

**Automated design, accessible
fabrication, and learning-based control
on cable-driven soft robots with complex
shapes**

Kai-Hung Chang
CMU-RI-TR-18-35
May 29, 2018



The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:

Nancy S. Pollard, *chair*
Stelian Coros, *ETH Zurich*
Oliver Kroemer
Wen Sun

*Submitted in partial fulfillment of the requirements
for the degree of Master in Robotics.*

Copyright © 2018 Kai-Hung Chang. All rights reserved.

To everyone that makes me who I am today.

Abstract

The emerging field of soft robots has shown great potential to outperform their rigid counterparts due to the soft and safe nature and the capability of performing complex and compliant motions. Many are built, but the designs are conservative and limited to regular shapes. The widely-used fabrication method contains bulky pumps, tethered tubings, and silicone elastomer that takes hours to fully cure. Control methods for soft robots, both model-based and learning-based, are investigated, but none of them is applied to soft robots with complex shapes. In this thesis, a set of tools and methodology is developed for cables-driven soft robots, including 1) an automated design system that generates optimal cable placement designs given desired configuration, 2) a forward simulator that predicts the soft body motion under cable contractions, 3) a fabrication pipeline that is fast, low-cost, and accessible to non-experts, and 4) various algorithms to solve inverse kinematics using finite-element simulation, supervised learning, and reinforcement learning.

Acknowledgments

First of all, many thanks to my advisors, Nancy Pollard and Stelian Coros for their academic guidance and omnipresent mentorship throughout the past two years of my master study. I am also grateful to both my thesis committee members, Oliver Kroemer and Wen Sun, and other faculties at CMU including David Held, Chris Atkeson, and Jim McCann for their insightful questions and valuable advice.

I would like to thank all my friends and labmates at CMU and Computer Graphics Lab for supporting my work on soft robots. Special thanks to James Bern, a brilliant PhD student, whom I had a wonderful time working with and learning from. I would also like to thank Ruta Desai for the support provided during my hardest time during research. Thank you to my collaborators: Dominik Bauer, Cornelia Schlangenhaus, Jonathan King, and Daniele Moro, for your effort in bringing foam robots to reality.

Last but not least, I'm grateful to my parents for always supporting me and encouraging me to pursue what I love. Thank you for everything.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Challenges	2
1.3	Contribution	3
1.4	Thesis Outline	3
2	Background	5
2.0.1	Design	5
2.0.2	Fabrication	6
2.0.3	Control	6
3	Cable-Driven Plush Toy Robots	9
3.1	Introduction	9
3.2	Fabrication	9
3.2.1	Body	10
3.2.2	Cables	11
3.2.3	Internal Winch	11
3.2.4	Stoppers	12
3.2.5	Cable Sheath	12
3.3	Modeling and Simulation	13
3.3.1	Deformation Energy	14
3.3.2	Cable Energy and Winch	15
3.3.3	Pin Energy	17
3.4	Inverse Kinematics for Soft Bodies	17
3.5	Winch-Cable Design Automation	19
3.6	Results	21
3.7	Discussion	22
3.7.1	Comparison between Fabricated Robots against Simulation	22
3.7.2	Future Work	24
3.8	Summary	25
4	Cable-Driven Foam Casting Robots	27
4.1	Introduction	27
4.2	Fabrication	28

4.3	Compatibility to Design Tools	30
4.4	Fabricated Results	31
4.5	Capability of Foam Manipulator	32
4.5.1	Grasping	33
4.5.2	Strength of Grasps	33
4.5.3	Manipulation Task	34
4.6	Comparison with other types of Soft Robots	36
4.6.1	Actuation	36
4.6.2	Material	38
4.6.3	Fabrication	40
4.7	Summary	41
5	Learning-based Control on Cable-Driven Foam Manipulator	43
5.1	Introduction	43
5.2	General Formulation: State, Action	44
5.3	Deep Inverse Kinematics Model	44
5.3.1	Introduction	44
5.3.2	Learning in Simulation	45
5.3.3	Learning on Real Robots	50
5.3.4	Application: Tele-Operation	51
5.3.5	Conclusion	51
5.4	Deep Reinforcement Learning	53
5.4.1	Introduction	53
5.4.2	Experiments on Reinforcement Learning	54
5.4.3	Final Results	57
5.4.4	Comparison with other Learning-based Methods	60
5.4.5	Conclusion	61
6	Conclusions	63
	Bibliography	65

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

Chapter 1

Introduction

1.1 Motivation

Robots bring convenience to human lives in automation, manipulation, navigation, etc. and thus have been a popular research topic in 21st century. Most robots are restricted to manufacture factories or research labs, but few go to human daily lives. These robots are composed of rigid components and can be dangerous when interacting with people. Therefore, for safety reasons, rigid robots are usually restricted to strictly-defined open area. In contrast to rigid robots, soft robots possess great advantages. Here we strictly refer soft robots to those that are made up of soft materials. Due to its soft nature, soft robots can be safer and thus more interactive with human. Another important advantage of soft robots is that they can perform more complex motions than rigid robots. There are a lot of biological examples, such as starfish, human tongue, elephant trunk, etc, that show this capability with soft tissues. Therefore, soft robots have the potential of performing more dexterous manipulation tasks and moving across unknown irregular terrains. Though plenty of researches have been conducted on different types of soft robot, there is not yet a single class of soft robot with well-developed design tools, fabrication method, and control theory.

To build robots that can genuinely be immersed within human daily lives, robots also need to be accessible to the public. In other words, they have to be low-cost and easy to design, fabricate, and control. The cost of electronics has vastly decreased

nowadays and cheap motors and sensor modules are affordable to non-expert people. Fabrication technologies have also evolved to democratize the fabrication process. People can design rigid robots with free 3D computer-aided design (CAD) softwares like SketchUp or Blender, fabricate with easy-to-use machine tools like 3D printers or laser cutters, and control with cheap micro-controllers like Arduino or Raspberry Pi. Soft robots, on the contrary, do not have these related tools that allow the public to easily design and build a functional soft robot.

1.2 Challenges

Soft robots, though have shown great potential in application, remain challenging in aspects including design, fabrication, and control.

First of all, soft robots' capability of performing complex motions not only brings a more complicated control problem, but also makes the design process even more non-intuitive. For instance, a rotational motor joint of a rigid robot indicates its motion space straightforwardly whereas the placement of actuators on soft robots can be difficult to decide during the design process. Most soft robots developed in research labs have regular non-interesting shapes, mostly geometry primitives[49][57]. Other soft robots that have more appealing shapes or motions require expert knowledge of material dynamics or huge amount of time for iterative design.

Secondly, different materials and fabrication methods for soft robots have been studied, but they have different limitations. Silicone elastomers[49][9][57][32][27] is a widely used material, but the long curing time and the need of oven can be undesirable for fabrication. The fluidic actuation systems that often come along with silicone-based soft robots can be impractical to build untethered soft robots, not to mention the bulky compressor or dangerous tanks attached on the other end of tether cord.

Last but not least, soft robots, unlike rigid robots, do not have a set of well-established theories for kinematics, dynamics, and control. When dealing with rigid robots, we can build kinematics model using linkages and joints, perform inverse kinematics, and compute force/torque propagation. But since soft robots do not have discrete components and should be modeled using continuum mechanics, rigid robot theories and strategies can not be transferred. More complicated control problem such as manipulation is still an unsolved problem.

1.3 Contribution

In conclusion, motivated by the great potential of soft robots and considering the accessibility of the developed soft robots, we explore a novel class of soft robot-cable-driven soft robots, and develop a set of related tools and methodologies, which includes:

- An interactive design tool that generates optimal cable placement design based on the desired motion given by the user
- A finite-element(FEM) based simulator that is physically realistic and real-time
- Fabrication method that is fast and easy to non-expert public
- Control method to perform inverse kinematics with different approaches: FEM-based analytic solution, behavior cloning, and reinforcement learning.

With these tools, any non-expert public should be able to design, build, and control a cable-driven soft robot of their own.

The entire thesis is a collaboration work with James Bern on plush toy robot projects and Dominik Bauer, Cornelia Schlagenhauf, Jonathan King, and Daniele Moro on foam robot projects. Both projects are advised by Nancy Pollard and Stelian Coros.

For the plush toy robot project, my contribution include developing simulation models for cable-driven plush toy robots, investigating the fabrication methods, and building the automated design system. As for the foam robot projects, I started the idea of foam-casting fabrication technique, coded software for all learning-based control algorithms as well as motor communication, and helped engineer the tele-operation hardware.

The plushy toy robot project is submitted to SIGGRAPH 2017[5].

1.4 Thesis Outline

Chapter 2 summaries other people’s work related to soft robots in detail. Chapter 3 introduces the interactive design system, fabrication, and inverse kinematics using FEM simulators for, but not limited to cable-driven plush toy robots. Chapter 4 improves the fabrication method to foam casting and compares other fabrication

1. Introduction

techniques used in other papers. Chapter 5 focuses on learning-based control of cable-driven foam manipulators to solve inverse kinematics and manipulation tasks.

Chapter 2

Background

2.0.1 Design

Our design system is inspired by the considerable attention the research community has given to the development of computational tools for designing physical surfaces. Recent investigations, for example, represent surfaces using paper-craft models [24] [37] and paper pop-ups [30][29], inflatable structures [50][52], wire-mesh sculptures [18], flexible rod and curve networks [41] [62], procedurally-generated filigrees [12], tensegrities [19], inextensible sheets with cut patterns that lead to auxetic behavior [26], modular interlocking [53] or self-supporting primitives[54][59], and, closest to our work, plush toys [23][39]. While these efforts are aimed at creating static depictions of shape, the goal of our work is to create physical soft robots that are capable of producing purposeful motions.

The challenge of creating physical robots that are specifically designed to produce interesting motions is also fueling an active area of research. Design methods for articulated characters [2][8], mechanical automata [10][15][55][63], and even walking [36] and flying [16] robots have been recently proposed. While these works study mechanical structures composed primarily of rigid body parts, the method we propose considers soft physical systems that undergo large deformations. Consequently, our work is most closely related to the methods introduced by [6] and [51]. The former employs shape optimization techniques to control the deformations of a silicone skin driven by an underlying animatronic system. The latter optimizes a distribution of

2. Background

soft and rigid materials, as well as external actuation forces, so that the deformations of elastic characters match a set of input poses. In contrast to these methods, our design system lets users author the motions of soft robots through an intuitive posing interface that automatically computes contractions for embedded muscle-like elements. While the motion is being authored, our design system also creates an internal actuation system that will drive the motions of soft robots.

2.0.2 Fabrication

Please refer to Chapter 4 Section 4.6.

2.0.3 Control

Controlling soft robots can be a hard problem due to the complexity of the continuum motion. Previous works have studied particularly the inverse kinematics problem for soft robots that solves for the optimal actuation that drives the robot to a desired configuration. There are two main categories: model-based approaches and data-driven approaches.

In model-based approaches, the dynamics of the soft robot motion is formulated using a physics model. [45] models caterpillar-like soft robots as a series of extensible linkages. Another model that is often used for tentacle-shaped soft robots is piecewise-constant-curvature model[34][33][11]. For soft robots with arbitrary shapes, [17] presents a real-time solution using finite element method(FEM). In this thesis, an FEM-based solution to inverse kinematics is presented in Section 3.4, but we propose a very different treatment of cable-driven control of soft robots which explicitly computes the relationship between changes in cable contraction and the resulting changes in deformed shape of the overall robot.

Data-driven approach is accomplished by learning a universal function approximator to represent the mapping between the deformation and the actuation. A neural network is built to learn inverse kinematics on a cable-driven soft tentacle manipulator with 2 degrees of freedom[21]. In Section 5.3, similar supervised-learning approach is utilized on various 3D soft manipulators, including anthropomorphic and non-anthropomorphic ones, with 10 degrees of freedom. Other than supervised-learning,

we also include experiments and results using reinforcement learning algorithms to compare the pros and cons between the two learning approaches.

2. Background

Chapter 3

Cable-Driven Plush Toy Robots

3.1 Introduction

In this chapter, a set of related tools and methodologies for cable-driven soft robots is presented, which covers the aspect of design, simulation, fabrication, and control. A sub-class of cable-driven soft robot, cable-driven plush toy robot, is selected to validate our work in this section. To build a functional cable-driven soft robot, an interactive design software is required to help the user to explore the massive design space of cable placement and to provide non-trivial winch-cable network design to achieve the desired pose assigned by the user. A simulator can be useful to validate and finalize the proposed design. The fabrication pipeline should be not only low cost, but also accessible to non-experts. Last but not least, control algorithms are necessary to pose the robot so that it matches the target desired pose.

This is a joint work with James Bern and Grace Kumagai and advised by Professor Stelian Coros and Professor Nancy Pollard.

3.2 Fabrication

We present in this section the fabrication methodology for cable-driven soft robots, which is inspired by [61]. Note that we have several desired requirements for the fabrication method: The robot should 1) be mainly composed of soft material, which

3. Cable-Driven Plush Toy Robots

is not the case in [61] because a large internal rigid box occupies most of the space in the torso 2) be able to deform easily . Moreover, the overall fabrication method should be low-cost and easily-accessible to non-experts.

Plush toy, one type of soft material, serves as a good media for soft robots. First of all, plush toys are so safe that kids are allowed to play them without much parental supervision. What’s more important, if we want to build a friendly and lovely robot that people want to interact with, plush toy robots have appealing appearances and comfortable textures. Other than using plush toy as the main body, the fabrication method utilizes cheap off-the-shelf components and 3D printed parts to actuate the plush toy. An overview of the fabrication pipeline is illustrated in Figure 3.1

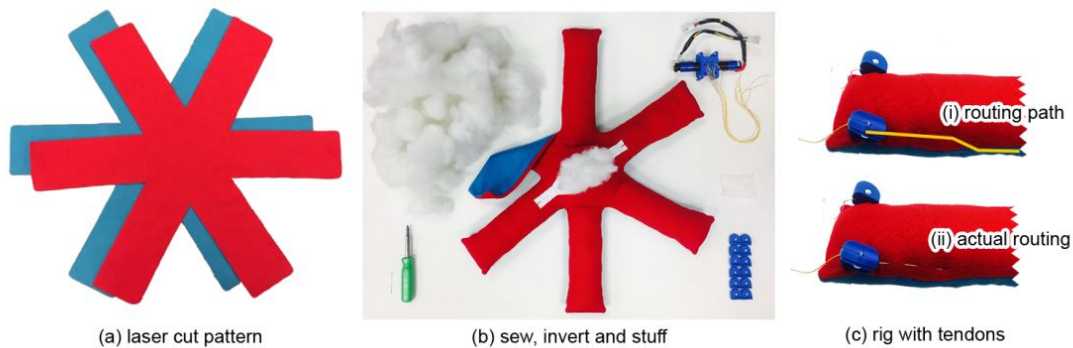


Figure 3.1: Overview of the fabrication process. Laser cut patterns (a) with pinholes are sewn, inverted, and stuffed(b). In (b), we show all the components of the cable-driven plush toy robot and tools needed. Stuffing of one limb is removed to show its volume. A zipper is sewn to allow easy winch assembly. The cables are already connected to the winch and the stoppers (in the bottom right) will be attached to the end of the cables and requires screw driver(in the bottom left) to clamp it. (c) shows a close look up of the cable pattern design and the actual routing.

3.2.1 Body

The main body can be any plush toy purchased online or in stores or it can be easily designed and fabricated using our pipeline. To build a plush toy, we laser cut a user-designed geometry as well as small pinholes out of felt fabric where we later route the cables. Zippers can be sewn to provide easy access to embedded motors.

The body is sewn together with all-purpose sewing thread and inverted to hide seams. Lastly, polyester fiber filling is filled inside the fabric skin.

3.2.2 Cables

Cables are routed throughout the laser cut pinholes with one end attached to an internal winch and the other end attached to a stopper or a knot. Here we use braided fishing line due to its strong tension and low friction. These factors should be kept in mind when choosing the cables as low tension cable breaks when large motor torque is applied and high friction cable hinders the recovery motion which is solely powered by the elasticity of the main body.

3.2.3 Internal Winch

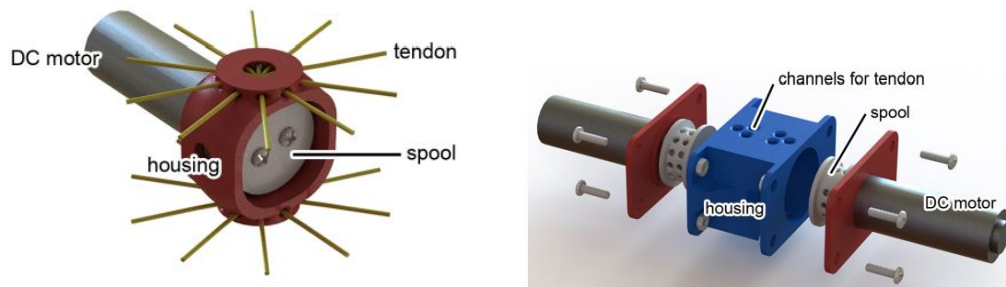


Figure 3.2: An internal winch should include a geared DC motor, a spool attached to the shaft of the motor, and a housing that collects all the cables.

