-looking winding and wrapping movements. The abstractness and simplicity of this control make it possible to model operations as independent states and employ Finite State Machines to sequence them for higher-level tasks like furoshiki wraps.

Finally, I presented two models with their respective local planning algorithms to quantify free wrapping tasks. Under this setting, the progress of wrapping can be

measured intuitively for scenarios with target objects that are static or dynamic, rigid or deformable. Motions of the wrapper (sock, bag and t-shirt) can be planned via linearly interpolating between the current and target progress values. The measures are independent of geometry and topology.

6.2 The bigger picture

What idea is behind all of the methods suggested in this thesis? To answer that question, I would like to take one step back and look back onto the research framework raised in Figure 1.1, in Section 1.2. For all the problems in this research project, two primary difficulties are: the curse of dimensionality (Bellman [1961, 2003]) and the lack of good representations of complex tasks. Intuitively, the corresponding solutions are a reduction of the dimensionality and new task-related representations. This logic is the central line through all of our solutions. And the research diagram in Figure 1.1 forms up the framework of the methodology. The bi-mapping explained in Section 1.2 bridges two kinds of manifolds: object-centric manifolds and task-centric manifolds. The forward mapping is where we reduce the dimensionality and the backward mapping is where we map the solution back to the original object-centric manifold. The bi-mapping mitigates the curse of dimensionality. The task-centric manifold, which is a new and quantitative representation of the task, solves the problem of lacking representations of complex tasks.

Bearing the general framework in mind, to solve specific problems, three key elements are needed and they were mentioned in Section 1.2: **task-related representations**, **governing properties** and **local control algorithms**. I will briefly go through the methodology again by explaining how our methods provide the key elements of the framework in each of the task.

In the arbitrary posture interpolation problem, as the task-centric manifold can be attained by adding one scalar onto every point of the object-centric manifold, the forward and backward mapping are trivial. We constructed the task-centric manifold by defining the energy function. Together with the canonical configuration, it gives the global structure of this manifold: a convex manifold where a low energy region near the global minimum (canonical configuration) is safe for interpolation. For the three key elements, this energy function implicitly gives the task-related representation by reinterpreting the interpolation task as: moving the postures into the safe region then linearly interpolate them. It also explicitly gives a governing property, the energy,

which for our purpose just needs to monotonically decrease. Given the global structure of the task-centric manifold and the monotonicity of the governing property, the planning is easily handled by a local control algorithm.

In furoshiki wraps, the global structure of the task-centric manifold is difficult to abstract in general. However, given specific tasks, they can be modelled as Finite State Machines. The states are given by the user and they work as sampled points on this task-centric manifold. More importantly, for each transition, the governing property can be defined as the contact area between the control line and the target line. Here, the forward mapping is done by sampling control lines and target lines which reduce the dimensionality of the problem. And for each transition in the FSMs, the task-centric manifold is governed by the contact area. By gradually increasing the area, not only is the transition carried out, but the quality of the motion is also controlled. Again, since it changes monotonically, the control problem is convex so that local control algorithms can be used. Finally, the physical simulation handles the update, the step 4 in the bi-mapping.

In free wrapping, the task-centric manifold is governed by one property: coverage. In both of the models (Wrapping Control and Electrostatic control), it is invariant of geometry and topology and monotonically increases during wrapping tasks. This representation grasps the key information of the wrapping task. Furthermore, it comes with an intuitive forward mapping that reduces the dimensionality significantly based on the fact that only DoFs at the opening of the wrapper matter. Once again, since the governing property just needs to increase monotonically, local motion control can be used. Finally, the update is handled by a physical simulator.

From the analysis above, one can easily see that the task-centric manifold is the core of the framework. For some tasks, the forward and backward mappings are not necessary. In contrast, for all the tasks, the task-centric manifold is essential. In addition, the three key elements are all tightly associated with the representation of the task-centric manifold.

6.3 Future work

With respect to the future work, there are several directions. In this section I will explain how the current research diagram can be improved in these directions.

6.3.1 Detailed control

A control layer dealing with finer control is a good direction. The core part of this thesis is the task-centric manifold. The construction of this manifold is based on the assumption that there is, at least locally, a governing property which can be monotonically changed to accomplish the task. However, it also means the property is highly abstracted thus loses information of details to some extent. For example, in the sock experiment in Section 5.5, we can control the wrapping and can even specialise the initial configuration. But, the detailed wrapping behaviours are only subject to the progress requirement of the task and physical simulation. There is no mechanism to easily introduce constraints such as controlling the individual wrapping contributions of the degrees of freedom around the opening dynamically. This kind of detailed control requires information with smaller granularities. Generally, there is room for this kind of control. Back to the sock example, when we want the sock to wrap the foot a bit more, there are many solutions, which means there are redundancies. A hierarchical control strategy might be a good approach to handle this problem. It keeps the succinctness of the task-related representation and all the benefits coming with it at the highest level, but locally controls more by exploring redundant dimensions.

6.3.2 Working with higher-level planning methods

Our methods can improve higher-level planning methods in two ways: (1) providing a good initial guess. (2) Working as a filter. Although our methods can find out a good solution quickly, thanks to the task-related representations and local planning algorithms, the solution might not be able to satisfy all the criteria due to the fact that the planning is mainly done based on the governing property. However, the solution can serve as a good initial guess for methods such as spacetime optimisation. For example, when doing the arbitrary posture interpolation, our method can work out a penetration-free solution first, and then the spacetime optimisation can be introduced to incorporate other constraints. This kind of improvement is crucial for methods that require initial solutions to be given either explicitly or implicitly.

The other way to improve high-level planning methods is filtering out samples that do not satisfy task-related requirements. In our experiments, we do not need this since we use local control algorithms. However, for other tasks where the constraints are different, this task-related representation can be used as a filter for global planning algorithms such as Probabilistic Roadmaps and Rapidly-exploring Random Trees. The

filter can help to boost the performance of the sampling of the space by quickly eliminating samples that do not satisfy the change requirement of the governing property of the task. For an intuitive example, refer to Figure 1.1 again. In the original setting, when we want to move the particle closer to the goal, we move it little by little because we want the particle to move continuously. This is a valid constraint for animation or robotics, but not universal. By setting a range of values indicating the progress to be made, our task-related representation can eliminate all the samples where the particle is either too close to or too far away from the goal. And this redundant subspace can be used by higher-level planning methods for other purposes later.

6.3.3 Completion of a new coordinate system

Electrical Coordinates is a good parameterisation of the space and has the potential to be used for other problems. Given any point outside the conductor (The target object in the free wrapping) in the space, it has a corresponding point, the *anchor point*, on the surface of the conductor and also has a potential value. The anchor point and the potential value uniquely define the position of that point in this space with respect to the conductor. This correspondence is maintained even under deformations of the conductor. Only when the topology of the conductor changes, the corresponding positions can be merged or split. If barycentric Coordinates are universal representations for points inside a body, Electrical Coordinates can be its counterparts for points outside the body. However, more things can be done to fully make use of this coordinate system. One thing is defining the merge and split operations when the topology of the central body changes. Another one is properly defining translations and rotations in this system. In this case, flux lines equipped with local coordinates over the curve can be a good direction. It is like the bishop frame for rods. The translation and rotation can be integrated along the flux line.