An internal winch, illustrated in Figure 3.2, is the core element of our actuation system. It contains a geared DC motor, a spool attached to the shaft of the motor, and a housing that is mounted on the motor body. Every cable should be collected into internal winches in the way that it first goes through a channel on the housing and then being routed on the spool. One internal winch can collect multiple cables to reduce the mechanical complexity, but the control of these cables will be coupled. Internal winch is the only rigid part, but can be deeply embedded within the soft body.

3.2.4 Stoppers

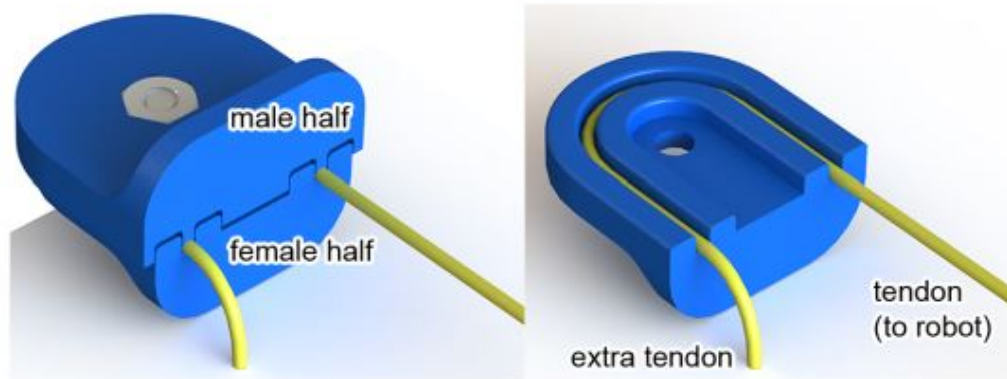


Figure 3.3: The 3D printed adjustable clamping stopper allows easy assembly and fast adjustment. The screw holds the two pieces of the clamp stopper. When screwed tightly, the stopper seizes the cable inside the slot without slippery.

When the spool rotates without stoppers, the cables would be pulled all the way through out the pinholes and eventually detach from the fabric skin. The simplest mechanism of a stopper is a knot at one end of the cables, but there are still chances that the entire knot is pulled through the pinholes. Buttons were considered, but the labor of tying them with cables can be time-consuming and labor-expensive. A 3D printed adjustable clamping stopper, shown in Figure 3.3, is designed to allow easy assembly and fast adjustment. Adjustment is a necessary step when calibrating the cable routing after they are attached to the internal winches.

3.2.5 Cable Sheath

There are designs that we do not want certain regions of the plush toy to deform. For example, we want only the limbs bending on a humanoid plush toy which the internal winch is placed in the center of torso. In such case, we would like to avoid deformation on regions between the limb and the torso. To fulfill this deformation, cable sheaths, shown in Figure 3.4, are incorporated. They work similar to bowden cables, fix the length of the cable between the two ends, and thus prohibit cable contraction within the region. Here we use flexible PTFE tubes.

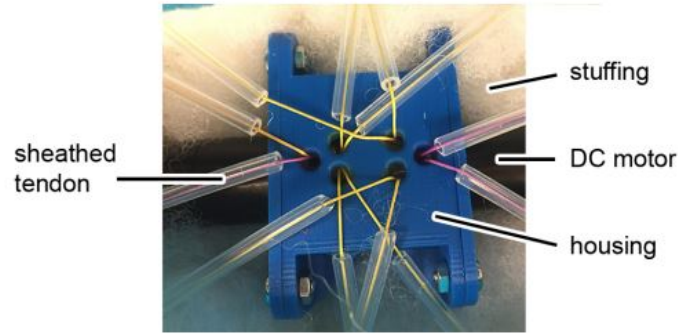


Figure 3.4: Cable sheaths are used to prevent undesirable deformation in particular regions. When all deformations happen far away from the winch, all cables coming out from the winch need to be sheathed.

3.3 Modeling and Simulation

In this section, we present a forward simulation model that predicts the deformation of cable-driven soft robots. With the simulation, the users are able to explore the massive design space and workspace of a soft robot without spending time and effort to actually fabricate and test it.

The underlying physics of soft body motion is basically principle of least total energy. In other words, the configuration of the soft robot x^* under cable contraction α is defined by the minimum of total energy $E(x, \alpha)$, which is the sum of the overall deformation energy of the robot E_{plush} , conservative energy stored in the cable E_{cable} , pin energy stored at the fixed base E_{pin} , and kinetic energy of the robot E_k . This can be formulated as an optimization problem given by

$$x^* = \underset{x}{\operatorname{argmin}} E(x, \alpha) \quad (3.1)$$

$$E(x, \alpha) = E_{plush} + E_{cable} + E_{pin} + E_k \quad (3.2)$$

The gradient of energy with respect to the position of a node x_i gives us the force on that node and the Hessian of energy is the force Jacobian.

$$\begin{aligned}
 F_i &= -\frac{\partial E}{\partial x_i} && \text{for } E = E_{plush}, E_{cable}, E_{pin} \\
 \frac{\partial F}{\partial x} &= -\frac{\partial^2 E}{\partial x^2} && \text{for } E = E_{plush}, E_{cable}, E_{pin}
 \end{aligned}$$

The gradient of the kinetic energy is the nodal momentum, and the total energy equation 3.2 is equivalent to the total force equation:

$$F(x, \alpha) = F_{plush} + F_{cable} + F_{pin} + p$$

, where p is the vector of momentum of each node.

In daily application where the contraction speed of the cable is low, quasi-static assumption can be made and the kinetic energy of the soft robot is set to zero. This static simulation skips the transient response of the soft body motion and directly solves for static equilibrium pose. On the contrary, dynamic simulation should be used when we care about the transitory motion before the robot reaches equilibrium. For example, if we want to simulate contacts between a soft body and other objects, dynamic simulation should be used.

To solve for the least energy configuration of the soft robot using the optimization in Equation 3.1, Newton's method is used for the reason that gradients and Hessians of each energy can be efficiently pre-computed with explicit equations.

3.3.1 Deformation Energy

The soft body motion is modeled using finite element method. The body of the soft robot can be discretized into finite elements by triangulation for 2D meshes or tetrahedralization for 3D meshes. In this chapter, we create examples only in 2D whereas 3D examples are shown in Chapter 4 and 5. The initial un-deformed configuration of the soft robot is denoted as *rest pose* X whereas deformed configuration, given the cable activation α , is denoted as *pose* x . Both X and x store positions of every mesh node in a vector.

In continuum mechanics, the deformation gradient \mathcal{F} of a finite element e can be

computed by $\mathcal{F} = \frac{\partial x^e}{\partial X^e} = \mathbf{d}\mathbf{D}^{-1}$. \mathbf{d} stores the edge vectors of the finite element under deformed pose and \mathbf{D} stores the edge vectors under rest pose, i.e. $\mathbf{d}_i = x_i^e - x_0^e$ and $\mathbf{D}_i = X_i^e - X_0^e$ with i denoting the node index in an finite element. Neo-Hookean material model is chosen to model the soft plush toy body and the equation of energy density is given by

$$\Psi(x, X) = \frac{\mu}{2} \text{tr}(\mathcal{F}^T \mathcal{F} - I) - \mu \ln J + \frac{\kappa}{2} (\ln J)^2$$

where μ and κ are the Lamé parameters, I is the identity matrix, and $J = \det \mathcal{F}$. The total deformation energy of the soft body can be computed by integrating the energy density over its domain, and in finite element formulation the integration is simply summing all contributions from every finite element.

3.3.2 Cable Energy and Winch

We consider cable as a piecewise linear curve that connects a sequence of cable points $x_s, x_1, \dots, x_n, x_t$ inside the mesh, where x_s and x_t are the starting point and end point respectively and in between are called via points. For any arbitrary pose x , the length of a cable is

$$l(x) = \|x_s - x_1\| + \sum_{i=1}^{n-1} \|x_i - x_{i+1}\| + \|x_n - x_t\|$$

In our model, we consider the entire cable to be frictionless, and thus the cable energy is only dependent on the positions of all cable points. There's no requirement that the cable points need to fall exactly on mesh nodes. Any arbitrary point inside a finite element can be parametrized by Barycentric coordinates with respect to the nodes of the outer finite element. Figure 3.5 illustrates some examples of cable placements.

We model each cable as a unilateral spring which energy is piecewise \mathcal{C}^2 polynomial. Instead of e.g. using hard constraints to enforce to enforce cable lengths, we set the cable energies as soft constraints. The contraction of a cable α is the difference in cable length between current pose x and rest pose X , i.e. $l(X) - l(x)$. We assume at rest pose, all cables store zero energy and have zero slack. Given cable contraction α ,

3. Cable-Driven Plush Toy Robots

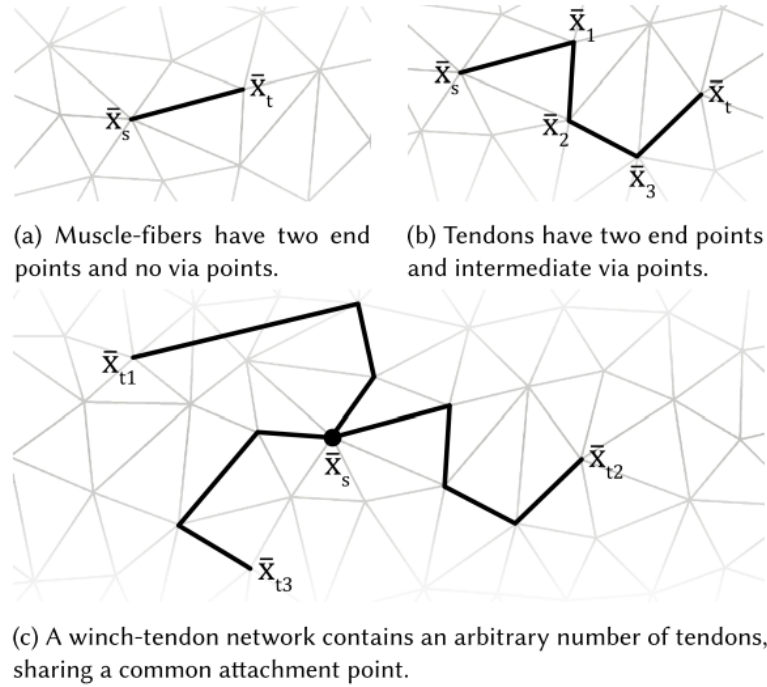


Figure 3.5: Possible placement of a cable. It can be made up of at least two nodes. Multiple cables are are bundled together at the winch.

the cable energy $U(\alpha)$ is defined as

$$U(\alpha) = \begin{cases} 0, & \alpha \leq -\epsilon \\ \frac{K}{6\epsilon}\alpha^3 + \frac{K}{2}\alpha^2 + \frac{K\epsilon}{2}\alpha + \frac{K\epsilon^2}{6}, & -\epsilon \leq \alpha \leq \epsilon \\ K\alpha^2 + \frac{K\epsilon^2}{3}, & \text{otherwise} \end{cases}$$

and plotted in Figure 3.6. K is the spring constant, and *epsilon* is a small constant that specifies the range of cubic region which smoothly interpolates between 0 (meaning no energy stored when slack) and quadratic equation(modeled by Hooke's Law). The tension τ of a cable is given by the derivative of the cable energy with respect to contraction α , i.e. $\tau = \partial U(\alpha)/\partial \alpha$. When a winch reels in (or lets out) a cable, the contraction α increases(or decreases). In cases where a winch collects multiple cables, these cables share the same contraction value.

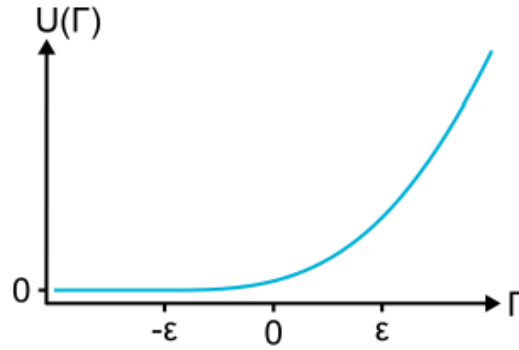


Figure 3.6: Cable energy curve consists of zero on the left, quadratic energy on the right and cubic interpolation in between.

3.3.3 Pin Energy

Pins are placed on points inside the mesh to anchor the soft robot. We model pins as zero-length springs which energy is given by $E_{pin} = \frac{1}{2}K_{pin}||x_{pin} - x_i||^2$, where K_{pin} is a large spring constant, x_i is the position of the node that is anchored to x_{pin} . The pins are assigned by the user and different pin placements will result in different soft robot motions.

3.4 Inverse Kinematics for Soft Bodies

In this section, an inverse kinematics solution for controlling the soft robot is presented based on the static simulation mentioned in the previous section. Inverse kinematics problem is to solve the desired cable contraction to reach some target pose given by the user. The key insight is that the quasi-static assumption made in static simulation establishes a relationship between the pose of the soft robot and the contractions for the cables.

The proposed control scheme is illustrated in Figure 3.7. Through a graphical user interface visualizing the static simulation, the user can specify the desired target pose with drag and drop interaction - dragging nodes on the simulated soft robot mesh and drop them at desired target positions. The desired contractions for achieving the



Figure 3.7: The user specifies the desired pose through a graphical interface and the physical soft robot poses accordingly in real time.

target pose can be computed by minimizing an objective function \mathcal{O} ,

$$\alpha^* = \operatorname{argmin}_{\alpha} \mathcal{O}(x; \alpha)$$

The computed contraction is passed to the physical soft robot through serial communication in real time. An interesting observation of our method is that we do not need to wait until the optimal solution is solved. The intermediate contraction steps can form a reasonable trajectory of motion that ends up at target pose.

To express our objective, which is setting the simulated node positions x to desired positions x' , we can formulate our objective function as

$$\mathcal{O}(x; \alpha) = \frac{1}{2}(x - x')Q(x - x')^T$$

where Q is a diagonal matrix that encodes whether a node is given a desired target position or not. This objective function penalizes any deviation of node positions from the target positions.

To solve our optimization problem with a simple gradient-based approach, we need to compute the gradient of the objective function with respect to contractions, which can be decomposed using chain rule:

$$\frac{\partial \mathcal{O}}{\partial \alpha} = \frac{\partial \mathcal{O}}{\partial x} \frac{\partial x}{\partial \alpha}$$

where the first term can be trivially computed while the second term is unknown.

To solve for the second term $\frac{\partial x}{\partial \alpha}$, the quasi-static assumption is used to provide the relation between x and α . The mathematical form of this assumption is basically the sum of all forces is zero-

$$F(x, \alpha) = F_{plush} + F_{cable} + F_{pin} = 0$$

The total derivative of the above equation gives us

$$\frac{dF(x, \alpha)}{d\alpha} = \frac{\partial F}{\partial \alpha} + \frac{\partial F}{\partial x} \frac{\partial x}{\partial \alpha} = 0$$

The first component $\frac{\partial F}{\partial \alpha}$ only considers the cable forces as they are the only forces that depend on contractions. By rearranging, we can express cable forces with contractions $F_{cable} = A\alpha$ and substitute A into $\frac{\partial F}{\partial \alpha}$. From the definition $F = -\frac{\partial E}{\partial x}$, the coefficient $\frac{\partial F}{\partial x} = -\frac{\partial^2 E}{\partial x^2}$ is the negative of Hessian. Plugging in the two terms gives us

$$\frac{\partial^2 E}{\partial x^2} \frac{\partial x}{\partial \alpha} = A$$

The Jacobian $\frac{\partial x}{\partial \alpha}$ can be solved numerically and then the gradient of the objective function can be computed. We solve iteratively the desired contraction with inverse kinematics and the corresponding pose with forward simulation.

3.5 Winch-Cable Design Automation

In this section, we build a design tool that automatically generates a physically realizable winch-tendon network given some target poses. The pipeline is illustrated in Figure 3.8. This design task can be non-trivial as it requires iterative design to explore different winch-cable networks and validate them. This can be very difficult especially when the target pose is complex and non-trivial.