Appendix A

Supplementary materials for Furoshiki wrapping

Here we show the sequence of manoeuvres of the FSMs for the two granny-knot box wrapping (Table A.2), wine bottle wrapping (Table A.3) and basket wrapping (Table A.4). The identities of corners, area and control lines are defined as shown in Figure A.1.

For the sake of simplicity, we first define the granny-knot manoeuvre which we used for all our demos. The manoeuvre takes two control lines and the knotting position as input, and produces a granny knot.

Table A.1: The manoeuvres and attributes of GrannyKnot

	manoeuvre	***attributes***
1	Crossing	control line C and T, top of object center
2	Winding	control line C around T, top of object center, negative
3	Tightening	winding made at step 2, towards center
4	Crossing	control line T and C, top of winding made at step 3
5	Winding	control line T around C, top of winding made at step 3, negative
6	Tightening	winding made at step 4, towards winding made at step 5

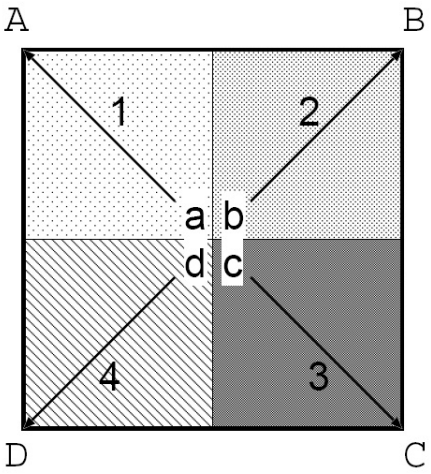

Figure A.1: The identities of the corners, area and control lines defined for our experimental results.

Table A.2: The manoeuvres and attributes of two granny-knot box wrapping

	manoeuvre	***attributes***
1	Anchoring	centre
2	Wrapping	corner A and C, area a and c
3	GrannyKnot	control line 1 and 3,top of object centre
4	Wrapping	corner B and D, area b and d.
5	GrannyKnot	control line 2 and 4, top of knot made at step 3

Table A.3: The manoeuvres and attributes of wine bottle wrapping box wrapping

	manoeuvre	***attributes***
1	Anchoring	centre
2	Wrapping	corner A,C and area a,c
3	GrannyKnot	control line 1 and 3, top of object center
4	Wrapping	corner B and area b
5	Wrapping	corner D, and area d
6	GrannyKnot	control line 2 and 4, front of object

Table A.4: The manoeuvres and attributes of a basket wrapping.

	manoeuvre	***attributes***
1	Anchoring	corner A
2	Wrapping	corner A,C and area a,c
3	GrannyKnot	control line 1,3, top of bottom half of basket
4	Winding	control line 2, basket handle, negative
5	Winding	control line 4, basket handle, negative
6	GrannyKnot	control line 2,4, top of basket handle

Bibliography

Yeuhi Abe, C. Karen Liu, and Zoran Popović. Momentum-based parameterization of dynamic character motion. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 173–182, Grenoble, France, 2004. Eurographics Association. ISBN 3-905673-14-2. doi: 10.1145/1028523.1028546. URL http://portal.acm.org/citation.cfm?id=1028523.1028546&coll=GUIDE&dl=GUIDE&CFID=34639582&CFTOKEN=49101429. 14

Jürgen Acker, Dominik Henrich, Lehrstuhl Angewandte, and Informatik Iii. Manipulation of deformable linear object; from geometric model towards program generation. *PROC. IEEE INT. CONF. ROBOTICS AND AUTOMATION*, pages 1553—1559, 2005. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.149.8001. 6, 30

Aichholzer Aichholzer, Cortés Cortés, Demaine Demaine, Dujmovic Dujmovic, Erickson Erickson, Meijer Meijer, Overmars Overmars, Palop Palop, Ramaswami Ramaswami, and Toussaint Toussaint. Flipturning polygons. *Discrete & Computational Geometry*, 28(2):231–253, July 2002. ISSN 0179-5376. doi: 10.1007/s00454-002-2775-7. URL http://www.springerlink.com/content/hbw9ejxrj1hyg5nd/abstract/. 26

O. M Al-Jarrah and Y. F Zheng. Intelligent compliant motion control. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 28(1):116–122, February 1998. ISSN 1083-4419. doi: 10.1109/3477.658590. 29

Helmut Alt, Christian Knauer, G. Rote, and Sue Whitesides. On the complexity of the linkage reconfiguration problem. *Towards a Theory of Geometric Graphs*, 342:1–14, 2003. 27

Byoungkwon An, Nadia Benbernou, Erik D. Demaine, and Daniela Rus. Planning to

fold multiple objects from a single Self-Folding sheet. *Robotica*, 29(Special Issue 01):87–102, 2011. doi: 10.1017/S0263574710000731. 32

Kiam Heong Ang, G. Chong, and Yun Li. PID control system analysis, design, and technology. *IEEE Transactions on Control Systems Technology*, 13(4):559– 576, July 2005. ISSN 1063-6536. doi: 10.1109/TCST.2005.847331. 12

Oscar Kin-Chung Au, Hongbo Fu, Chiew-Lan Tai, and Daniel Cohen-Or. Handle-aware isolines for scalable shape editing. *ACM Trans Graph*, 26(3), 2007. 96

Paolo Baerlocher and Ronan Boulic. An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. *Vis. Comput.*, 20(6):402–417, 2004. URL http://portal.acm.org/citation.cfm?id=1016493. 11

Devin J. Balkcom and Matthew T. Mason. Robotic origami folding. *The International Journal of Robotics Research*, 27(5):613 –627, May 2008. doi: 10.1177/0278364908090235. URL http://ijr.sagepub.com/content/27/5/613.abstract. ix, 33

David Baraff and Andrew Witkin. Large steps in cloth simulation. *Computer Graphics (Proc of SIGGRAPH)*, page 43–54, 1998. 1, 12, 40, 42, 80

Julien Basch, Leonidas J Guibas, and John Hershberger. Data structures for mobile data. *JOURNAL OF ALGORITHMS*, 31:747—756, 1997. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.134.6921. 42

K.J. Bathe. *Finite Element Procedures*. Klaus-Jurgen Bathe, February 2007. ISBN 097900490X. 35

Richard Ernest Bellman. *Adaptive control processes: a guided tour*. Princeton University Press, 1961. 2, 112

Richard Ernest Bellman. *Dynamic Programming*. Courier Dover Publications, March 2003. ISBN 9780486428093. 2, 112

Mitchell A Berger and Chris Prior. The writhe of open and closed curves. *Journal of Physics A: Mathematical and General*, 39(26), 2006. 65

Miklos Bergou, Saurabh Mathur, Max Wardetzky, and Eitan Grinspun. TRACKS: toward directable thin shells. *ACM Trans. Graph.*, 26(3), 2007. 80

D. Bertram, J. Kuffner, R. Dillmann, and T. Asfour. An integrated approach to inverse kinematics and path planning for redundant manipulators. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*, pages 1874–1879. IEEE, May 2006. ISBN 0-7803-9505-0. doi: 10.1109/ROBOT.2006.1641979. 29

Subhrajit Bhattacharya, Maxim Likhachev, and Vijay Kumar. Identification and representation of homotopy classes of trajectories for search-based path planning in 3D. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011. 76

Gerald Bianchi, Barbara Solenthaler, Gabor Szekely, and Matthias Harders. Simultaneous topology and stiffness identification for Mass-Spring models based on FEM reference deformations. *IN MICCAI (2*, 2004:293—301, 2004. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.61.6341. 40

Therese Biedl, Erik Demaine, Martin Demaine, Sylvain Lazard, Anna Lubiw, Joseph O'Rourke, Steve Robbins, Ileana Streinu, Godfried Toussaint, and Sue Whitesides. A note on reconfiguring tree linkages: Trees can lock. *Discrete Applied Mathematics*, 117(1-3):293–297, 2002. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.27.4274. viii, 25

Therese C. Biedl, Erik D. Demaine, Martin L. Demaine, Sylvain Lazard, Anna Lubiw, Joseph O'Rourke, Mark H. Overmars, Steve Robbins, Ileana Streinu, Godfried T. Toussaint, and Sue Whitesides. Locked and unlocked polygonal chains in three dimensions. *Discrete & Computational Geometry*, pages 269–281, 2001. ix, 27, 28

Prosenjit Bose, Francisco Gómez, Pedro Ramos, and Godfried Toussaint. Drawing nice projections of objects in space. *Journal of Visual Communication and Image Representation*, 10(2):155–172, June 1999. ISSN 1047-3203. doi: 10.1006/jvci.1999.0415. URL http://www.sciencedirect.com/science/article/pii/S1047320399904157. 28

Eddy Boxerman and Uri Ascher. Decomposing cloth. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, page 153–161, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association. ISBN 3-905673-14-2. doi: 10.1145/1028523.1028543. URL http://dx.doi.org/10.1145/1028523.1028543. 42

R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '03, page 28–36, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. ISBN 1-58113-659-5. URL http://dl.acm.org/citation.cfm?id=846276.846281. 1, 40, 42

Joel Brown, Jean-Claude Latombe, and Kevin Montgomery. Real-time knot-tying simulation. *Vis. Comput.*, 20(2):165–179, May 2004a. ISSN 0178-2789. doi: 10.1007/s00371-003-0226-y. URL http://dl.acm.org/citation.cfm?id=1008158.1008159. 30, 76

Joel Brown, Jean-Claude Latombe, and Kevin Montgomery. Real-time knot tying simulation,. *The Visual Computer*, 20(2-3):165–179, 2004b. 1

Armin Bruderlin and Lance Williams. Motion signal processing. *Computer Graphics (Proc of SIGGRAPH 95)*, 29:97–104, 1995. 17

Jorge Alberto Calvo, Danny Krizanc, Pat Morin, Michael Soss, and Godfried Toussaint. Convexifying polygons with simple projections. *Information Processing Letters*, 80(2):81–86, October 2001. ISSN 0020-0190. doi: 10.1016/S0020-0190(01)00150-8. URL http://www.sciencedirect.com/science/article/pii/S0020019001001508. 28

Jason Cantarella and Heather Johnston. Nontrivial embeddings of polygonal intervals and unknots in 3-space. *Journal of Knot Theory Ramifications*, 7:1027–1039, 1998. ix, 28

Jason H. Cantarella, Erik D. Demaine, Hayley N. Iben, and James F. O'Brien. An energy-driven approach to linkage unfolding. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 134–143, Brooklyn, New York, USA, 2004. ACM. ISBN 1-58113-885-7. doi: 10.1145/997817.997840. URL http://portal.acm.org/citation.cfm?id=997840. 25, 45, 47, 49

Kwang-Jin Choi and Hyeong-Seok Ko. Stable but responsive cloth. *ACM Trans. Graph.*, 21(3):604–611, July 2002. ISSN 0730-0301. doi: 10.1145/566654.566624. URL http://doi.acm.org/10.1145/566654.566624. 42

Min Gyu Choi, Jehee Lee, and Sung Yong Shin. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Trans. Graph.*, 22(2):

182–203, April 2003. ISSN 0730-0301. doi: 10.1145/636886.636889. URL http://doi.acm.org/10.1145/636886.636889. 19

Fehmi Cirak and Matthew West. Decomposition contact response (DCR) for explicit finite element dynamics. *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING*, 64, 2005. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.61.8900. 42

Michael B. Cline, KangKang Yin, and Dinesh K. Pai. Motion perturbation based on simple neuromotor control models. Technical report, University of British Columbia, 2002. URL http://portal.acm.org/citation.cfm?id=902071&coll=GUIDE&dl=GUIDE&CFID=34639582&CFTOKEN=49101429. 12

Michael F. Cohen, John Wallace, and Pat Hanrahan. *Radiosity and realistic image synthesis*. Academic Press Professional, Inc., San Diego, CA, USA, 1993. ISBN 0-12-178270-0. 84

Robert Connelly, Erik D Demaine, and Günter Rote. Straightening polygonal arcs and convexifying polygonal cycles. *DISCRETE & COMPUTATIONAL GEOMETRY*, 30:432—442, 2000. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.103.7461. 24

Robert Connelly, Erik D Demaine, and Günter Rote. Infinitesimally locked self-touching linkages with applications to locked trees. *PHYSICAL KNOTS: KNOTTING, LINKING, AND FOLDING OF GEOMETRIC OBJECTS IN 3-SPACE*, 2002: 287—311, 2002. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.116.753. viii, 25, 26

Stelian Coros, Philippe Beaudoin, KangKang Yin, and Michiel van de Panne. Synthesis of constrained walking skills. *ACM Trans. Graph. (Proc. Siggraph Asia)*, 27(5), 2008. 12

Franklin C. Crow. Shadow algorithms for computer graphics. In *Computer Graphics (SIGGRAPH '77 Proc)*, volume 11, page 242–248, July 1977. 83

Sean Curtis, Rasmus Tamstorf, and Dinesh Manocha. Fast collision detection for deformable models using representative-triangles. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pages 61–69, Redwood City, California, 2008. ACM. ISBN 978-1-59593-983-8. doi: 10.1145/1342250.1342260. URL http://portal.acm.org/citation.cfm?id=1342250.1342260. 80

M. da Silva, Abe Y., and J. Popović. Simulation of human motion data using Short-Horizon Model-Predictive control. *Computer Graphics Forum*, 27(2):371—380, 2008. 14

T. A. Davis. Algorithm 832: UMFPACK, an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30(2 (Jun.)):196–199, 2004a. 90

T. A. Davis. Algorithm 832: UMFPACK, an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30(2 (Jun.)):196–199, 2004b. 105

Edilson de Aguiar, Leonid Sigal, Adrien Treuille, and Jessica K. Hodgins. Stable spaces for real-time clothing. *ACM Trans. Graph.*, 29(4):106:1–106:9, July 2010. ISSN 0730-0301. doi: 10.1145/1778765.1778843. URL http://doi.acm.org/10.1145/1778765.1778843. 41, 80

M. de Lasa and A. Hertzmann. Prioritized optimization for task-space control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009*, pages 5755–5762. IEEE, October 2009. ISBN 978-1-4244-3803-7. doi: 10.1109/IROS.2009.5354341. 14

Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. Feature-based locomotion controllers. *ACM Transactions on Graphics*, 29:1, July 2010. ISSN 07300301. doi: 10.1145/1778765.1781157. URL http://portal.acm.org/citation.cfm?doid=1778765.1781157. 14

G. Debunne, M. Desbrun, M. -P Cani, and A. Barr. Adaptive simulation of soft bodies in real-time. In *Computer Animation 2000. Proceedings*, pages 15–20. IEEE, 2000. ISBN 0-7695-0683-6. doi: 10.1109/CA.2000.889022. 37