Our design process is inspired by muscular hydrostats - a biological structure composed of only muscles without skeleton support. There are examples of this kind of soft structure performing complex motions, such as elephant trunks, human tongue, starfish, octopus, etc. These tiny muscles (here we call them muscle-fibers) can activate independent to each other and the large degree of freedom is the key to

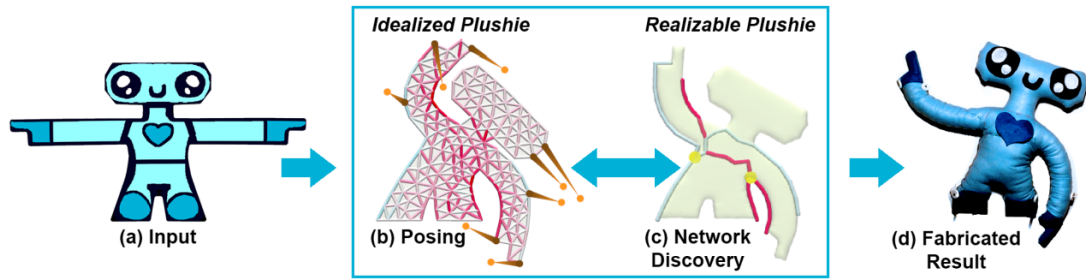


Figure 3.8: Our design tool bridges the gap between desired motion and realizable winch-cable network. The input is the rest pose of the soft robot. After the user poses the soft robot to desired target pose, our system automatically discovers a realizable winch-cable network that generates an approximation of the target pose.

performing complex motions. Unfortunately, a mechanical system of this complexity is not realizable. Simulations, however, are not bound to fabrication constraints. Our approach starts with simulating soft robot as if it has many such independent muscle-fibers, observes the co-activation pattern of the muscle-fiber network, and extracts physically realizable winch-cable network composed of muscle-fibers that contribute most to the motion.

We consider a muscle-fiber network that matches our finite element discretization, and thus each finite element represents a muscle-fiber. After the user set the desired target pose, a co-activation pattern, called activation graph, can be constructed based on the tension value in each finite element. The activation graph provides us how much contribution each muscle-fiber made to the pose, as shown in Figure 3.9 (a).

A thresholding process can be done to greatly reduce the amount of candidate muscle-fibers that will be selected to finalize our winch-cable network. The activation graph after thresholding is illustrated in Figure 3.9 (b). We start with the muscle-fiber that has the largest tension and extend the muscle-fiber by connecting the nearest muscle-fiber candidates that are sufficiently close. The result of this joining process is called winch-tree, as shown in Figure 3.9 (c). Several parameters can be explore to generate different results, such as maximum amount of candidate muscle-fibers after thresholding, maximum allowed distance for a muscle-fiber to be connected to the winch-tree, maximum amount of connection steps, etc.

At this point, we are left to decide where to place the winch and how many of

these winch-trees are finalized as cables. It can be difficult to assemble if too many cables are collected in a single winch. Our approach is to linear search all possible winch placements and select the best winch-cable network based on a score function and simple heuristic with the concept of "total contraction". The total contraction of a candidate cable is the sum of all tensions of all the muscle fiber along the cable. A single winch with multiple cables will have multiple corresponding total contractions. Based on the heuristic, we want to find the winch-cable network that contributes most to the motion and thus has the greatest mean total contractions. Furthermore, we also want to penalize the variance of total contractions for the reason that all cables collected into a single winch share the same contraction. From these two heuristic, a reasonable score function is $\phi Mean(T) + (1 - \phi)Variance(T)$, where ϕ is a user-tunable parameter in range $[0, 1]$. The final winch-cable network generated by our design system is shown in Figure 3.9 (d).

3.6 Results

We used our design system to generate six cable-driven plush toy robot designs, four of which are fabricated and controlled with our inverse kinematics method. Table 3.6 summarizes the specifications and the figure indices of the designs.

	muscle fibers	target constraints	winches	cables	fabricated	Figure
robot	414	8-11	2	6	✓	3.8
pig	571	2	1	2	✓	3.10
E	442	6	1	3	✓	3.11
S	276	4	1	2	✓	3.12
puppy	307	3-6	3	3		3.13
squid	429	4-8	8	11		3.14

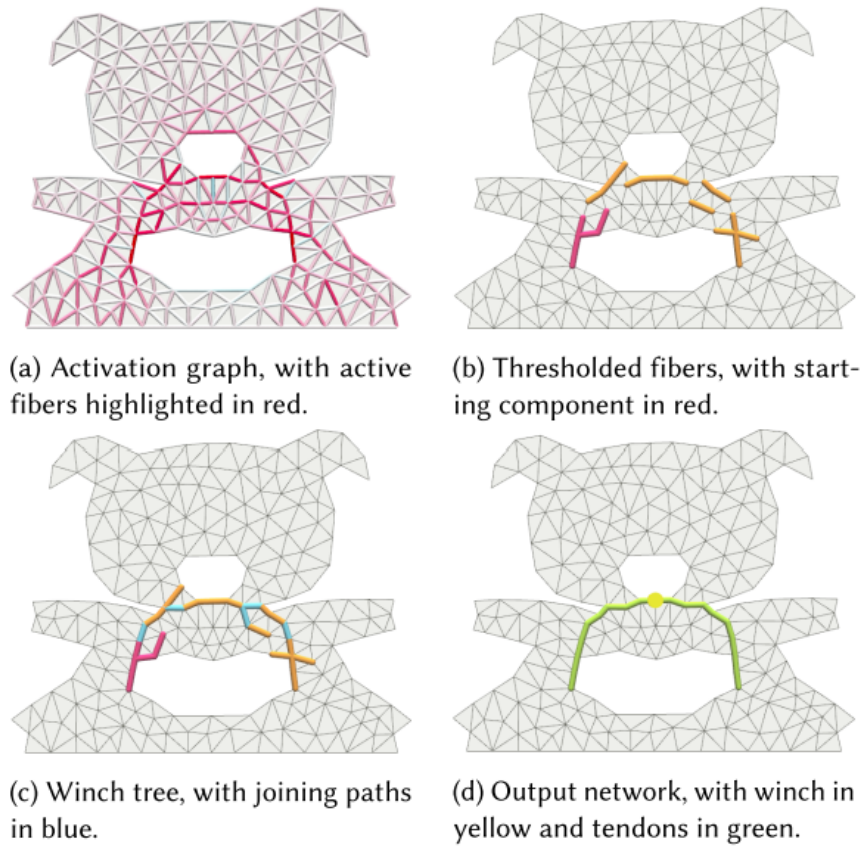


Figure 3.9: Process of extracting realizable winch-cable network from activation graph.

3.7 Discussion

3.7.1 Comparison between Fabricated Robots against Simulation

In all the result figures, the deformations of the fabricated robots match those in the simulation. Nonetheless, differences exist for two reasons. Firstly, in our model, no friction is considered between the cables and the fabric skin nor between the fabric skin and the ground surface the robots lie on. Secondly, serious hysteresis effect in motion can be observed on the fabricated robots. In other words, they do not perfectly recover to the rest pose when all the cables are released. This is due to the

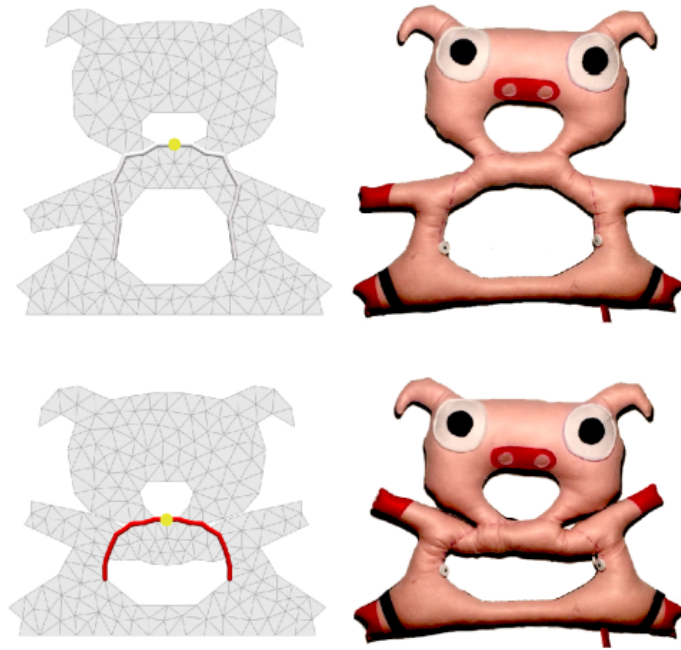


Figure 3.10

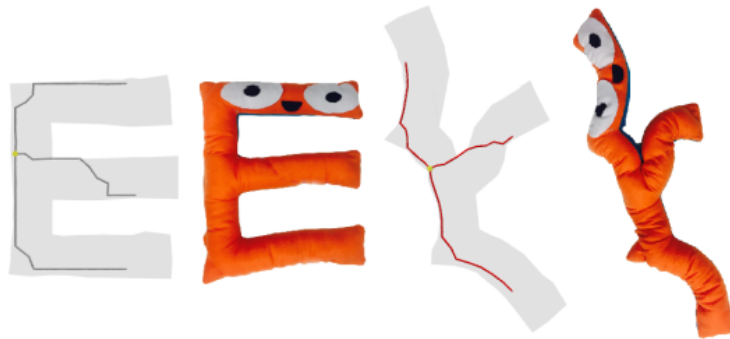


Figure 3.11

fact that the polyester filling can clump and rearrange in an irreversible way when compressed too much. Based on this observation, foam-based stuffing is explored in Chapter 4. Other approaches can be taken to alleviate this effect, such as using low-friction cables and fabric skins.



Figure 3.12

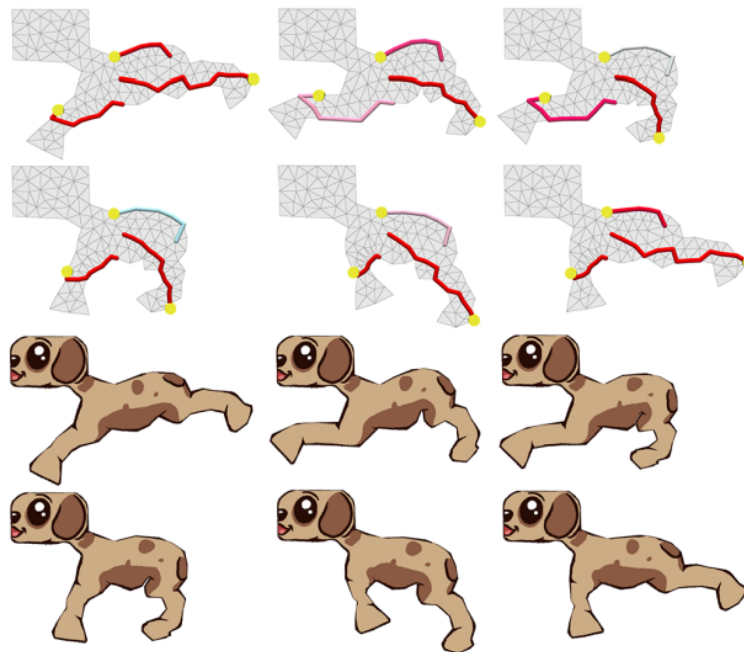


Figure 3.13

3.7.2 Future Work

Though the concept of our design system is to generate the best winch-cable network according to a specific target pose, a combination of multiple winch-cable networks can form a complex motion space. Figure 3.13 shows a sequence of different poses making up a complex locomotion. We also find that in practice the more complex the winch-cable network is, the less likely it can be reused for other motions. An interesting future work is to explore the winch-cable design with respect to the motion

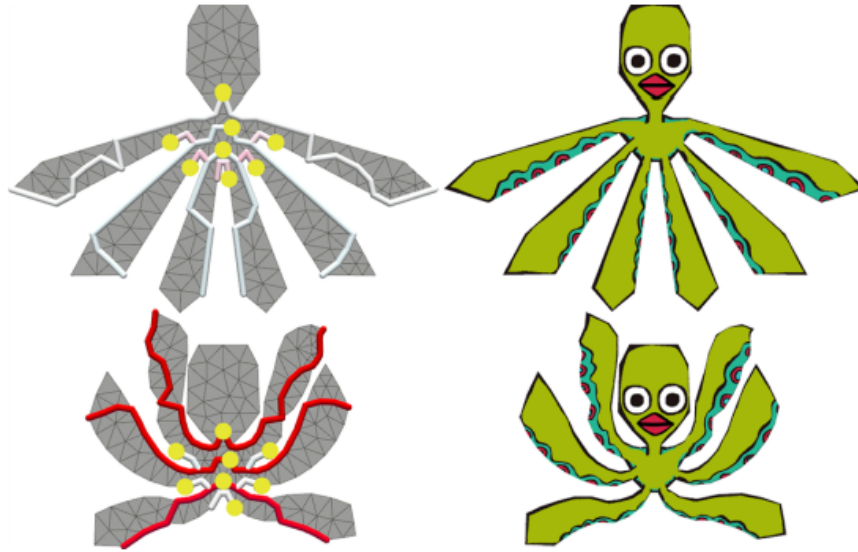


Figure 3.14

space. For instance, if the user has several target poses to reach, what is the least amount of winch-cable networks that are required to reach all poses and how much more design space a new winch-cable network can provide?

3.8 Summary

In this chapter, we present a set of universal tools and methodologies for cable-driven soft robots. It includes a cheap public-accessible fabrication method, finite element based simulator to forward simulate the motion, a method to solve inverse kinematics based on the simulation, and a design system that outputs the optimal winch-cable design for target poses.

3. Cable-Driven Plush Toy Robots

Chapter 4

Cable-Driven Foam Casting Robots

4.1 Introduction

The scope of this thesis falls in a broad class of cable-driven soft robots. Plush toy robot is only an instance of this class. In this chapter, we would like to present another subclass of robot that also belongs to this class. The need comes from the fact that plush toys have undesired non-negligible hysteresis defect. Without including it into our method, this defect is bearable only when the approximation of the motion is of interest. For instance, entertaining or companion robots often do not need to perform precise control to fulfill their purpose, such as dancing, non-contact interaction, etc. However, a huge part of robotics application, like locomotion and manipulation, requires high precision. Building upon the structure of plush toy robots, we use a foam body to replace the stuffing that causes the hysteresis defect, and we call these robots *foam robots*. Though the fabrication pipeline is the same for other shapes, to later explore the precise control problem, an anthropomorphic foam hand, which takes the shape of a human hand, is built. For convenience, We call this robot *foam hand* later throughout this thesis.

This is a joint work with Jonathan King who helped experiment on the fabrication pipeline.

4.2 Fabrication

An initial rest pose needs to be determined at the beginning. For the foam hand, a neutral pose between fingers full extension and flexion is chosen. Different rest poses should be considered along with the range of desired motions. To duplicate the hand shape, the user keeps their hand in the rest pose in a bath of alginate until the alginate fully cures. Advantages of using alginate are 1) it cures in less than 5 minutes during which human hand needs to be posed still, and 2) it is safe for human skin exposure. Then the hand is removed and a plaster of paris positive is poured and allowed to cure for several hours before demolding. The end result is a high-fidelity plaster of paris replica of the human hand shape in the rest pose. This plaster of paris casting is only needed to prevent human hand posing fixed for long hours, so if the initial rest pose of soft robot comes from other physical objects, we can skip the steps in this paragraph.

Then using this plaster of paris replica, object of the rest pose shape, or a 3D printed object from a CAD design, we build a 2 piece silicone (*Mold Max 30*) mold. Silicone mold is required because alginate molds are sensitive to aging, temperature, and humidity. Unless only one foam robot is needed, we recommend using silicone mold. The 2 piece silicone mold is created with traditional method:

1. Surround one half of the object with clay
2. Pour the silicone to form the first half of the mold and let it fully cure
3. Turn the mold over and remove the clay
4. Pour the silicone to form the second half of the mold and let it fully cure before demolding

The curing time for the silicone we use is 24 hours but the process can be sped up to 6 hours under 60°C in oven. Degassing the silicone mold in vacuum chamber to eliminate air bubbles does not affect the functionality of the mold. The fabrication steps for making the alginate mold and silicone mold for foam casting is summarized in Figure 4.1.