Gilles Debunne, Mathieu Desbrun, Alan Barr, and Marie-paule Cani. Interactive multiresolution animation of deformable models. *Eurographics Workshop on Computer Animation and Simulation*, September 1999. URL http://maverick.inria.fr/Publications/1999/DDBC99. 37

Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H. Barr. Dynamic real-time deformations using space & time adaptive sampling. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, page 31–36, New York, NY, USA, 2001. ACM. ISBN 1-58113-374-X. doi: 10.1145/383259.383262. URL http://doi.acm.org/10.1145/383259.383262. 37

Erik D Demaine and Joseph O'Rourke. A survey of folding and unfolding in computational geometry. In *Combinatorial and Computational Geometry*, volume 52 of *Mathematical Sciences Research Institute Publications*, pages 167–211. Cambridge University Press, August 2005. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.136.7449. viii, 23, 24

Erik D. Demaine and Joseph O'Rourke. *Geometric Folding Algorithms: Linkages, Origami, Polyhedra*. Cambridge University Press, New York, NY, USA, reprint edition, 2008. ISBN 9780521715225. 32

Erik D. Demaine, Martin L. Demaine, John Iacono, and Stefan Langerman. Wrapping spheres with flat paper. *Comput. Geom. Theory Appl.*, 42(8):748–757, October 2009. ISSN 0925-7721. doi: 10.1016/j.comgeo.2008.10.006. URL http://dl.acm.org/citation.cfm?id=1540674.1541290. 32

Mathieu Desbrun, Peter Schröder, and Alan Barr. Interactive animation of structured deformable objects. In *Proceedings of the 1999 conference on Graphics interface '99*, page 1–8, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1-55860-632-7. URL http://dl.acm.org/citation.cfm?id=351631.351638. 42

S. Dong, S. Kircher, and M. Garland. Harmonic functions for quadrilateral remeshing of arbitrary manifolds. *Computur Aided Geometric Design*, 22(5):392–423, 2005. 96

Albrecht Durer. *The painter's manual : a manual of measurement of lines, areas, and solids by means of compass and ruler assembled by Albrecht Du?rer for the use of all lovers of art with appropriate illustrations*. Abaris Books, New York, 1977. ISBN 9780913870525. 23

Jeffrey W. Eischen, Shigan Deng, and Timothy G. Clapp. Finite-Element modeling and control of flexible fabric parts. *IEEE Comput. Graph. Appl.*, 16(5): 71–80, September 1996. ISSN 0272-1716. doi: 10.1109/38.536277. URL http://dl.acm.org/citation.cfm?id=616042.618380. 38

Emmanuel Turquin, Marie-Paule Cani, and John Hughes. Sketching garments for virtual characters. In *Proc of Eurographics Workshop on Sketch-Based Interfaces and Modeling*, 2004. 80

Paul Erdos. Problem 3763. *American Mathematical Monthly*, 42:627, 1935. 26

Erik D. Demaine, Martin L. Demaine, Duks Koschitz, and Tomohiro Tachi. Curved crease folding: a review on art, design and mathematics. In *Proceedings of the IABSE-IASS Symposium: Taller, Longer, Lighter (IABSE-IASS 2011)*, London, England, September 2011. 32

Martin Fêdor. Application of inverse kinematics for skeleton manipulation in real-time. In *Proceedings of the 19th spring conference on Computer graphics*, pages 203–212, Budmerice, Slovakia, 2003. ACM. ISBN 1-58113-861-X. doi: 10.1145/984952.984986. URL http://portal.acm.org/citation.cfm?id=984952.984986&coll=GUIDE&dl=GUIDE&CFID=37953298&CFTOKEN=26917198. 10, 11

Wei-Wen Feng, Yizhou Yu, and Byung-Uck Kim. A deformation transformer for real-time cloth animation. *ACM Transactions on Graphics (TOG)*, page 108:1–108:9, 2010. doi: 10.1145/1833349.1778845. ACM ID: 1778845. 41

William Fong, Eric Darve, and Adrian Lew. Stability of asynchronous variational integrators. In *Proceedings of the 21st International Workshop on Principles of Advanced and Distributed Simulation*, PADS '07, page 38–44, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2898-8. doi: 10.1109/PADS.2007.29. URL http://dx.doi.org/10.1109/PADS.2007.29. 42

Thomas Geijtenbeek, Nicolas Pronost, Arjan Egges, and Mark H. Overmars. Interactive character animation using simulated physics. In *Eurographics - State of the Art Reports*, 2011. 14

Rony Goldenthal, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun. Efficient simulation of inextensible cloth. *ACM Transactions on Graphics*, 26(3):49, July 2007. ISSN 07300301. doi: 10.1145/1276377.1276438. URL http://portal.acm.org/citation.cfm?doid=1276377.1276438. 1, 40, 66

E. Goto, Y. Shi, and N. Yoshida. Extrapolated surface charge method for capacity calculation of polygons and polyhedra. *Journal of Computational Physics*, 100(1):105–115, 1992. ISSN 0021-9991. doi: 10.1016/0021-9991(92)90313-N. URL http://www.sciencedirect.com/science/article/pii/002199919290313N. 95, 97, 98

Kazushige Goto and Robert Van De Geijn. High-performance implementation of the level-3 BLAS. *ACM Transactions on Mathematical Software*, 35(1):1–14, 2008. ISSN 0098-3500. doi: http://doi.acm.org/10.1145/1377603.1377607. 90, 105

Sachin Goyal, N. C Perkins, and Christopher L Lee. Nonlinear dynamic intertwining of rods with self-contact. *arXiv:math-ph/0701017*, January 2007. URL http://arxiv.org/abs/math-ph/0701017. 38

Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popović. Style-based inverse kinematics. *ACM Trans. Graph.*, 23(3):522–531, August 2004. ISSN 0730-0301. doi: 10.1145/1015706.1015755. URL http://doi.acm.org/10.1145/1015706.1015755. 17

S Gupta. Automated process planning for sheet metal bending operations. *Journal of Manufacturing Systems*, 17(5):338–360, 1998. ISSN 02786125. doi: 10.1016/S0278-6125(98)80002-2. URL http://www.mendeley.com/research/automated-process-planning-for-sheet-metal-bending-operations/. 33

David Harmon, Etienne Vouga, Breannan Smith, Rasmus Tamstorf, and Eitan Grinspun. Asynchronous contact mechanics. *ACM Trans. Graph.*, 28:87:1–87:12, July 2009. ISSN 0730-0301. doi: http://doi.acm.org/10.1145/1531326.1531393. URL http://doi.acm.org/10.1145/1531326.1531393. 42, 70, 80

E. Hawkes, B. An, N. M. Benbernou, H. Tanaka, S. Kim, E. D. Demaine, D. Rus, and R. J. Wood. Programmable matter by folding. *Proceedings of the National Academy of Sciences*, June 2010. doi: 10.1073/pnas.0914069107. URL http://www.pnas.org/content/early/2010/06/24/0914069107.abstract. 32

D. Henrich, T. Ogasawara, and H. Worn. Manipulating deformable linear objects - contact states and point contacts. In *(ISATP '99) Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning, 1999*, pages 198–204. IEEE, 1999. ISBN 0-7803-5704-3. doi: 10.1109/ISATP.1999.782959. 6, 30

Michael H.Freedman, Zheng-Xu He, and Zhenghan Wang. Möbius energy of knots and unknots. *Annals of Mathematics, Second Series*, 139(1):1–50, 1994. 107

E. S.L Ho and T. Komura. Planning tangling motions for humanoids. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pages 507–512. IEEE, December 2007. ISBN 978-1-4244-1861-9. doi: 10.1109/ICHR.2007.4813918. 30

Edmond S. L. Ho and Taku Komura. Indexing and retrieving motions of characters in close contact. *IEEE Transactions on Visualization and Computer Graphics*, 15(3): 481–492, 2009a. URL http://portal.acm.org/citation.cfm?id=1515609.1515688&coll=GUIDE&dl=GUIDE&CFID=34637267&CFTOKEN=47210557. 21

Edmond S. L. Ho, Taku Komura, and Chiew-Lan Tai. Spatial relationship preserving character motion adaptation. *ACM Transactions on Graphics*, 29:1, July 2010. ISSN 07300301. doi: 10.1145/1778765.1778770. URL http://portal.acm.org/citation.cfm?doid=1778765.1778770. 22, 48, 81

Edmond S.L. Ho and Taku Komura. Character motion synthesis by topology coordinates. *Computer Graphics Forum*, 28(2):299–308, 2009b. doi: 10.1111/j.1467-8659.2009.01369.x. URL http://dx.doi.org/10.1111/j.1467-8659.2009.01369.x. viii, 21, 65, 76, 82

Jessica K. Hodgins, Wayne L. Wooten, David C. Brogan, and James F. O'Brien. Animating human athletics. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '95, page 71–78, New York, NY, USA, 1995. ACM. ISBN 0-89791-701-4. doi: 10.1145/218380.218414. URL http://doi.acm.org/10.1145/218380.218414. 12

Eugene Hsu, Kari Pulli, and Jovan Popovic. Style translation for human motion. *ACM Trans Graph*, 24(3):1082–1089, 2005. 15, 17

Peter Hunter. FEM/BEM NOTES, 2005. URL http://www.bioeng.auckland.ac.nz/cmiss/fembemnotes/fembemnotes.pdf. 37

Hayley N. Iben, James F. O'Brien, and Erik D. Demaine. Refolding planar polygons. *Discrete and Computational Geometry*, 41(3):444–460, April 2009. doi: 10.1007/s00454-009-9145-7. URL http://graphics.cs.berkeley.edu/papers/Iben-RPP-2009-04/. 25, 45, 46, 47, 50

Takeo Igarashi and John F. Hughes. Clothing manipulation. In *UIST '02: Proc of ACM Symp on User Interface Software and Technology*, page 91–100, 2002. 81

Takeo Igarashi and Jun Mitani. Apparent layer operations for the manipulation of deformable objects. *ACM Trans. Graph.*, 29(4):1–7, 2010. doi: 10.1145/1778765.1778847. URL http://portal.acm.org/citation.cfm?id=1778765.1778847. 107

Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. Interactive cover design considering physical constraints. *Computer Graphics Forum*, 28(7), 2009. 80, 106

Sumit Jain, Yuting Ye, and C. Karen Liu. Optimization-based interactive motion synthesis. *ACM Trans. Graph.*, 28(1):1–12, 2009. doi: 10.1145/1477926.1477936. URL http://portal.acm.org/citation.cfm?id=1477926.1477936. 14

T. Jakobsen. Advanced character physics. *In Game Developers Conference Proc*, page 383–401, 2001. 66

Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic coordinates for character articulation. *ACM Trans. Graph.*, 26(3), 2007. 96

Michael Kass. An introduction to physically based modeling. In *SIGGRAPH 95 Course Notes*, 1995. 40, 42

D. Katic and M. Vukobratovic. A neural network-based classification of environment dynamics models for compliant control of manipulation robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 28(1):58–69, February 1998. ISSN 1083-4419. doi: 10.1109/3477.658578. 28

L. E Kavraki, P. Svestka, J. -C Latombe, and M. H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, August 1996. ISSN 1042-296X. doi: 10.1109/70.508439. 2

I. Klapper. Biological applications of the dynamics of twisted elastic rods. *Journal of Computational Physics*, 125(2):325–337, May 1996. ISSN 00219991. doi: 10.1006/jcph.1996.0097. URL http://adsabs.harvard.edu/abs/1996JCoPh.125..325K. 38

Yoshihito Koga, Koichi Kondo, James Kuffner, and Jean-Claude Latombe. Planning motions with intentions. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 395–408. ACM, 1994. ISBN 0-89791-667-0. doi: 10.1145/192161.192266. URL http://portal.acm.org/citation.cfm?id=192161.192266. 16

Ryo Kondo, Takashi Kanai, and Ken-ichi Anjyo. Directable animation of elastic objects. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '05, page 127–134, New York, NY, USA,

2005. ACM. ISBN 1-59593-198-8. doi: 10.1145/1073368.1073385. URL http://doi.acm.org/10.1145/1073368.1073385. 81

Lucas Kovar and Michael Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Trans. Graph.*, 23(3):559–568, August 2004. ISSN 0730-0301. doi: 10.1145/1015706.1015760. URL http://doi.acm.org/10.1145/1015706.1015760. 17

Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. *ACM Trans. Graph.*, 21(3):473–482, 2002. doi: 10.1145/566654.566605. URL http://portal.acm.org/citation.cfm?id=566605. 18

Steven M LaValle and James J Kuffner. Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20(5):378–400, May 2001. ISSN 0278-3649, 1741-3176. doi: 10.1177/02783640122067453. URL http://ijr.sagepub.com/content/20/5/378. 29

Steven M Lavalle, James J Kuffner, and Jr. Rapidly-Exploring random trees: Progress and prospects. *ALGORITHMIC AND COMPUTATIONAL ROBOTICS: NEW DIRECTIONS*, pages 293—308, 2000. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.36.7457. 2, 15

Neil D Lawrence. The gaussian process latent variable model. Technical Report CS-06-03, Department of Computer Science, The University of Sheffield, 2006a. 19

Neil D Lawrence. Local distance preservation in the gp-lvm through back constraints. *IN ICML*, 148:513—520, 2006b. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.124.2767. 20

Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Trans. Graph.*, 21(3):491–500, July 2002. ISSN 0730-0301. doi: 10.1145/566654.566607. URL http://doi.acm.org/10.1145/566654.566607. 17

Sung-Hee Lee and Demetri Terzopoulos. Heads up!: biomechanical modeling and neuromuscular control of the neck. *ACM Trans. Graph.*, 25(3):1188–1198, 2006. doi: 10.1145/1141911.1142013. URL http://portal.acm.org/citation.cfm?id=1141911.1142013. 14

C. Karen Liu. Synthesis of interactive hand manipulation. In *2008 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 163—172, 2008. 14

C. Karen Liu and Zoran Popović. Synthesis of complex dynamic character motion from simple animations. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 408–416, San Antonio, Texas, 2002. ACM. ISBN 1-58113-521-1. doi: 10.1145/566570.566596. URL http://portal.acm.org/citation.cfm?id=566596. 14

C. Karen Liu, Aaron Hertzmann, and Zoran Popović. Learning physics-based motion style with nonlinear inverse optimization. In *ACM SIGGRAPH 2005 Papers*, pages 1071–1081, Los Angeles, California, 2005. ACM. doi: 10.1145/1186822.1073314. URL http://portal.acm.org/citation.cfm?id=1186822.1073314&coll=GUIDE&dl=GUIDE&CFID=34637267&CFTOKEN=47210557. 14