Once we have a mold, foam can be casted by mixing the the two part urethane foam compound (*FlexFoam-iT!* ®*X*) with a household electric egg mixer and pouring the mixed liquid into the mold. The foam can be fully cured within 2 hours under

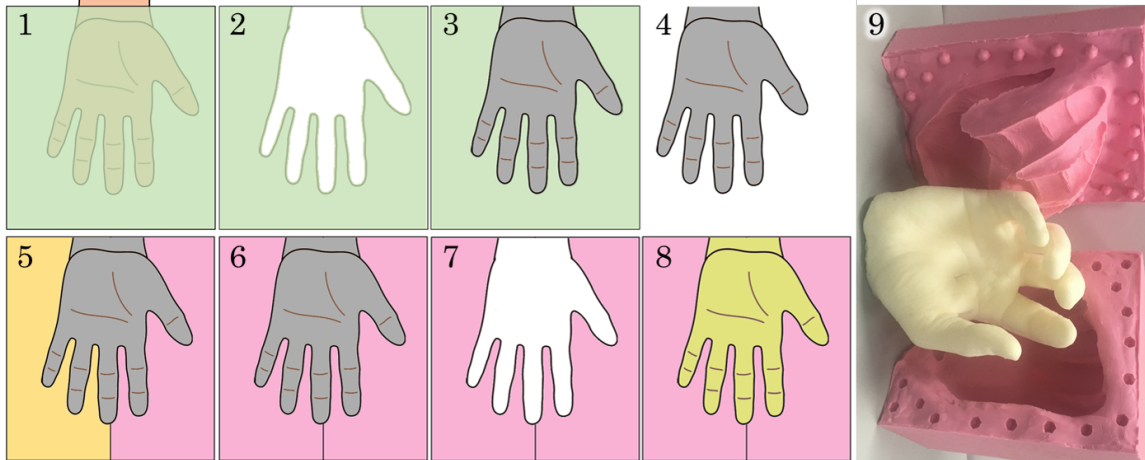


Figure 4.1: Fabrication Process: 1) Cast of human hand in alginate, 2) Remove hand from alginate mold, 3) Pour plaster of paris into alginate mold and allow to cure, 4) Carefully remove plaster of paris cast from alginate mold, 5) Cover one half of plaster of paris hand with clay and the other half with silicone, 6) Remove clay, apply mold release agent to silicone and fill second half with silicone, 7) Remove the plaster of paris hand from two part silicone mold and clean the mold, 8) Use the two-part silicone mold to cast foam hands, 9) Final result.

room temperature. The silicone mold costs approximately 100 US dollars and the foam core only costs a few dollars each. The fabrication steps have to be easy and fast and the ingredients have to be cheap for soft robots to be widely used among researchers or even the non-expert public.

The final step is to put the foam core inside a fabric skin on which, similar to making plush toy robots, we sew cables. For human hand shape, off-the-shelf knitted gloves can be used. For other complicated shapes, sewn knits from cut fabric or eventually by automatic digital knitting process [35] can be used. One extra step is to apply additive spray (*3M Super 77*) to prevent slipping between the foam core and the fabric skin. Cable placement patterns can be designed by the user or by the automatic design system introduced in Section 3.5. Though the motors can be embedded into an empty chamber inside the foam core, we mount the motors away from the soft robot for convenience and extend the cables sheathed with PTFE tubes from the motor winches to the end point of the cable designs. An acrylic base is built to fix the positions of the motors and the PTFE tubes. Figure 4.2 shows two complete setups of the foam hand. The setup on the right is designed to allow convenient

replacement to different foam cores.

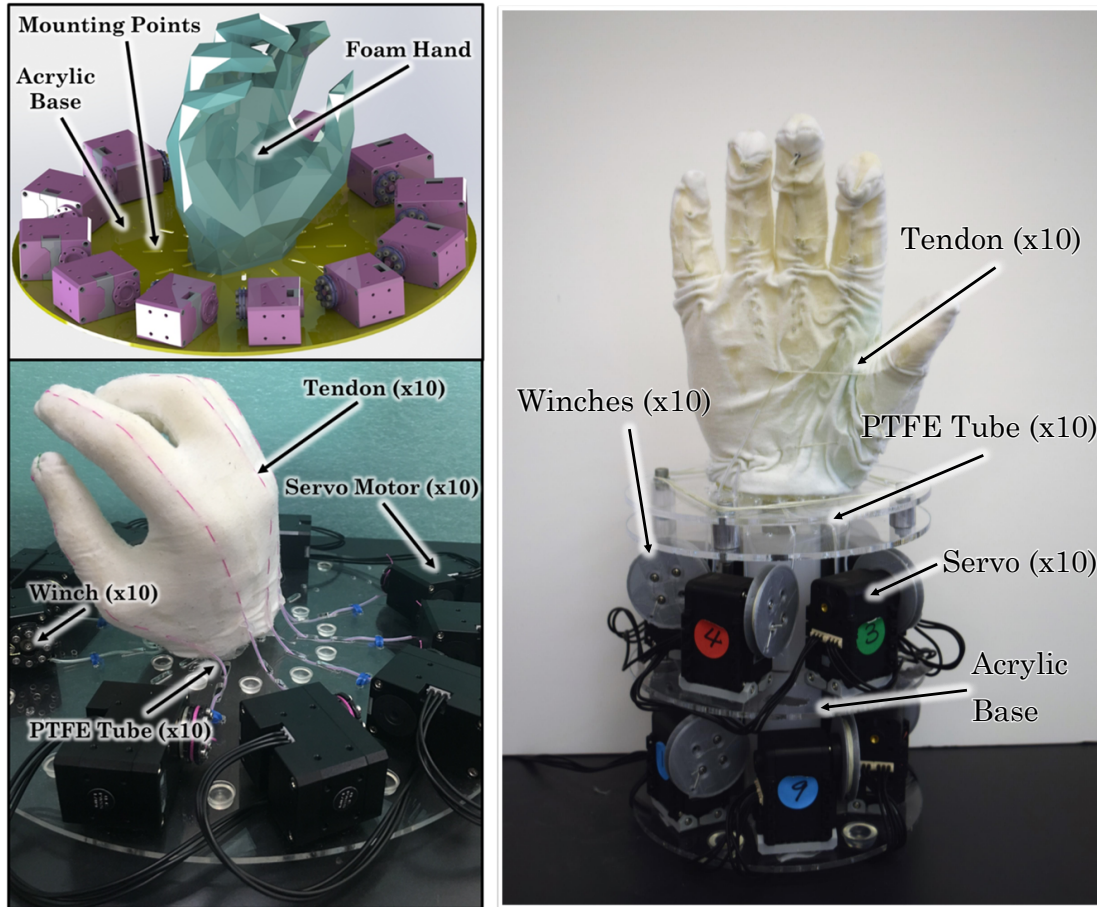


Figure 4.2: Assembled foam robots with different setups.

4.3 Compatibility to Design Tools

In Chapter 3, we present a set of tools that help users design winch-cable networks, forward simulate soft robot motions and solve optimal contractions using inverse kinematics. These tools are based on the finite element simulation of the soft robot, which requires a mesh model.

Soft robots originally designed by CAD software are cast in molds built with 3D printed objects and the digital CAD model and be directly imported into the simulator.

As for soft robots cast in molds built with physical objects, digital representations of such objects need to be created. With 3D reconstruction technology, this is accomplished by taking 50 photos uniformly around the static object and upload them to a software called *Autodesk ReMake*, which outputs a surface mesh of the object in .stl format. This surface mesh is then fed into *TetGen* to create a solid 3D finite element mesh that can be imported into the simulator. It should be noted that for systems with limited computation power, a simplified mesh can be created with *MeshLab*. Figure 4.3 shows a comparison between the 3D reconstruction of a human hand and the corresponding finite element mesh which contains 916 mesh vertices and 3030 finite elements.

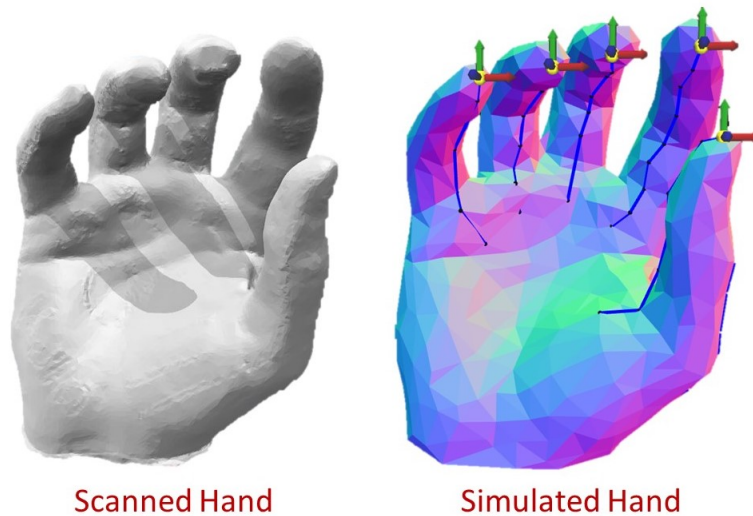


Figure 4.3: Left: Scanned hand using AutoDesk ReMake and 50 photos. Right: Finite element mesh imported into the simulator

4.4 Fabricated Results

Three foam hands are built using the above fabrication pipeline, two of which are anthropomorphic foam hands with different rest poses and cable patterns, and the other one is a non-anthropomorphic foam hand designed with CAD software. They are shown in Figure 4.4. The left foam hand in a cupping pose is our first design which cables are placed intuitively. The middle foam hand has an initial rest pose

4. Cable-Driven Foam Casting Robots

of extended fingers to fully make use of the range of motion. It also uses a different cable pattern on the thumb because the human thumb has a distinct motion space from the rest of the fingers. Figure 4.5 shows the improvement in thumb mobility by changing the cable pattern. The right foam hand is designed by CAD software and has a similar shape to industrial four-fingered robot grippers. In Figure 4.6, we show that the poses in simulation matches the poses on a physical robot given the same cable contractions.



Figure 4.4: Rest poses of three foam robots. From left to right: 1) Anthropomorphic hand in a cupping pose; 2) Anthropomorphic hand in extended pose; 3) Non-anthropomorphic hand in caging grasp pose.

4.5 Capability of Foam Manipulator

In this section, the capability of the foam robot hardware is evaluated without intelligent control strategy. We offer the user a graphical rotating knob as well as an input text box to set the motor positions which are linearly mapped to cable contractions. After motor positions are set, the motors will reach the input positions according to its velocity profile. The users can set motor positions one at a time or save several combinations of motor positions as keyframes. These keyframes will be linearly interpolated to form a smooth motion trajectory once the users hit the run button.

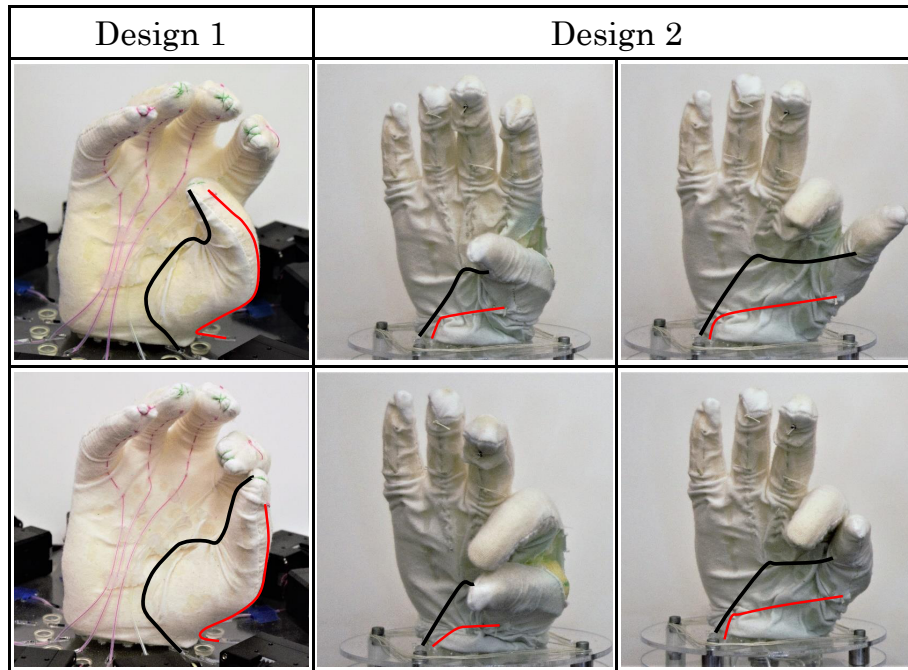


Figure 4.5: Cable patterns affect the motion space of the foam robots. In Design 2, the cable patterns on the thumb is designed for more realistic motions.

4.5.1 Grasping

All foam hands shows capability of grasping, and the two anthropomorphic foam hands can mimic human grasps. Figure 4.7 shows some example grasps using three different foam hands. It is worth noting that the precision grasp shown in the middle can be hardly achieved by rigid robots or other types of soft robots. We attribute this capability not only to the compliance of the soft foam, but also the complex shape of the foam hand.

4.5.2 Strength of Grasps

One common concern about soft robots is the lack of stiffness to resist disturbances. To test this capability, the object is pulled away from the grasp of foam hand. A spring scale is attached to the object to measure the disturbance force. The experiment results are shown in Figure 4.8. The anthropomorphic hand can resist 250 gram of disturbance force while the non-anthropomorphic hand can resist up to 600 gram of

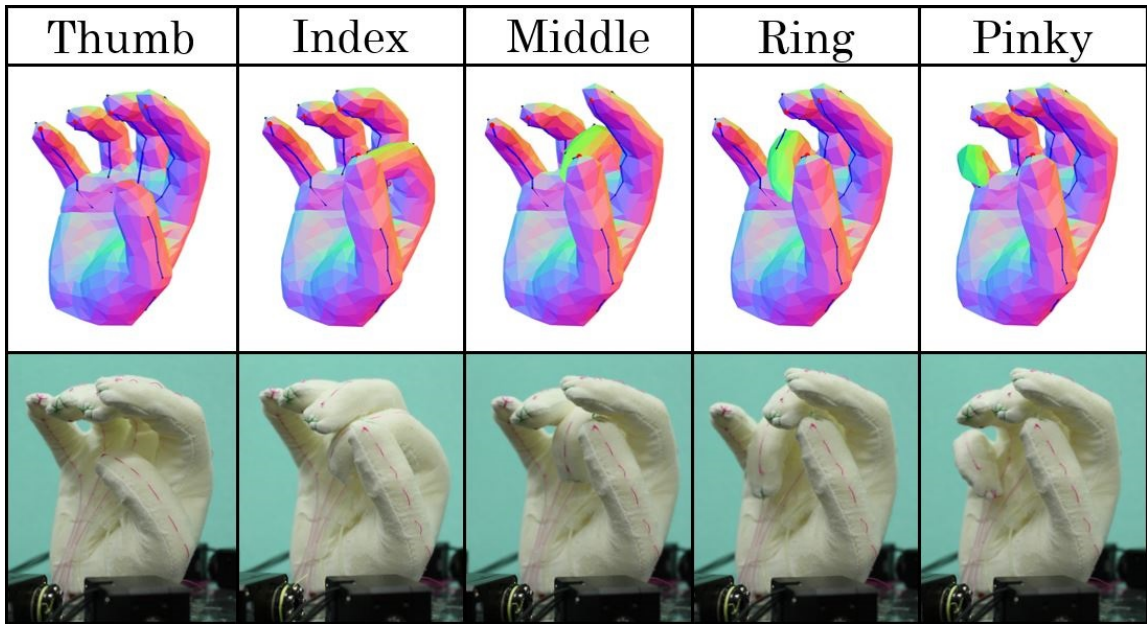


Figure 4.6: On the upper row shows the poses designed in the simulator. On the bottom row are the poses of the physical foam hand using the same contraction.



Figure 4.7: Demonstration of static grasping with a glue bottle (left), a screwdriver (middle) and a box cutter (right).

disturbance force. This capability can be further enhanced by improving the shape of the foam hand, the cable patterns, and the material of the fabric skin which affects friction forces.

4.5.3 Manipulation Task

Besides grasping, the foam hands are also capable of performing complex manipulation tasks, even with simple keyframe-based position control. Figure 4.9 shows a sequence

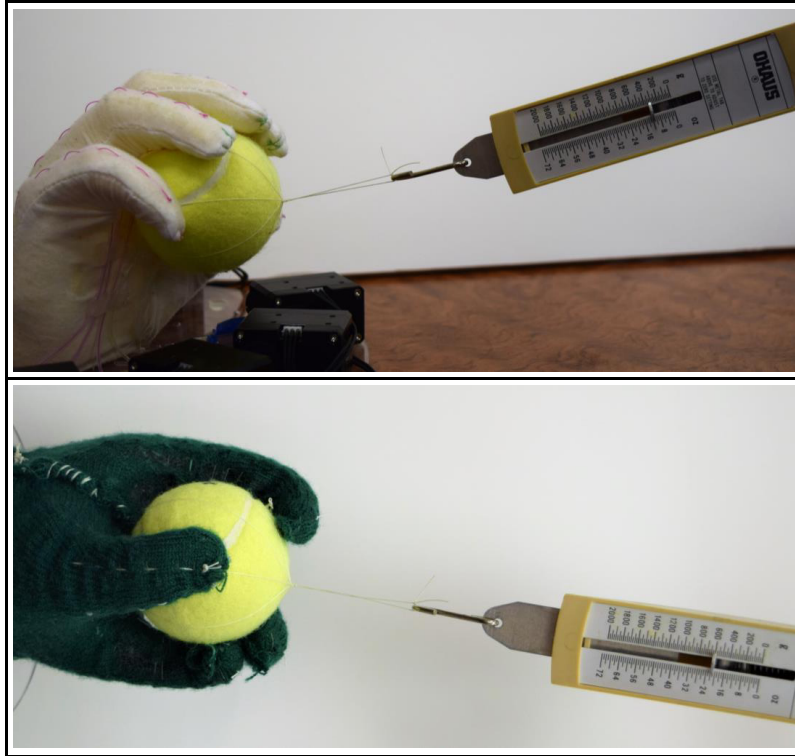


Figure 4.8: The strength of grasps was measured by pulling on a tennis ball under power grasp until failure.

of images of manipulation tasks performed by different foam hands. These manipulation tasks are difficult due to the need of fine balance, continuous contacts, and moderate forces to provide friction and counter-gravitational support. The advantage of soft robots is amplified in these dexterous manipulations and it can be difficult for rigid robot counterparts to accomplish these tasks with position control or force control.