Wan-Yen Lo and Matthias Zwicker. Bidirectional search for interactive motion synthesis. *Computer Graphics Forum*, 29(2):563–573, May 2010. ISSN 1467-8659. doi: 10.1111/j.1467-8659.2009.01626.x. URL http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2009.01626.x/abstract. 18, 19

Liang Lu and S. Akella. Folding cartons with fixtures: a motion planning approach. *IEEE Transactions on Robotics and Automation*, 16(4):346–356, August 2000. ISSN 1042-296X. doi: 10.1109/70.864227. 33

Jeremy Maitin-Shepard, Marco Cusumano-Towner, Jinna Lei, and Pieter Abbeel. Cloth grasp point detection based on Multiple-View geometric cues with application to robotic towel folding. *Proc of IEEE Int Conf on Robotics and Automation*, 2010. 33

N. H Malik. A review of the charge simulation method and its applications. *IEEE Transactions on Electrical Insulation*, 24(1):3–20, February 1989. ISSN 0018-9367. doi: 10.1109/14.19861. 96

T. Matsuno, T. Fukuda, and F. Arai. Flexible rope manipulation by dual manipulator system using vision sensor. In *2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2001. Proceedings*, volume 2, pages 677–682 vol.2. IEEE, 2001. ISBN 0-7803-6736-7. doi: 10.1109/AIM.2001.936748. 30

James McCann and Nancy Pollard. Local layering. *ACM Trans. Graph.*, 28(3): 1–7, 2009. doi: 10.1145/1531326.1531390. URL http://portal.acm.org/citation.cfm?id=1531326.1531390. 107

S. Ménardais, R. Kulpa, F. Multon, and B. Arnaldi. Synchronization for dynamic blending of motions. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, page 325–335, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association. ISBN 3-905673-14-2. doi: 10.1145/1028523.1028567. URL http://dx.doi.org/10.1145/1028523.1028567. 17

Michael Meredith and Steve Maddock. Adapting motion capture data using weighted real-time inverse kinematics. *Comput. Entertain.*, 3(1):5–5, 2005. doi: 10.1145/1057270.1057281. URL http://portal.acm.org/citation.cfm?id=1057270.1057281&coll=GUIDE&dl=GUIDE&CFID=37953298&CFTOKEN=26917198. 11

Jianyuan Min, Yen-Lin Chen, and Jinxiang Chai. Interactive generation of human animation with deformable motion models. *ACM Trans. Graph.*, 29(1):1–12, 2009. doi: 10.1145/1640443.1640452. 20

Luis Molina-Tanco, Adrian Hilton, Luis Molina, and Tanco Adrian Hilton. Realistic synthesis of novel human movements from a database of motion capture examples. *IN WORKSHOP ON HUMAN MOTION (HUMO'00*, pages 137—142, 2000. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.3357. 17

M. Moll and L. E Kavraki. Path planning for deformable linear objects. *IEEE Transactions on Robotics*, 22(4):625–636, August 2006. ISSN 1552-3098. doi: 10.1109/TRO.2006.878933. 31

Yuki Mori and Takeo Igarashi. Plushie: an interactive design system for plush toys. In *ACM SIGGRAPH 2007 papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM. doi: 10.1145/1275808.1276433. URL http://doi.acm.org/10.1145/1275808.1276433. 80

Tomohiko Mukai and Shigeru Kuriyama. Geostatistical motion interpolation. *ACM Transactions on Graphics*, 24:1062, July 2005. ISSN 07300301. doi: 10.1145/1073204.1073313. URL http://portal.acm.org/citation.cfm?doid=1073204.1073313. 15, 16, 17

R. Mukundan. A robust inverse kinematics algorithm for animating a joint chain. *Int. J. Comput. Appl. Technol.*, 34(4):303–308, 2009. URL http://portal.acm.org/citation.cfm?id=1521307.1521319&coll=GUIDE&dl=GUIDE&CFID=37953298&CFTOKEN=26917198. 10, 11

Matthias Müller, Leonard McMillan, Julie Dorsey, and Robert Jagnow. Real-time simulation of deformation and fracture of stiff materials. In *Proceedings of the Eurographic workshop on Computer animation and simulation*, page 113–124, New York, NY, USA, 2001. Springer-Verlag New York, Inc. ISBN 3-211-83711-6. URL http://dl.acm.org/citation.cfm?id=776350.776361. 37

Matthias Müller, Julie Dorsey, Leonard McMillan, Robert Jagnow, and Barbara Cutler. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '02, page 49–54, New York, NY, USA, 2002. ACM. ISBN 1-58113-573-4. doi: 10.1145/545261.545269. URL http://doi.acm.org/10.1145/545261.545269. 37

Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. Meshless deformations based on shape matching. *ACM Trans. Graph.*, 24(3):471–478, July 2005. ISSN 0730-0301. doi: 10.1145/1073204.1073216. URL http://doi.acm.org/10.1145/1073204.1073216. 41

Bela de Sz. Nagy. Solution to problem 3763. *American Mathematical Monthly*, 46: 176–177, March 1939. ix, 26, 27

Andrew Nealen, Matthias Mueller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically based deformable models in computer graphics. *Computer Graphics Forum*, 25(4):809–836, December 2006. ISSN 0167-7055. URL http://dx.doi.org/10.1111/j.1467-8659.2006.01000.x. 36, 39, 40

Michael Neff and Eugene Fiume. Modeling tension and relaxation for computer animation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '02, page 81–88, New York, NY, USA, 2002. ACM. ISBN 1-58113-573-4. doi: 10.1145/545261.545275. URL http://doi.acm.org/10.1145/545261.545275. 11, 12

NVidia. PhysX, 2008. 69, 86, 106

James F. O'Brien, Adam W. Bargteil, and Jessica K. Hodgins. Graphical modeling and animation of ductile fracture. *ACM Trans. Graph.*, 21(3):291–294, July 2002. ISSN 0730-0301. doi: 10.1145/566654.566579. URL http://doi.acm.org/10.1145/566654.566579. 37

Rick Parent. *Computer Animation: Algorithms and Techniques*. Morgan Kaufmann, 2002. ISBN 9781558605794. 5

Sang Il Park, Tae-hoon Kim, and Hyun Joon Shin. Online motion blending for real-time locomotion generation. computer animation and virtual worlds 15. *COMPUT. ANIMAT. VIRTUAL WORLDS*, 15:3—4, 2004. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.102.373. 17

Jeff Phillips, Andrew Ladd, and Lydia E Kavraki. Simulated knot tying. *IN PROCEEDINGS OF THE IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION*, pages 841—846, 2002. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.9803. 6, 30

G. Picinbono, H. Delingette, and N. Ayache. Real-Time large displacement elasticity for surgery simulation: Non-Linear Tensor-Mass model. *IN PROCEEDINGS OF THE THIRD INTERNATIONAL CONFERENCE ON MEDICAL ROBOTICS, IMAGING AND COMPUTER ASSISTED SURGERY: MICCAI 2000*, 2000:643—652, 2000. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.14.6382. 38

Guillaume Picinbono, Hervé Delingette, and Nicholas Ayache. Non-Linear and anisotropic elastic soft tissue models for medical simulation. 2001. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.14.5712. 38

Alla Safonova, Jessica K. Hodgins, and Nancy S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Trans. Graph.*, 23(3):514–521, August 2004. ISSN 0730-0301. doi: 10.1145/1015706.1015754. URL http://doi.acm.org/10.1145/1015706.1015754. 14, 17

M. Saha and P. Isto. Manipulation planning for deformable linear objects. *IEEE Transactions on Robotics*, 23(6):1141–1150, December 2007. ISSN 1552-3098. doi: 10.1109/TRO.2007.907486. 31

Johannes Schmid, Martin Sebastian Senn, Markus Gross, and Robert W Sumner. OverCoat: an implicit canvas for 3D painting. *ACM Trans. Graph.*, 30(4):28:1–28:10, August 2011. ISSN 0730-0301. doi: 10.1145/2010324.1964923. ACM ID: 1964923. 106

S. K. Semwal, Bill Watson, and Debra L. McCullough. Human muscle modeling using generalized cylinders for volume consideration. In *Proceedings of the 24th IASTED international conference on Biomedical engineering*, pages 118–123, Innsbruck, Austria, 2006. ACTA Press. ISBN 0-88986-578-7. URL http://portal.acm.org/citation.cfm?id=1166506.1166527&coll=GUIDE&dl=GUIDE&CFID=37962972&CFTOKEN=32089792. 14

Ari Shapiro, Marcelo Kallmann, and Petros Faloutsos. Interactive motion correction and object manipulation. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, I3D '07, page 137–144, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-628-8. doi: 10.1145/1230100.1230124. URL http://doi.acm.org/10.1145/1230100.1230124. 15, 16

Hubert P. H. Shum. *Simulating Interactions Among Multiple Characters*. PhD thesis, University of Edinburgh, UK, 2010. 17

Hubert P. H. Shum, Taku Komura, Masashi Shiraishi, and Shuntaro Yamazaki. Interaction patches for multi-character animation. *ACM Trans. Graph.*, 27(5):1–8, 2008a. doi: 10.1145/1409060.1409067. URL http://portal.acm.org/citation.cfm?id=1409060.1409067. 19

Hubert P. H. Shum, Taku Komura, and Shuntaro Yamazaki. Simulating interactions of avatars in high dimensional state space. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pages 131–138, Redwood City, California, 2008b. ACM. ISBN 978-1-59593-983-8. doi: 10.1145/1342250.1342271. URL http://portal.acm.org/citation.cfm?id=1342271. 18

Hubert P. H. Shum, Taku Komura, and Shuntaro Yamazaki. Simulating multiple character interactions with collaborative and adversarial goals. *IEEE Transactions on Visualization and Computer Graphics*, 99(PrePrints), 2010. ISSN 1077-2626. doi: http://doi.ieeecomputersociety.org/10.1109/TVCG.2010.257. 18, 19

Marco da Silva, Yeuhi Abe, and Jovan Popović. Interactive simulation of stylized human locomotion. In *ACM SIGGRAPH 2008 papers*, pages 1–10, Los Angeles,

California, 2008. ACM. doi: 10.1145/1399504.1360681. URL http://portal.acm.org/citation.cfm?id=1360681. 14

Guang Song and N. M Amato. A motion-planning approach to folding: from paper craft to protein folding. *IEEE Transactions on Robotics and Automation*, 20(1):60–71, February 2004. ISSN 1042-296X. doi: 10.1109/TRA.2003.820926. 33

Jones Spillmann and Matthias Teschner. An adaptive contact model for the robust simulation of knots. *Computer Graphics Forum (Proc of Eurographics 2008)*, 27 (2):497–506, 2008. 65

C. D Takahashi, R. A Scheidt, and D. J Reinkensmeyer. Impedance control and internal model formation when reaching in a randomly varying dynamical environment. *J. NEUROPHYSIOL*, 86:1047—1051, 2001. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.24.4015. 12

T. Tamei, T. Matsubara, A. Rai, and T. Shibata. Reinforcement learning of clothing assistance with a dual-arm robot. In *2011 11th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 733–738. IEEE, October 2011. ISBN 978-1-61284-866-2. doi: 10.1109/Humanoids.2011.6100915. 82

Jie Tan, K. Liu, and G. Turk. Stable Proportional-Derivative controllers. *IEEE Computer Graphics and Applications*, 31(4):34–44, August 2011. ISSN 0272-1716. doi: 10.1109/MCG.2011.30. 12

M. Tarokh and S. Bailey. Force tracking with unknown environment parameters using adaptive fuzzy controllers. In *, 1996 IEEE International Conference on Robotics and Automation, 1996. Proceedings*, volume 1, pages 270–275 vol.1. IEEE, April 1996. ISBN 0-7803-2988-0. doi: 10.1109/ROBOT.1996.503789. 29

Demetri Terzopoulos and Andrew Witkin. Physically based models with rigid and deformable components. *IEEE Comput. Graph. Appl.*, 8(6):41–51, November 1988. ISSN 0272-1716. doi: 10.1109/38.20317. URL http://dx.doi.org/10.1109/38.20317. 38

Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. *SIGGRAPH Comput. Graph.*, 21(4):205–214, August 1987. ISSN 0097-8930. doi: 10.1145/37402.37427. URL http://doi.acm.org/10.1145/37402.37427. 38

M Teschner, B Heidelberger, M Mueller, D Pomeranets, and M Gross. Optimized spatial hashing for collision detection of deformable objects. *Proceedings of Vision, Modeling, Visualization (VMV 2003)*, pages 47–54, 2003. 80

Matthias Teschner, Bruno Heidelberger, Matthias Muller, and Markus Gross. A versatile and robust model for geometrically complex deformable solids. In *Proceedings of the Computer Graphics International*, page 312–319, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2171-1. doi: 10.1109/CGI.2004.6. URL http://dl.acm.org/citation.cfm?id=1009379.1009610. 40

Adrien Treuille, Yongjoon Lee, and Zoran Popović. Near-optimal character animation with continuous control. *ACM Trans. Graph.*, 26(3):7, 2007. doi: 10.1145/1276377.1276386. URL http://portal.acm.org/citation.cfm?id=1276386. 18

Emmanuel Turquin, Marie-Paule Cani, and John F. Hughes. Sketching garments for virtual characters. In *ACM SIGGRAPH 2007 courses*, SIGGRAPH '07, New York, NY, USA, 2007. ACM. doi: 10.1145/1281500.1281539. URL http://doi.acm.org/10.1145/1281500.1281539. 80

Luis Unzueta, Manuel Peinado, Ronan Boulic, and Ángel Suescun. Full-body performance animation with sequential inverse kinematics. *Graph. Models*, 70 (5):87–104, 2008. URL http://portal.acm.org/citation.cfm?id=1450343.1450488&coll=GUIDE&dl=GUIDE&CFID=37953298&CFTOKEN=26917198. 11

A. Van Oosterom and J. Strackee. The solid angle of a plane triangle. *IEEE Transactions on Biomedical Engineering*, BME-30(2):125–126, February 1983. ISSN 0018-9294. doi: 10.1109/TBME.1983.325207. 95

S. T Venkataraman, S. Gulati, J. Barhen, and N. Toomarian. A neural network based identification of environments models for compliant control of space robots. *IEEE Transactions on Robotics and Automation*, 9(5):685–697, October 1993. ISSN 1042-296X. doi: 10.1109/70.258059. 29