Given the complex shape of the foam hand and the large cable pattern design shape it has, different strategies can be used to perform the same task. Figure 4.10 shows how three different foam hands can rotate a ball. Rigid robots with limited degree of freedoms or soft robots with simple geometry does not have this large and redundant motion space to accomplish the same task with multiple strategies.

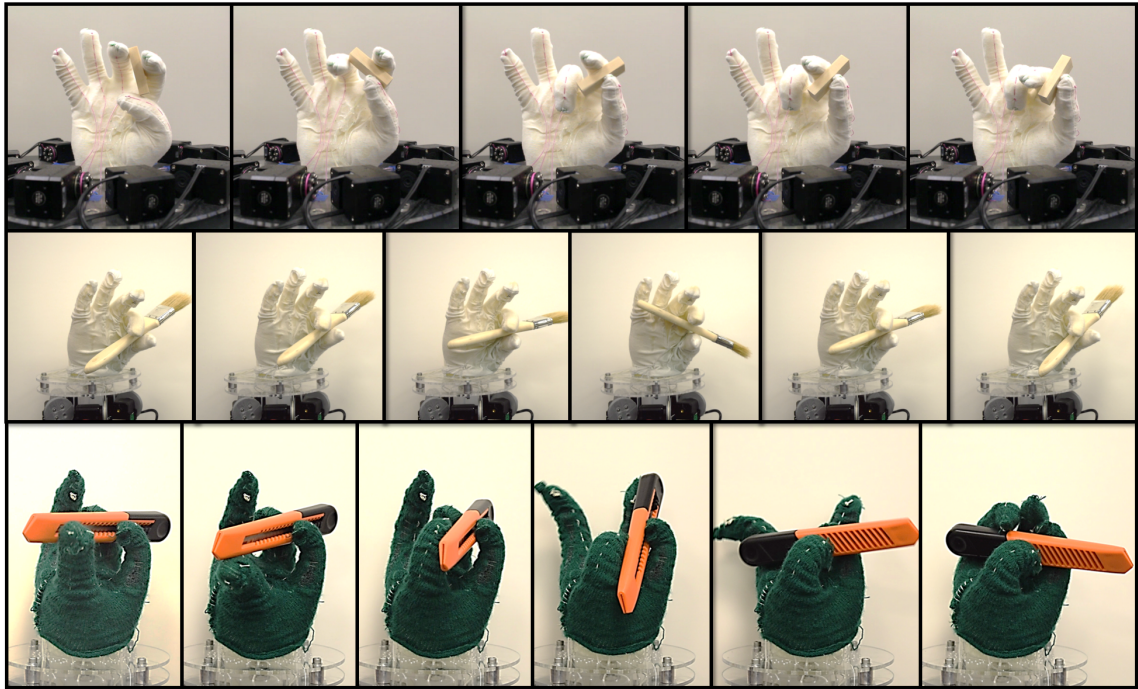


Figure 4.9: In-hand manipulation sequences of three foam hands. Top: Lateral Grasp Transition. Middle: Rocking Motion. Bottom: Utility Knife Spinning.

4.6 Comparison with other types of Soft Robots

It is important to know how foam robots are compared to other soft robots in the aspects of actuation, material, and fabrication. This section also serves as the Fabrication Section in Background Chapter.

4.6.1 Actuation

There are three main categories of actuation systems that are commonly used for actuating soft robots: electro-active polymer(EAP), fluidic actuation, and variable-length tendons [28].

First proposed by Wilhelm Rontgen in 1880s, EAPs are polymers that performs size or shape changes under electric field and can be categorized into electronic EAP and ionic EAP [25]. The main difference is that the requirement of the activation electric field is higher for electronic EAP. Ionic polymer-metal composite(IPMC), a

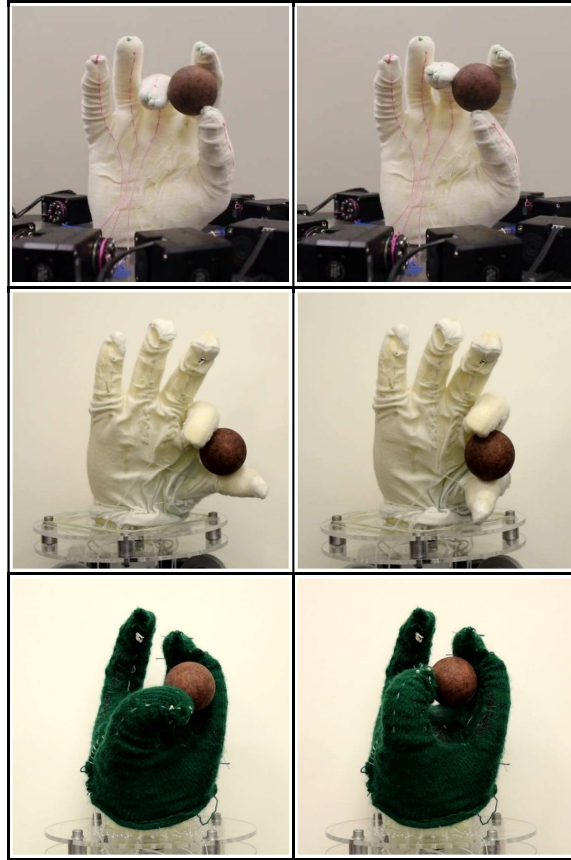


Figure 4.10: Three distinct foam hands performing precision in-hand manipulation (twisting a ball).

type of ionic EAP, can perform large deformation with relatively low electric field and thus is safer and more efficient. Some disadvantages of EAPs include requirement of encapsulation in open air or underwater environment. Though it is expected that patterned IPMC can provide multi-degree of freedom motions[28], current IPMC studies have only explored simple bending motion [22][38] due to the fact that IPMC works in the cantilevered form[3]. This largely restricts the motion space provided by a single actuator and thus complicates the actuation systems.

Fluidic actuation, which is widely-spread and has most reports on successful applications, can be either pneumatic or hydraulic. An early version of pneumatic actuator is the pneumatic artificial muscle(PAM), also known as McKibben actuator[14]. It is composed of deformable elastomer tubes wrapped with fiber sleeves. A more popular

type of fluidic actuator is fluidic elastomer actuator, also known as Pneu-Nets(PN). PN creates deformation by expanding embedded channels with pressurized fluid. Unfortunately, what is usually not shown in photos of these fascinating fluidic-actuated soft robots is the air compressor. It takes a lot of space, generates heat over time, and has continuous vibration. A recommended air compressor for beginners on [4] can weigh 2.4 lb and has a volume of 5.9" \times 2.1" \times 4.5". This inevitable limitation forces the robot to be tethered with tubings and restricts the mobility. Furthermore, it conflicts the major advantage that robot components are soft and safe. Such fluidic-actuated robots are also fatal to leakage. Another fluidic actuation is achieved by combustion power[56] or gas-generating chemical reactions[40]. These sources of power, if not safely designed, can be dangerous to non-expert public.

Variable-length tendons are embedded in the soft body to exert controlled force and generate deformation to a certain segment. [48][32][27] uses shape memory alloy(SMA) as actuators. SMA has an advantage that the actuator itself is soft, but it is not readily developed yet due to the defects in expensive price, poor fatigue property, and slow cooling speed [47]. Before these challenges are solved, SMA is not considered as a satisfactory actuator. Embedded cables controlled by spooler motors are also commonly used for actuating soft robots[9][13][61]. This technique is chosen to be more desirable over all other methods. Conventional motors are well-studied and heavily-used and many off-the-shelf hobby motors are cheap and accessible to non-expert public. Though rigid, motors are usually small enough to be deeply embedded into soft bodies.

4.6.2 Material

Here we compare flexible foam with silicone, rubber, granular materials, and air when used as the main material for building soft robots.

The most commonly used materials for building soft robots are silicone and rubber due to their soft nature[49][9][57][32][27]. [49] uses a mixture of silicone-based materials, *Ecoflex 00-30 Smooth-On Inc.* and polydimethylsiloxane(PDMS, *Sylgard 184, Dow Corning*) as the main body of their soft quadrupedal robot. Here we specifically compare them with the flexible foam(*FlexFoam-iT! X, Smooth-On Inc.*) we use. Table 4.6.2 summarizes some crucial material properties and fabrication

parameters. Though foam robots also require a silicone mold, the cure time for foam casting is a lot shorter than silicone casting and doesn't need oven for high temperature. Moreover, in terms of specific gravity, which the lower the better and safer, foam of the same volume can be six times lighter than silicone and PDMS.

	Cure Time	Specific Gravity (<i>g/cc</i>)
PDMS	48 hrs @ 25°C	1.03
Ecoflex 00-30	4 hrs @ 25°C + 2 hrs @ 80°C + 1 hr @ 100°C = 7 hrs in total	1.07
FlexFoam-iT! X	2 hrs @ 25°C	0.16

Granular materials is a common media to create variable stiffness mechanisms. This mechanism exhibits high strength in jammed state and allows the granular media to flow in unjammed state [7] [13]. Various granular materials are used and tested in [13] to evaluate their material property under compression load and their result is shown in Figure 4.11. They concluded that grounded coarse coffee bean is the most ideal media since its stress-strain curve shows a plateau region and thus stiffer behavior under compression. They also stated that flexible foam behaves similarly to sawdust, which locally collapses before densifying to become a solid mass. For soft robot application, flexible foams are more preferred because the stiffness(i.e. compressive modulus) of coffee bean in both jammed and unjammed state are larger than that of flexible foams. This property for grounded coffee bean is computed by multiplying the density and the effective compression modulus per density both given in the paper $445kg/m^3 \times 6.8kN - m/kg = 3026kN/m^2 = 3.026MPa$ while the effective compressive modulus of a flexible foam with a similar density of 0.24 is $77psi = 0.531MPa$ given in [43]. These granular media also have a larger density of $0.445g/cc$ as well as hysteresis effect, similar to polyester fiber stuffing used in Chapter 3 and [61], due to the positional rearrangement of particles.

Inflatable robots can be viewed as soft robot which main body is air encased in a soft boundary skin[44]. The work uses thermal welding machines to seal straight seams on plastic films. Hot air welding machine is also suggested to fabricate curved

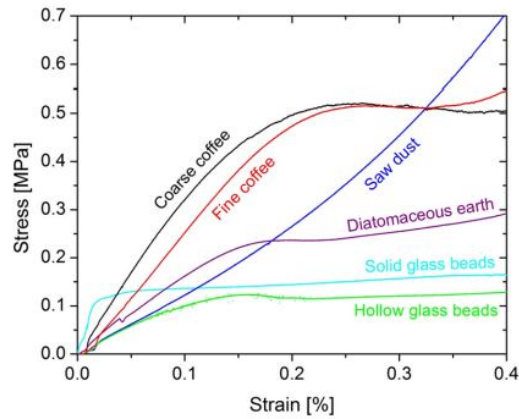


Figure 4.11: Compressive stress vs. strain data for the grains tested. [13]

seams. However, these machines are expensive to buy and not accessible to public individuals. An alternative and more accessible approach is heat sealing, which uses any heat sources (steam iron, soldering iron, hot air gun, etc.) to meld and external forces to seal the plastic films. This seemingly trivial method turns out to be difficult for untrained people. Figure 4.12 is an inflatable robot built with the alternative method. The imperfect seams result in unappealing appearance as well as rough surfaces. Other than fabrication difficulty, inflatable robots are fatal to air leakage caused by long-period operation or penetration. Without holding enough air inside the bladder, no longer can it maintain its shape. Due to the light weight of inflatable robots, they lack the resistance against perturbations and capability to maintain stability. For human robot interactions, the tactile touch of foam (for example, a household foam pillow) is more preferred than that of air bladder (likewise, an inflatable travel pillow).

After comparing multiple materials used in fabricating soft robots, we can conclude that foams possess advantages of porous structure like granular media and light weight like inflatable robots. It is also cheap, fast and easy to fabrication, and available in multiple stiffness and densities.

4.6.3 Fabrication

Taking the advantage of advancement in rapid fabrication techniques, researchers have created many soft robots of different kinds. 3D printing is a powerful and



Figure 4.12: An inflatable robot built using solder iron for sealing the plastic cloth.

widely-used technology to generate physical objects given digital designs. It can be used to print plastic molds for silicone casting [32] as well as the body of the soft robot using soft materials[58]. Molding and casting is simple and common when using soft materials such as silicone rubber or flexible foam[49][9][57][32][27]. Soft lithography [60], a technique used to create multi-gait soft robot[49], enables complex patterns to be created on polymer materials, which can be served as channels for fluidic actuation systems.

4.7 Summary

In this section, we improve the hysteresis effect of plush toy by introducing foam robots. The fabrication pipeline of foam robots is fast, low cost, and accessible to non-experts and the design tools developed in Chapter 3 are compatible to foam robots. We demonstrate fabricated results with capability of complex motions and manipulations under manual control. Finally, we compare our fabrication method to previous works.

4. Cable-Driven Foam Casting Robots

Chapter 5

Learning-based Control on Cable-Driven Foam Manipulator

5.1 Introduction

In Section 3.4, we present an inverse kinematics(IK) solution for cable-driven soft robots based on finite element simulation. However, in real-world applications, finite element simulation may not always be accessible, especially to non-experts. For instance, a user is given a soft robot and would like to control it right away. Without a simulator, the previously proposed solution is no longer valid. Instead, with data-driven approach, we can use learning-based approach to learn a model that takes desired configurations as input and outputs corresponding actuations. Two types of learning are experimented in this section. One is supervised learning and the other is reinforcement learning. Both techniques have shown successful performance in similar cases. Supervised learning is applied to solve the IK on a simple 2 degree of freedom tentacle-shaped soft manipulator in [21] and [20]. Recently, [42] has reported successes in using reinforcement learning to solve IK and manipulation tasks on a rigid robot arm and a rigid robot hand. Both techniques will be applied for solving IK on our cable driven foam hands.

5.2 General Formulation: State, Action

In this section, we formalize some common terms later used in the thesis. The soft robots on which we apply our control methods are the three foam hands in Figure 4.4.

States: The state of the foam hand fully represents the deformation of the current pose and is denoted as s . Feature points are selected at each finger tips and their positions in 3D space are concatenated to form the state. Take our anthropomorphic foam hand as an example, we have a total of five fingers and therefore $s = [s_1, s_2, s_3, s_4, s_5]$ and thus the dimension of input is 15. More feature points can be selected to provide finer representation of the deformation, but here only the five finger tip positions are considered.

Actions: The action a , unless specifically denoted, is not an incremental step, but the absolute continuous cable contraction, which is linearly normalized to $[-1, 1]$. Note that this definition breaks the dependency of the policy on current state in inverse kinematics(IK) tasks. That is, the absolute action will pose the foam hand to a unique pose regardless of the initial pose.

5.3 Deep Inverse Kinematics Model

5.3.1 Introduction

We first explore using a deep neural network to learn the IK on the foam hand using supervised learning. Two situations are considered. In the first situation, a neural network model is trained in simulation. The trained neural network can be later distributed to many non-expert users and the users can control their soft robot of the same design simply by forward propagating the neural network. The other situation has no access to simulator and the only data we can collect is from the physical robot. The former learns in simulation and the latter learns on real robots. The main difference comes from the difficulty in collecting data in real world, which can be laborious and time consuming. Therefore, different models are proposed to deal with different sizes of training data available.

5.3.2 Learning in Simulation

In this subsection, the finite element mesh of the anthropomorphic hand with rest cupping pose, shown in 4.3, is used to validate our method.

The IK problem can be formulated into a supervised learning problem with data-driven approach. By randomly sampling different cable contractions and running forward simulation to obtain the resulting poses, we can collect a database of actions and resulting states. Figure 5.1 visualizes these sampled poses. A deep IK model can be built with a feed-forward neural network(FNN) whose input is the resulting states and output is the actions. The FNN will fit the inverse mapping from resulting states to the corresponding actions by minimizing the mean squared loss and during testing, outputs the corresponding cable contraction(action) given the desired target pose(state). Note that starting pose does not matter to the process because the output action, which is not incremental but absolute cable contractions, determines the resulting pose independent to the starting pose.

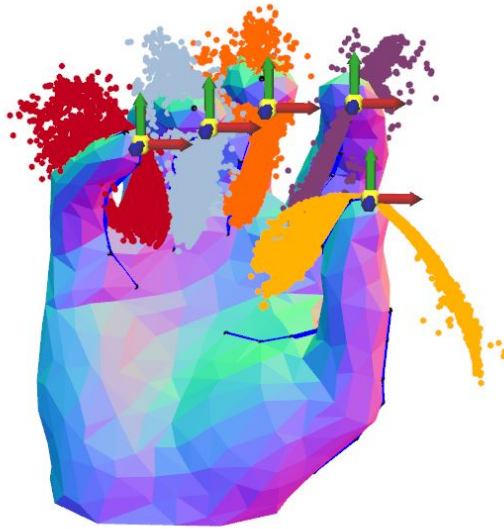


Figure 5.1: Visualization of fingertip positions of the randomly sampled poses.

The neural network we use with supervised learning, shown in Figure 5.2, has four intermediate layers with 30 units each and ReLU non-linear activations. The output activation is $\tanh(x)$ activation to match the action range $[-1, 1]$. Arbitrary amount

5. Learning-based Control on Cable-Driven Foam Manipulator

of randomly sampled data is fed into the network and is trained with 300 epochs, batch size 20, and Adam optimizer. In this work, we compare models trained with 10,000 and 100,000 sampled data among which 90% of the data is used for training while the rest is used for validation. The model that has a lower loss in validation set is always saved. During test, the same test data set of data size 5,000 is used for comparison. The learning curve and some qualitative visualization are shown in Figure 5.3 and 5.4-5.5. Both FNN losses converge to a value under 0.02 out of the cable contraction range of $[-1, 1]$. In the visualization, the left pose is the target pose assigned by the user and the right pose is actuated by the absolute cable contractions generated by the deep IK model. The visualization shows that the FNN successfully learns how to reach the target pose.

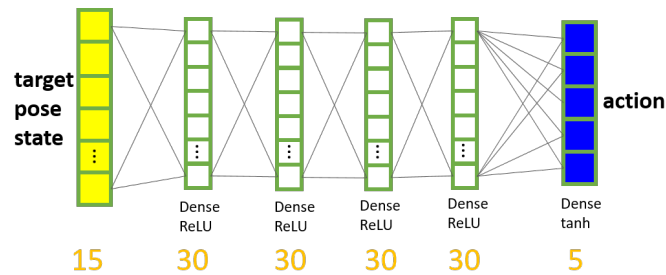


Figure 5.2: Neural network structure for deep IK model

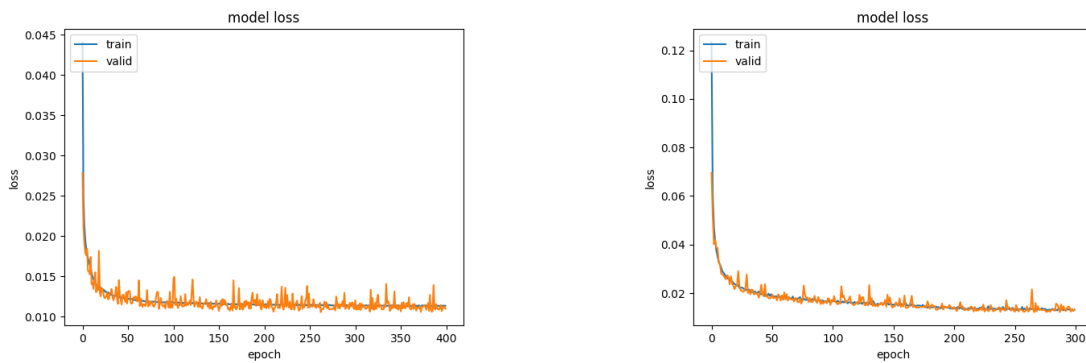


Figure 5.3: Learning curves with data set size of 100,000(Left) and 10,000(Right).

To evaluate the similarity between the target pose and generated pose, we run 5,000 poses for testing and compute the average distance error for each feature points.

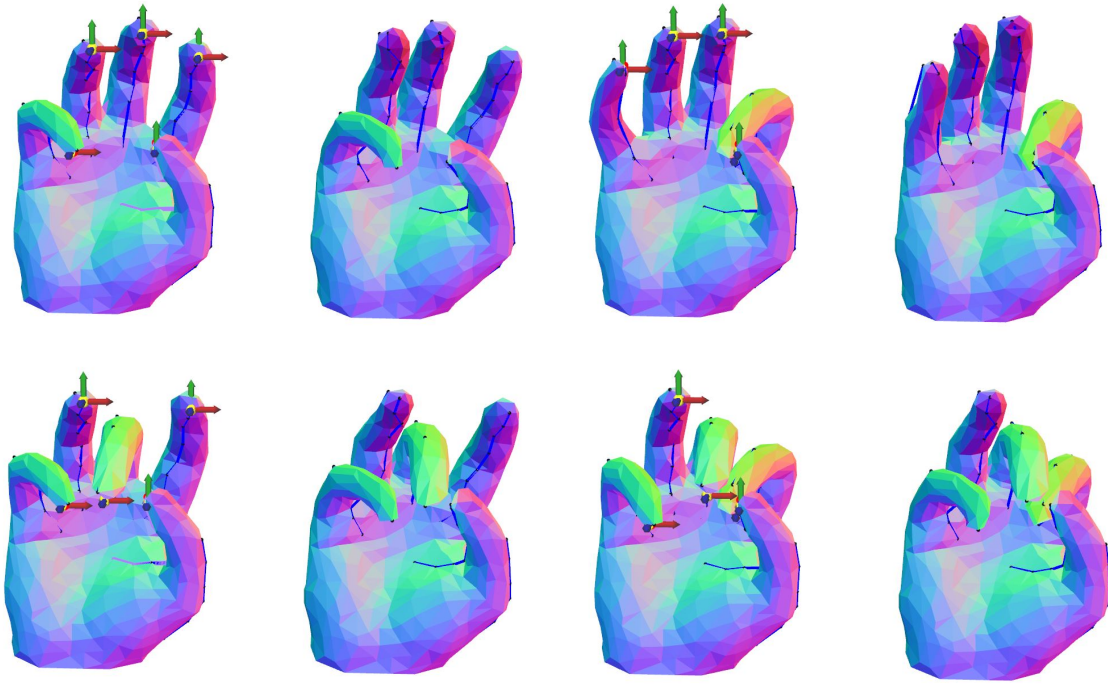


Figure 5.4: Four pairs of user-designed pose (Left) and the FNN generated pose (Right)

The average distance errors for each finger are listed for the two FNNs trained with 10,000 and 100,000 data. The average distance error of ~ 0.6 cm is acceptable for approximation, but is not satisfactory for industrial application where precise control is required. Though training with more data does improve the average error, the improvement is relatively small. The limitation is inside the nature of function approximator that to fit a more complex function, a more complex model and more data are needed.

	thumb	index	middle	ring	pinky	average
10,000 train data	0.4559	0.5780	0.5498	0.8395	0.6245	0.6095
100,000 train data	0.4438	0.5878	0.5213	0.7869	0.5848	0.5849

Table 5.1: Average distance error (in cm) on test data

We compare our implementation detail and results with two similar works [21] and [20] that also uses neural network for solving IK problem in reality and simulation

5. Learning-based Control on Cable-Driven Foam Manipulator

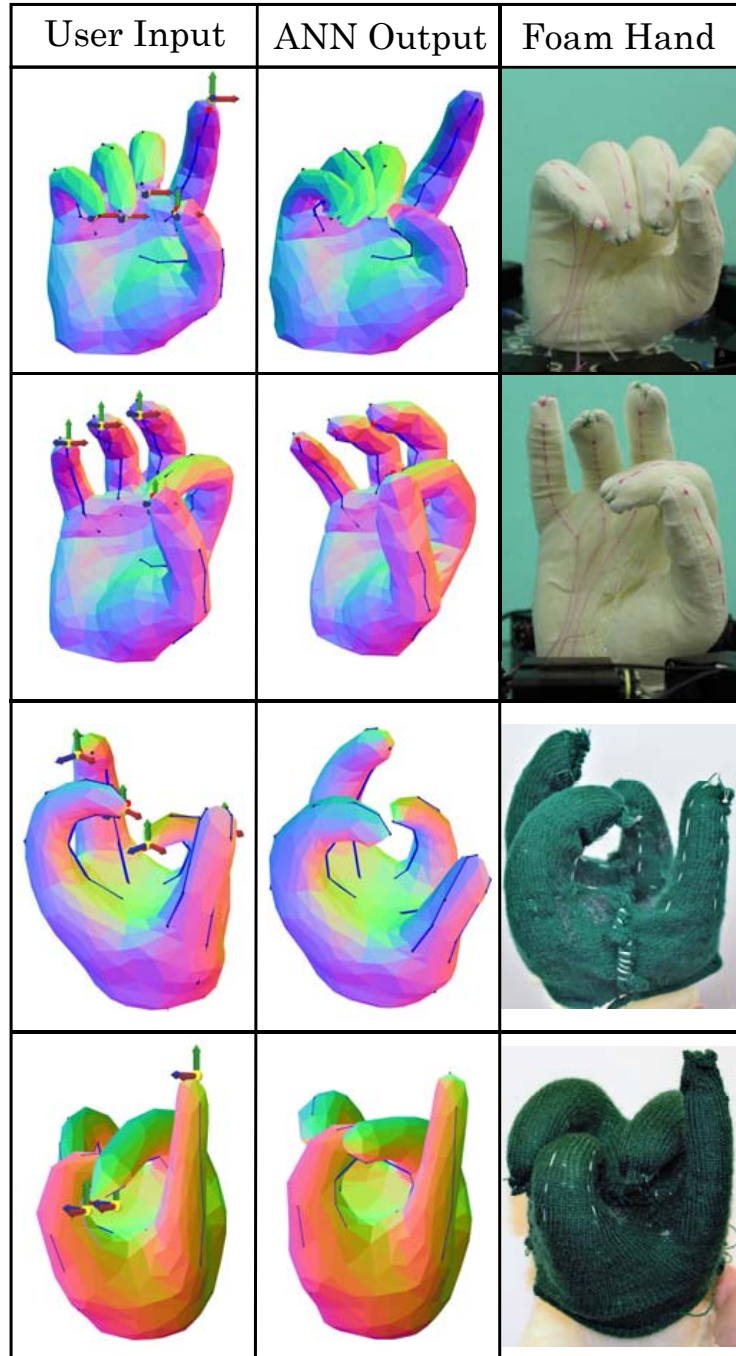


Figure 5.5: Comparison of user inputs, FNN generated poses, and realized poses on foam hands.

respectively, but on a cable-driven tentacle soft manipulator with 2 degree of freedom and in 2D motion space. There are several differences between our implementation and theirs:

1. Task domain:

They solve IK in 2D motion space with a 2 degree of freedom tentacle soft manipulator. We solve IK in 3D space with a 10 degree of freedom anthropomorphic soft manipulator.

2. Physical quantity of the output of neural network:

Their neural network outputs cable tensions in Newton. Our neural network outputs cable contractions in centimeter.

3. Neural network structure:

Neural networks used in [21] and [20] consist of 1 hidden layer with 6 and 34 nodes respectively and $\tanh(x)$ activation. Other learning techniques used include early-stopping method, pre-processing on input data, and momentum. Our neural network consists of 4 hidden layers with 30 nodes each and $ReLU$ (Rectified Linear Unit) activation. The model is saved whenever the performance on validation set improves. No momentum is applied on gradient and the position inputs are directly fed into FNN without pre-processing.

4. Hyper parameters:

	[21] approach	[20] approach	Our approach
number of epochs	13000	25000	300 & 400
number of train data	405	500	10,000 & 100,000
momentum coefficient	0.001	0.015	0
learning rate	0.04	0.01	0.001

The result reported in [21] and [20] is compared with our result in Table 5.2. Learning from data collected from [21] performs worst due to large bias caused by the real-world sensors. In the perfect world of simulation, we solve the IK problem on a more complicated soft robot with higher degree of freedom. In particular, the coupled motion between the fingers can be hard to learn. We confirm the supervised learning approach is able to solve the IK problem on our anthropomorphic foam hand in simulation.

	[21] result	[20] result	Our result
mean (mm)	7.35	4.2	6.095
std (mm)	4.72	2.8	0.607
max (mm)	22.22	12.3	7.352

Table 5.2: Result comparison with [21] and [20]. The former learns from data collected in reality while the latter collects data in simulation.

5.3.3 Learning on Real Robots

In this section, we present IK solution specifically for cases when simulator is not available. Solutions proposed in previous sections use simulator either to directly solve cable contractions from optimization formulation, or to collect simulated data for neural network training. Without a simulator, IK can still be solved using data-driven approach which data is collected directly on physical robots. Therefore, In this case, we want a model that learns with as least data as possible to reduce the time and effort to run data collection on the physical robot. Neural networks are definitely not ideal due to the fact that they require a massive amount of data for learning. As a result, a simpler model, *Kernel Ridge Regression* with linear kernel in Keras (a high-level neural networks API), is applied. The model performs linear regression with L2 Ridge regression.

The state representation in simulation contains finger tip positions, which can be obtained with sensing systems in real world, such as visual cameras used in [21]. *CyberGlove* that contains 22 bend sensors are used to capture the pose configuration. The *CyberGlove* can fit directly on anthropomorphic foam hands. Furthermore, to control foam robots that are not of human hand shape, human users can wear the *CyberGlove* and control with creative hand gestures. Even though the shapes of the foam hand and human hand are not the same. The model is capable of learning the mapping between foam hand pose space and human hand pose space.

The regression model takes the 22 bend sensor values as input and learns to generate the cable contractions. During data collection, the user mimics the shape of the foam hand and stores the corresponding sensor values on the *CyberGlove*. Trained with a small data set composed of 120 poses, the model predicts the actions in validation set with an average root-mean-square error of 0.0026 (note that the

tendon contraction range is $[-1, 1]$).

The performance of the trained model on the anthropomorphic hand is shown in Figure 5.6. The figure shows the capability of reproducing the pose of human hand on the foam robot.

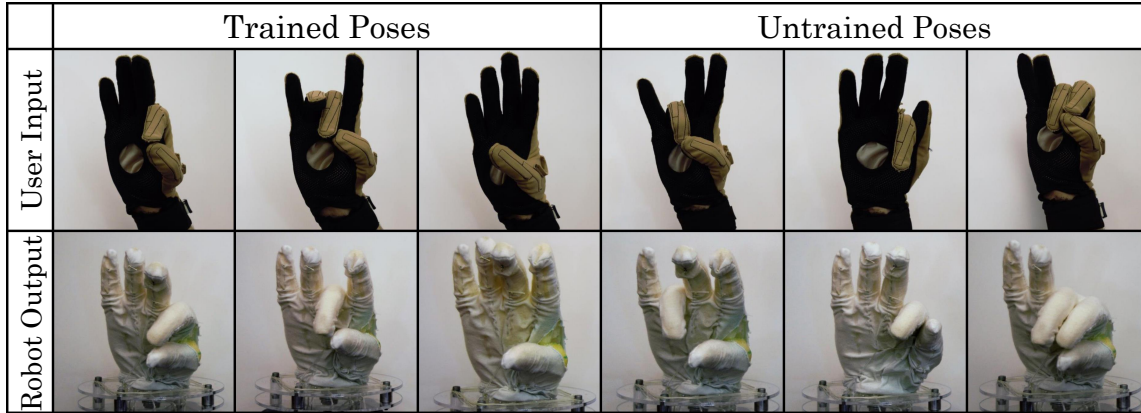


Figure 5.6: Top) Input poses from user wearing a *CyberGlove*. Bottom) Output poses from the learned regression model. Left) Poses taken from the training set. Right) Poses not included in the training set.

5.3.4 Application: Tele-Operation

A useful application of this IK work is tele-operation on soft robots. With the model trained in simulation or on real robots, we can compute the cable contractions of the poses in each time frame. Therefore, we can tele-operate the foam hand in a position-based key-frame control manner. The user can control either with the graphical user interface in the simulation or directly with the *CyberGlove*. Figure 5.7 shows an example of two keyframes in a pen-rolling motion. This motion is achieved by first compute the cable contractions of the two keyframes using the trained model and linear interpolate the motor position values to form a smooth motion in between.