Pascal Volino and Nadia Magnenat Thalmann. Implementing fast cloth simulation with collision response. In *Proceedings of the International Conference on Computer Graphics*, CGI '00, page 257–266, Washington, DC, USA, 2000. IEEE Computer Society. ISBN 0-7695-0643-7. URL http://dl.acm.org/citation.cfm?id=792759.793086. 42

H. Wakamatsu, A. Tsumaya, E. Arai, and S. Hirai. Manipulation planning for Knotting/Unknotting and tightly tying of deformable linear objects. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005. ICRA 2005*, pages 2505– 2510. IEEE, April 2005. ISBN 0-7803-8914-X. doi: 10.1109/ROBOT.2005.1570489. 30

Hidefumi Wakamatsu, Eiji Arai, and Shinichi Hirai. Knotting/Unknotting manipulation of deformable linear objects. *The International Journal of Robotics Research*, 25(4):371 –395, April 2006. doi: 10.1177/0278364906064819. URL http://ijr.sagepub.com/content/25/4/371.abstract. 30

He Wang and Taku Komura. Energy-Based pose unfolding and interpolation for 3D articulated characters. In *Motion in Games*, volume 7060, pages 110–119. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-25089-7, 978-3-642-25090-3. URL http://www.springerlink.com/content/41613nh014697048/. 2, 44

He Wang and Taku Komura. Manipulation of flexible objects by geodesic control. *Computer Graphics Forum*, 31(2), 2012. 58, 81

Huamin Wang, Florian Hecht, Ravi Ramamoorthi, and James O'Brien. Example-based wrinkle synthesis for clothing animation. *ACM Trans. Graph.*, 29(4):1–8, 2010a. doi: 10.1145/1778765.1778844. URL http://portal.acm.org/citation.cfm?id=1778765.1778844. 1, 40, 80

Huamin Wang, James F. O'Brien, and Ravi Ramamoorthi. Data-driven elastic models for cloth: modeling and measurement. *ACM Trans. Graph.*, 30(4):71:1–71:12, August 2011. ISSN 0730-0301. doi: 10.1145/2010324.1964966. URL http://doi.acm.org/10.1145/2010324.1964966. 41

Jack M Wang, David J Fleet, and Aaron Hertzmann. Gaussian process dynamical models. *IN NIPS*, 18:1441—1448, 2006. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.72.4340. 19

Jack M Wang, David J Fleet, Senior Member, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE TRANS. PATTERN ANAL. MACHINE INTELL*, 2007. URL http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.132.5571. 19

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Optimizing walking controllers. *ACM Transactions on Graphics*, 28:1, December 2009. ISSN 07300301. doi: 10.1145/1618452.1618514. URL http://portal.acm.org/citation.cfm?doid=1618452.1618514. 14

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Optimizing walking controllers for uncertain inputs and environments. *ACM Transactions on Graphics*, 29:1, July 2010b. ISSN 07300301. doi: 10.1145/1778765.1778810. URL http://portal.acm.org/citation.cfm?doid=1778765.1778810. 14

Xiaolin Wei, Jianyuan Min, and Jinxiang Chai. Physically valid statistical models for human motion generation. *ACM Transactions on Graphics*, 30:1–10, May 2011. ISSN 07300301. doi: 10.1145/1966394.1966398. URL http://dl.acm.org/citation.cfm?id=1966398. 13, 14, 19, 20

Wikipedia contributors. Winding number, May 2012. URL http://en.wikipedia.org/w/index.php?title=Winding_number&oldid=494139574. Page Version ID: 494139574. 83

Andrew Witkin and Michael Kass. Spacetime constraints. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 159–168. ACM, 1988. ISBN 0-89791-275-6. doi: 10.1145/54852.378507. URL http://portal.acm.org/citation.cfm?id=54852.378507&coll=GUIDE&dl=GUIDE&CFID=34639582&CFTOKEN=49101429. 14, 15

Chris Wojtan, Peter J. Mucha, and Greg Turk. Keyframe control of complex particle systems using the adjoint method. In *Proc of ACM SIGGRAPH/Eurographics Symp on Computer Animation*, page 15–23, 2006. 81

Y. Yamakawa, A. Namiki, M. Ishikawa, and M. Shimojo. Knotting manipulation of a flexible rope by a multifingered hand system based on skill synthesis. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008*, pages 2691–2696. IEEE, September 2008. ISBN 978-1-4244-2057-5. doi: 10.1109/IROS.2008.4650802. 30

Katsu Yamane, James J. Kuffner, and Jessica K. Hodgins. Synthesizing animations of human manipulation tasks. *ACM Trans. Graph.*, 23(3):532–539, 2004. doi: 10.1145/1015706.1015756. URL http://portal.acm.org/citation.cfm?id=1015756. 15, 16

Yang Yang, Irwin Tobias, and Wilma K. Olson. Finite element analysis of DNA supercoiling. *The Journal of Chemical Physics*, 98(2):1673, 1993. ISSN 00219606. doi: 10.1063/1.464283. URL http://adsabs.harvard.edu/abs/1993JChPh..98.1673Y. 38

Yuting Ye and C. Karen Liu. Animating responsive characters with dynamic constraints in near-unactuated coordinates. In *ACM SIGGRAPH Asia 2008 papers*, pages 1–5, Singapore, 2008. ACM. doi: 10.1145/1457515.1409065. URL http://portal.acm.org/citation.cfm?id=1457515.1409065&coll=GUIDE&dl=GUIDE&CFID=34636021&CFTOKEN=22217172. 14

Yuting Ye and C. Karen Liu. Synthesis of responsive motion using a dynamic model. *Computer Graphics Forum*, 29:555–562, June 2010. ISSN 01677055, 14678659. doi: 10.1111/j.1467-8659.2009.01625.x. URL http://www.mendeley.com/research/synthesis-responsive-motion-using-dynamic-model/. 19

KangKang Yin, Kevin Loken, and Michiel van de Panne. SIMBICON: simple biped locomotion control. In *ACM SIGGRAPH 2007 papers*, page 105, San Diego, California, 2007. ACM. doi: 10.1145/1275808.1276509. URL http://portal.acm.org/citation.cfm?id=1276509. 10, 12, 14

KangKang Yin, Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. Continuation methods for adapting simulated skills. In *ACM SIGGRAPH 2008 papers*, pages 1–7, Los Angeles, California, 2008. ACM. doi: 10.1145/1399504.1360680. URL http://portal.acm.org/citation.cfm?id=1399504.1360680&coll=GUIDE&dl=GUIDE&CFID=35756497&CFTOKEN=11094901. 12

Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. Interactive cover design considering physical constraints. *Computer Graphics Forum*, 28(7), 2009. 82

Kun Zhou, Weiwei Xu, Yiying Tong, and Mathieu Desbrun. Deformation transfer to Multi-Component objects. *Computer Graphics Forum*, 29(2):319–325, May 2010. ISSN 1467-8659. doi: 10.1111/j.1467-8659.2009.01601.x. URL http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2009.01601.x/abstract. 81

Victor B Zordan and Jessica K Hodgins. Tracking and modifying upper-body human motion data with dynamic simulation. *IN COMPUTER ANIMATION AND*

SIMULATION '99, pages 13—22, 1999. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.6909. 11, 12

Victor Brian Zordan and Jessica K. Hodgins. Motion capture-driven simulations that hit and react. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '02, page 89–96, New York, NY, USA, 2002. ACM. ISBN 1-58113-573-4. doi: 10.1145/545261.545276. URL http://doi.acm.org/10.1145/545261.545276. 11, 12