5.3.5 Conclusion

In this section, we use supervised learning to train two kinds of model, a neural network and a regression model, to learn the IK of soft robots. The neural network can learn from massive data set generated by the simulator while the simpler regression

5. Learning-based Control on Cable-Driven Foam Manipulator








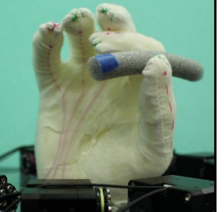
	Contraction	Extension
User Input		
ANN Output		
Foam Hand		
Foam Hand with Pen		

Figure 5.7: Pen Rolling: Row 1) User defined keyframe poses, Row 2) FNN generated poses, Row 3) Realized poses on physical foam hand, Row 4) Manipulation action performed by physical foam hand.

model is recommended when data is collected on real robot. The static IK can be used not only to pose the foam hand statically, but also for tele-operation to achieve smooth and complex manipulation motions, such as rolling a pen.

5.4 Deep Reinforcement Learning

5.4.1 Introduction

Based on the success of learning IK on both rigid robot arms and hands in [42], reinforcement learning(RL) algorithms are expected to transfer successfully to soft robots. In particular, we apply the method that combines deep deterministic policy gradient(DDPG) and hindsight experience replay(HER) in [42] in this section and stick to the anthropomorphic foam hand with rest cupping pose, shown in Figure 4.3 for evaluating the results.

We continue using the terms defined in Section 5.2 and add the following new terms related to RL and HER:

Goals: The goals for each feature point \hat{s}_i are the target positions of each finger tip. Simply speaking, the goal g is the state that represents the goal pose with the same dimension, 15.

Tasks: Our task is to generate an action such that the actuated pose is similar to the target pose. The similarity is measured by the average of distance error $d = 1/5 \times \sum_{i=1}^5 |s_i - \hat{s}_i|^2$. A task is successful if this average distance error is smaller than some tolerance ϵ , i.e. $d < \epsilon$.

Rewards: Here two types of reward are discussed, sparse and binary reward and shaped reward. The sparse and binary reward returns 0 if the task succeeds, otherwise returns -1. The shaped reward is manually designed as the negative of the average distance error, $-d$.

Hyperparameters used in [42] serves as a good starting point, and their results will be used to compare with our result. The core RL algorithm used is DDPG [31], which is a model-free, off-policy, actor-critic algorithm that enables learning policies in high dimensional, continuous action space. Furthermore, we incorporate the idea of multi-goal RL [46] and hindsight experience replay(HER) [1] to generalize the policy to random target poses.

The main concept of HER is to learn from unsuccessful experiences, which is a

common strategy for human to learn things. If the goal can be parameterized, an unsuccessful roll-out to our goal (here we call it desired goal) can be viewed as a successful roll-out for a different goal (here we call it achieved goal). Take IK as an example, a roll-out that ends up in a final state different from the desired goal state, is successful if the goal state is the final state. In this case, the final state can be viewed as the achieved goal state. Therefore, the IK problem can be turned into a multi-goal RL problem which takes not only the current state and action but also the goal state as input.

Different achieved goals can be considered and the end state is just one option. As described in the paper [1], an action taken at some state can be treated as a valid policy to some goals like end state, arbitrary future states, or previous states. The result of the paper shows selecting additional four goals from future states works best and we will use the same strategy. Therefore the roll-out in the experience buffer stores information of states, actions, desired goals, and achieved goals.

In [42], their results show that learning complex robotics manipulation tasks with sparse and binary reward can be difficult without HER, but with HER, sparse and binary reward is more preferred than shaped rewards. In addition to manipulation, they also test the algorithm on IK for rigid robot arms and hands (FetchReach-v0 and HandReach-v0 respectively). For FetchReach-v0 case, using shaped reward performs better than using sparse reward both with and without HER, but for HandReach-v0 case, only using sparse reward with HER successfully learns within 50 epochs. As a result, we will start with sparse reward to learn our soft robot hand IK task. Further comparison will be discussed in detail in Section 5.4.3.

In conclusion, the main method would be DDPG combining HER. Experimental detail, such as parameter tuning as well as curriculum learning will be described in the next section.

5.4.2 Experiments on Reinforcement Learning

One-step or Sequential Decision Making Formulation

IK problem can be formulated as either a one-step decision making or sequential decision making. Here we use T to denote the number of maximal steps in each episode. One-step decision making ($T = 1$) means only one action is taken in an

entire episode while sequential decision making ($T > 1$) allows taking multiple actions in each episode.

We start from the parameters used in [42] except for the action noise which is set to 0.01, five percent of the total action range $[-1, 1]$. Here Figure 5.8 shows two learning curves for both one-step ($T = 1$) and sequential ($T = 10$) decision making formulation with the tolerance set to 1cm. The amount of actions in the entire training process is set the same for comparison baseline (i.e. one-step formulation performs 10 times the amount of roll-outs than sequential formulation). Note that during testing, optimal policy is applied while during training, epsilon greedy policy is applied, which means that there's a probability of $\epsilon = 0.3$ to take random policy for exploration. From the learning curve, we can observe that the sequential formulation learns much faster than one-step formulation. We believe that allowing the agent to try multiple times for the same goal helps the learning.

Regardless of the better capacity in preserving successful experience, policy learned in sequential formulation still learns to finish the task as if it is using one-step formulation. More specifically, the first action a trained policy takes will finish the IK task and ignore the further steps allowed during training. This feature is expected because the best policy that maximize the reward should reach to the goal with first action and avoid receiving further penalty in the future steps.

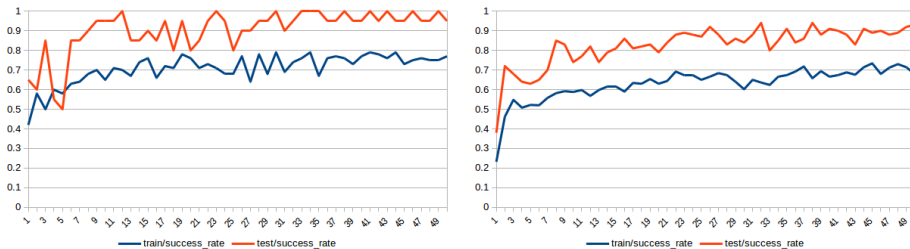


Figure 5.8: Learning curves with $d = 1cm$. (Left: $T = 10$, Right: $T = 1$). Note that during training, epsilon-greedy policy is exploited with $\epsilon = 0.3$.

Curriculum Learning

Even though the above learning with both formulations reach success rate larger than 0.9, the average error tolerance(1cm) is too large to be useful in robotics application where accurate IK solution is highly demanded for precise control. However, setting

5. Learning-based Control on Cable-Driven Foam Manipulator

average error tolerance with a fixed small value can be unrealistic for the reason that the task is too hard that during training the success rate is always zero and no successful experience is learned. Of course, this problem can be solved with longer training time and more computation power. However, unlike [42] using 16 CPUs, in this work only one single machine with 1 CPU is used. To prevent long training hours, curriculum learning designed manually is incorporated. Figure 5.9a uses a curriculum that divide the tolerance by 2 every time the agent is able to reach 0.9 success rate with current tolerance. The steep curriculum ends up disastrous results with a massive success rate drop when starting the second task ($d = 0.5cm$). The result is equivalent to start learning the second task from scratch as the policies for two tasks differ a lot. Here a success curriculum is presented: The tolerance is multiplied by 0.9 every time the agent is able to reach 0.9 success rate with current tolerance. As shown in Fig 5.9b, the agent performs fine-tuning on the policy and learns the harder task successfully.

To expedite the learning process and reach a better accuracy, the critic learning rate is increased from 0.001 to 0.01 and the same curriculum is applied. Figure 5.9c shows the learning curve for this setting. The agent learns tasks of different tolerances ($d = 1 \rightarrow 0.9$), but the learning stops improving for tolerance 0.81. This is the same with the Adam optimizer parameter t is reset to 0 every time the agent starts learning a harder task to avoid learning rate decay. Furthermore, if the training starts learning with tolerance 0.81, it can still reach success rate 0.9 after 50 epochs. Thus, in this case, the curriculum hurts the learning and the agent is stuck at a local minimum.

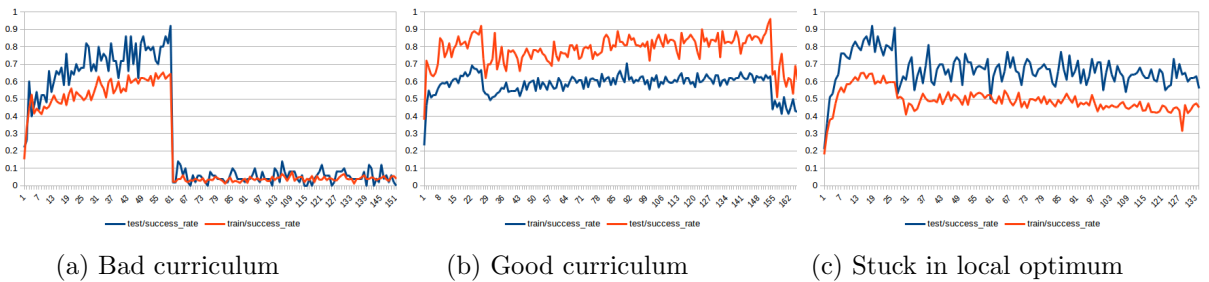


Figure 5.9: Learning curves with curriculum learning. Note that during training, epsilon-greedy policy is exploited with $\epsilon = 0.3$.

Reward Shaping

Multiple changes are made to solve the local minimum problem. First of all, Instead of using the neural network in [42] (3 intermediate layers with 256 units each), we switch to the same smaller neural network structure (4 intermediate layers with 30 units each) which is used for deep IK model. Secondly, even though the HandReach-v0 result in [42] shows that rigid robot hand IK is impossible to train with shaped reward, we still want to try if this is also true in our soft robot case. The shaped reward used is the negative of the average distance error. These modifications are applied for all results later in this work and the results are presented in the next section.

5.4.3 Final Results

Performance Evaluation

The final results presented use the aforementioned methods including curriculum learning and reward shaping. Figure 5.10 shows the learning curves for one-step ($T = 1$) and sequential ($T = 5$) formulation. The yellow line indicates that the current optimal policy meets the tolerance requirement of current task and will switch to a harder task with a smaller tolerance. The tolerance decreases from $1 \rightarrow 0.9 \rightarrow 0.81 \rightarrow 0.729 \rightarrow 0.656 \rightarrow 0.59 \rightarrow 0.531 \rightarrow 0.478 \rightarrow 0.430 \rightarrow 0.387\dots$ (*unit is cm*). After 200 epochs, one-step formulation finishes learning task with tolerance 0.531cm while sequential formulation learns much faster and finishes learning a harder task with tolerance 0.387cm.

In 5.10, we formulate our task as "achieving average distance error smaller than tolerance with 90% success rate", which is not necessary with shaped reward. In other words, the average distance error should decline during training without knowing the task. In addition, the actual average distance error should be lower than the tolerance. Figure 5.11 shows the actual average distance error after finishing each task. The actual average distance error is lower than the tolerance most of the time. This indicates that more aggressive curriculum can be applied to expedite the learning process. For example, one possible curriculum is to set the next tolerance as current average distance error.

5. Learning-based Control on Cable-Driven Foam Manipulator

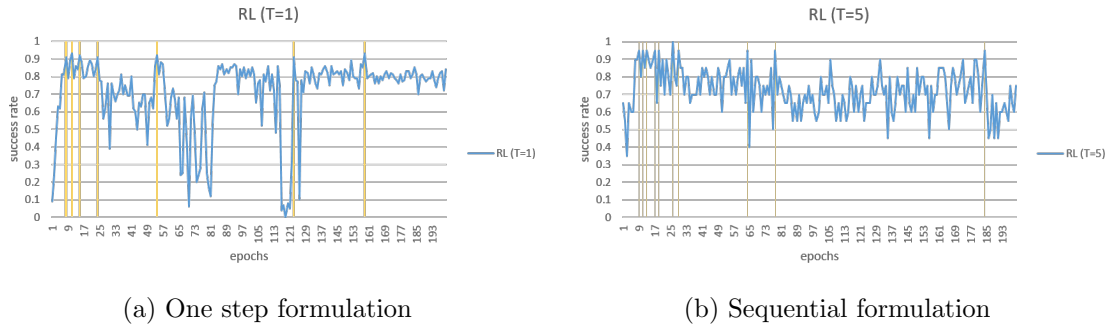


Figure 5.10: Learning curves for (1) one-step formulation (2) sequential formulation. The yellow vertical lines shows the current policy meets the requirement of current task and will switch to a harder task afterward.

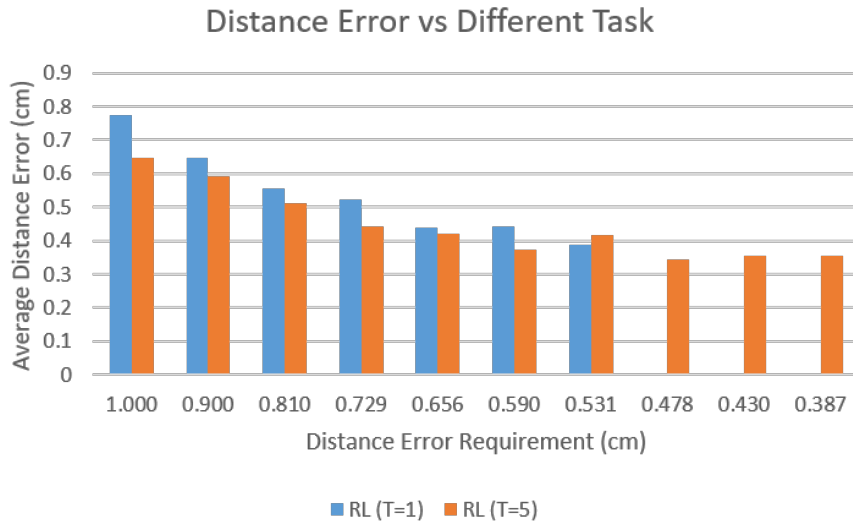
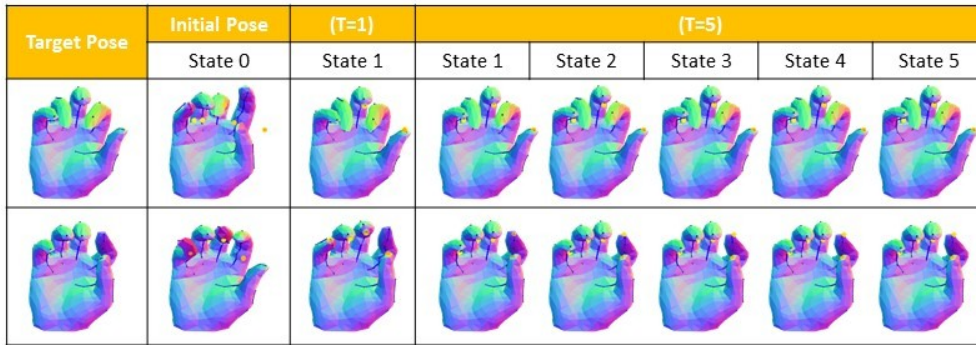


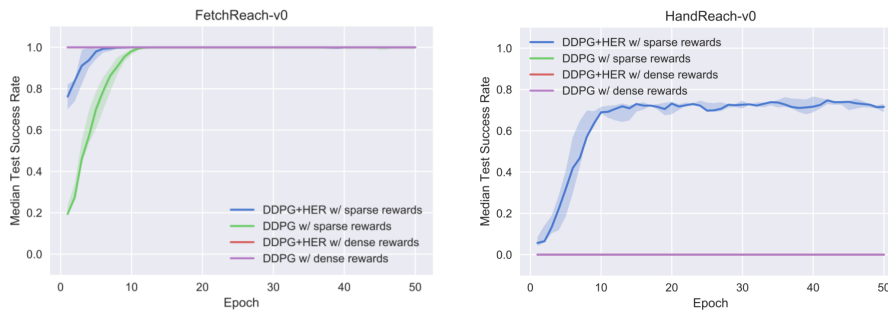
Figure 5.11: Performance evaluation of best policy for different task/distance error requirement. Note that the RL(T=1) only learns up to task ($d=0.531$) after 200 epochs and that the performance of the best policy are 0.332cm for RL(T=1) and 0.328cm for RL(T=5)

Qualitative Performance

Figure 5.12 shows the qualitative results. The target pose and initial pose is generated by random actuation of cables. The policies used are the best policies trained after 200 epochs. As mentioned in 5.4.2, the policy with sequential formulation still learns to achieve the goal directly in one step.

Figure 5.12: IK Visualization for both one-step ($T = 1$) and sequential ($T = 5$)

Comparison with [42]



(a) FetchReach

(b) HandReach

Figure 5.13: Learning curves for FetchReach-v0 and HandReach-v0 copied from [42]

Here our result is compared with the result presented in [42]. The FetchReach-v0 is solving IK on a rigid robot arm, which can be viewed as a one-finger robot in our case. The result figure, Figure 5.13a, is consistent with our result that shaped reward is better in IK task. However, interestingly the HandReach-v0 example shown in Figure 5.13b shows a different result. The HandReach-v0 is solving IK on a rigid anthropomorphic five-fingered hand, which looks similar to our soft robot hand. Here we compare their formulation with ours.

- State: Aside from the finger tip positions used in our work, they also include 24 positions and velocities of the robot joints.
- Action: In our work, the actuation value of the 10 cables are coupled to form a 5

dimension action vector. In their work, they have 20 degrees of freedom(DOFs).

- Number of steps in an episode: They set the maximum number of steps taken in an episode to 200. In our work, we use a smaller, and thus harder value, 5.
- Reward: The definitions of both shaped reward and sparse and binary reward are the same.
- Task: They only train one single task with fixed tolerance 1cm, while we start with tolerance 1cm, but make the task harder after the task is well-learned.

The HandReach-v0 figure shows that using dense reward to reach tolerance 1cm is impossible after epoch. It is true that their task can be harder with 20 DOFs, but for sparse reward the task should also be equivalent hard. In our soft hand IK task, shaped reward learns faster than sparse reward with DDPG + HER.

5.4.4 Comparison with other Learning-based Methods

For real world practice, it is also of interest to compare the sample efficiency between different methods. Here we compare 1)Reinforcement Learning, 2)Supervised Learning, 3)Linear Ridge Regression, and 4)Nearest Neighbor. To compare reinforcement learning method with the same baseline as supervised learning, we use the same network work structure as well as network input. Current states are excluded in the network input. In ($T = 1$) formulation, every epoch has 50 cycles, 10 roll-outs per cycle, and 1 action per episode. One datum is a pair of action taken and goal state. Consequently, a total of $50 \times 10 \times 1 = 500$ data is collected per epoch during training. The collected training data is later used for other methods. A fixed set of test data of size 100 is used for evaluating the performance for all methods. Here the supervised learning method is behavior cloning, which minimizes the mean squared error of the predicted action in training data. Linear ridge regression uses a linear model with additional L2 ridge regularizer. Nearest neighbor method is basically kNN(k nearest neighbor) method with $k = 1$ and directly uses the action of the nearest neighbor.

Figure 5.14 plots the performance (average distance error in centimeter) with respect to number of data. Linear Ridge Regression has the worst performance due to the deficient model complexity. Nearest Neighbor method shows the best performance as well as sample efficiency. This can be reasoned by the nature of instance-based

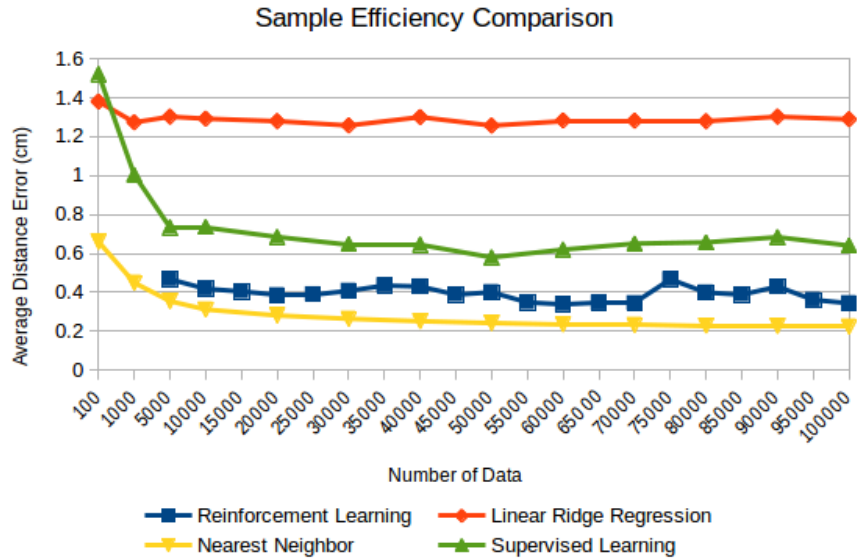


Figure 5.14: Sample efficiency comparison

learning and the large and randomly distributed training data that covers the robot workspace (illustrated in Fig. 5.1). Lastly, reinforcement learning defeats supervised learning. The main difference between the two methods is the essence of loss function. In supervised learning, the network tries to fit the actions in the training data while the reinforcement learning objective function is to maximize rewards that is computed based on the average distance error. This can be fundamentally different and the RL objective function is physically more meaningful. Another possible cause is that the RL algorithm, DDPG, has a more complex actor-critic mechanism which might also help the learning.

5.4.5 Conclusion

In our work, we have applied behavior cloning as well as reinforcement learning to solve soft robot hand IK problem. Behavior cloning is simple to train and the average distance error trained with 10,000 data is 0.624cm. With 100,000 data, the average distance error can be improved slightly to 0.584cm. Indeed, more data can be used to improve performance, but we also show that RL with shaped reward can be more sample efficient. We use DDPG and HER as our core RL algorithm and incorporate

5. Learning-based Control on Cable-Driven Foam Manipulator

curriculum learning to enable learning harder task once the current task is learned. Sequential formulation can also be helpful to preserve successful actions and thus expedite the learning. During testing, the policy trained with sequential formulation also reaches the target state in one step. Last but not least, we show that a shaped reward learns faster than a sparse and binary reward in our IK task, which overthrows the result in [42].

Chapter 6

Conclusions

In this thesis, we presented a set of tools and methodologies related to the broad category of cable-driven soft robots. Fabrication pipelines are developed for two such robots, including plush toy robots and foam robots. Foam robots are more preferred than plush toy robots because of the better material property in recovery motion. Simulations are built to model the motions of cable-driven soft robots and an interactive design system is developed based on the simulation. With this design tool, the user is allowed to explore the massive design space of cable placement and it helps the user generate an optimal and realizable winch-cable network given desired motions. Inverse kinematics problem is particularly investigated for the control of soft robots. Other than an IK solution that is based on the finite element simulation, we also propose two learning-based approaches. One is supervised learning and the other one is reinforcement learning. All these techniques are valid solutions and our result shows smaller average distance error using reinforcement learning than using supervised learning.

6. Conclusions

Bibliography

- [1] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pages 5048–5058, 2017. [5.4.1](#)
- [2] Moritz Bächer, Bernd Bickel, Doug L James, and Hanspeter Pfister. Fabricating articulated characters from skinned meshes. *ACM Trans. Graph.*, 31(4):47–1, 2012. [2.0.1](#)
- [3] Yousef Bahramzadeh and Mohsen Shahinpoor. A review of ionic polymeric soft actuators and sensors. *Soft Robotics*, 1(1):38–52, 2014. [4.6.1](#)
- [4] Zaphod Beetlebrox. How to create a pneumatic system for (f.i.r.s.t) robots. URL <http://www.instructables.com/id/How-to-Create-a-Pneumatic-System-for-FIRST-Robots/>. [4.6.1](#)
- [5] James M Bern, Kai-Hung Chang, and Stelian Coros. Interactive design of animated plushies. *ACM Transactions on Graphics (TOG)*, 36(4):80, 2017. [1.3](#)
- [6] Bernd Bickel, Peter Kaufmann, Mélina Skouras, Bernhard Thomaszewski, Derek Bradley, Thabo Beeler, Phil Jackson, Steve Marschner, Wojciech Matusik, and Markus Gross. Physical face cloning. *ACM Transactions on Graphics (TOG)*, 31(4):118, 2012. [2.0.1](#)
- [7] Eric Brown, Nicholas Rodenberg, John Amend, Annan Mozeika, Erik Steltz, Mitchell R Zakin, Hod Lipson, and Heinrich M Jaeger. Universal robotic gripper based on the jamming of granular material. *Proceedings of the National Academy of Sciences*, 107(44):18809–18814, 2010. [4.6.2](#)
- [8] Jacques Calì, Dan A Calian, Cristina Amati, Rebecca Kleinberger, Anthony Steed, Jan Kautz, and Tim Weyrich. 3d-printing of non-assembly, articulated models. *ACM Transactions on Graphics (TOG)*, 31(6):130, 2012. [2.0.1](#)
- [9] Marcello Calisti, Michele Giorelli, Guy Levy, Barbara Mazzolai, B Hochner, Cecilia Laschi, and Paolo Dario. An octopus-bioinspired solution to movement and manipulation for soft robots. *Bioinspiration & biomimetics*, 6(3):036002, 2011. [1.2](#), [4.6.1](#), [4.6.2](#), [4.6.3](#)

- [10] Duygu Ceylan, Wilmot Li, Niloy J Mitra, Maneesh Agrawala, and Mark Pauly. Designing and fabricating mechanical automata from mocap sequences. *ACM Transactions on Graphics (TOG)*, 32(6):186, 2013. [2.0.1](#)
- [11] FJ Chen, Steven Dirven, WL Xu, and XN Li. Soft actuator mimicking human esophageal peristalsis for a swallowing robot. *IEEE/ASME Transactions on Mechatronics*, 19(4):1300–1308, 2014. [2.0.3](#)
- [12] Weikai Chen, Xiaolong Zhang, Shiqing Xin, Yang Xia, Sylvain Lefebvre, and Wenping Wang. Synthesis of filigrees for digital fabrication. *ACM Transactions on Graphics (TOG)*, 35(4):98, 2016. [2.0.1](#)
- [13] Nadia G Cheng, Maxim B Lobovsky, Steven J Keating, Adam M Setapen, Katy I Gero, Anette E Hosoi, and Karl D Iagnemma. Design and analysis of a robust, low-cost, highly articulated manipulator enabled by jamming of granular media. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4328–4333. IEEE, 2012. [4.6.1](#), [4.6.2](#), [4.11](#)
- [14] Ching-Ping Chou and Blake Hannaford. Measurement and modeling of mckibben pneumatic artificial muscles. *IEEE Transactions on robotics and automation*, 12(1):90–102, 1996. [4.6.1](#)
- [15] Stelian Coros, Bernhard Thomaszewski, Gioacchino Noris, Shinjiro Sueda, Moira Forberg, Robert W Sumner, Wojciech Matusik, and Bernd Bickel. Computational design of mechanical characters. *ACM Transactions on Graphics (TOG)*, 32(4):83, 2013. [2.0.1](#)
- [16] Tao Du, Adriana Schulz, Bo Zhu, Bernd Bickel, and Wojciech Matusik. Computational multicopter design. 2016. [2.0.1](#)
- [17] Christian Duriez. Control of elastic soft robots based on real-time finite element method. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3982–3987. IEEE, 2013. [2.0.3](#)
- [18] Akash Garg, Andrew O Sageman-Furnas, Bailin Deng, Yonghao Yue, Eitan Grinspun, Mark Pauly, and Max Wardetzky. Wire mesh design. *ACM Trans. Graph.*, 33(4):66–1, 2014. [2.0.1](#)
- [19] Damien Gauge, Stelian Coros, Sandro Mani, and Bernhard Thomaszewski. Interactive design of modular tensegrity characters. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 131–138. Eurographics Association, 2014. [2.0.1](#)
- [20] Michele Giorelli, Federico Renda, Gabriele Ferri, and Cecilia Laschi. A feed-forward neural network learning the inverse kinetics of a soft cable-driven manipulator moving in three-dimensional space. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 5033–5039. IEEE, 2013. [5.1](#), [5.3.2](#), [3](#), [4](#), [5.3.2](#), [??](#), [5.2](#)

- [21] Michele Giorelli, Federico Renda, Marcello Calisti, Andrea Arienti, Gabriele Ferri, and Cecilia Laschi. Neural network and jacobian method for solving the inverse statics of a cable-driven soft arm with nonconstant curvature. *IEEE Transactions on Robotics*, 31(4):823–834, 2015. [2.0.3](#), [5.1](#), [5.3.2](#), [3](#), [4](#), [5.3.2](#), [??](#), [5.2](#), [5.3.3](#)
- [22] Shuxiang Guo, Toshio Fukuda, and Kinji Asaka. A new type of fish-like underwater microrobot. *Ieee/Asme Transactions on Mechatronics*, 8(1):136–141, 2003. [4.6.1](#)
- [23] Yuki Igarashi and Takeo Igarashi. Pillow: Interactive flattening of a 3d model for plush toy design. In *International Symposium on Smart Graphics*, pages 1–7. Springer, 2008. [2.0.1](#)
- [24] Martin Kilian, Simon Flöry, Zhonggui Chen, Niloy J. Mitra, Alla Sheffer, and Helmut Pottmann. Curved folding. In *ACM SIGGRAPH 2008 Papers*, SIGGRAPH '08, pages 75:1–75:9, New York, NY, USA, 2008. ACM. ISBN 978-1-4503-0112-1. doi: 10.1145/1399504.1360674. URL <http://doi.acm.org/10.1145/1399504.1360674>. [2.0.1](#)
- [25] Kwang J Kim and Satoshi Tadokoro. Electroactive polymers for robotic applications. *Artificial Muscles and Sensors (291 p.)*, Springer: London, United Kingdom, 23, 2007. [4.6.1](#)
- [26] Mina Konaković, Keenan Crane, Bailin Deng, Sofien Bouaziz, Daniel Piker, and Mark Pauly. Beyond developable: computational design and fabrication with auxetic materials. *ACM Transactions on Graphics (TOG)*, 35(4):89, 2016. [2.0.1](#)
- [27] Cecilia Laschi, Matteo Cianchetti, Barbara Mazzolai, Laura Margheri, Maurizio Follador, and Paolo Dario. Soft robot arm inspired by the octopus. *Advanced Robotics*, 26(7):709–727, 2012. [1.2](#), [4.6.1](#), [4.6.2](#), [4.6.3](#)
- [28] Chiwon Lee, Myungjoon Kim, Yoon Jae Kim, Nhayoung Hong, Seungwan Ryu, H Jin Kim, and Sungwan Kim. Soft robot review. *International Journal of Control, Automation and Systems*, 15(1):3–15, 2017. [4.6.1](#)
- [29] Xian-Ying Li, Chao-Hui Shen, Shi-Sheng Huang, Tao Ju, and Shi-Min Hu. Popup: Automatic paper architectures from 3d models. *ACM Trans. Graph.*, 29(4):111:1–111:9, July 2010. ISSN 0730-0301. doi: 10.1145/1778765.1778848. URL <http://doi.acm.org/10.1145/1778765.1778848>. [2.0.1](#)
- [30] Xian-Ying Li, Tao Ju, Yan Gu, and Shi-Min Hu. A geometric study of v-style pop-ups: Theories and algorithms. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 98:1–98:10, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0943-1. doi: 10.1145/1964921.1964993. URL <http://doi.acm.org/10.1145/1964921.1964993>. [2.0.1](#)
- [31] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom

- Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015. [5.4.1](#)
- [32] Huai-Ti Lin, Gary G Leisk, and Barry Trimmer. Goqbot: a caterpillar-inspired soft-bodied rolling robot. *Bioinspiration & biomimetics*, 6(2):026007, 2011. [1.2](#), [4.6.1](#), [4.6.2](#), [4.6.3](#)
- [33] Andrew D Marchese and Daniela Rus. Design, kinematics, and control of a soft spatial fluidic elastomer manipulator. *The International Journal of Robotics Research*, 35(7):840–869, 2016. [2.0.3](#)
- [34] Andrew D Marchese, Konrad Komorowski, Cagdas D Onal, and Daniela Rus. Design and control of a soft and continuously deformable 2d robotic manipulation system. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2189–2196. IEEE, 2014. [2.0.3](#)
- [35] James McCann, Lea Albaugh, Vidya Narayanan, April Grow, Wojciech Matusik, Jennifer Mankoff, and Jessica Hodgins. A compiler for 3d machine knitting. *ACM Transactions on Graphics (TOG)*, 35(4):49, 2016. [4.2](#)
- [36] Vittorio Megaro, Bernhard Thomaszewski, Maurizio Nitti, Otmar Hilliges, Markus Gross, and Stelian Coros. Interactive design of 3d-printable robotic creatures. *ACM Transactions on Graphics (TOG)*, 34(6):216, 2015. [2.0.1](#)
- [37] Jun Mitani and Hiromasa Suzuki. Making papercraft toys from meshes using strip-based approximate unfolding. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 259–263, New York, NY, USA, 2004. ACM. doi: 10.1145/1186562.1015711. URL <http://doi.acm.org/10.1145/1186562.1015711>. [2.0.1](#)
- [38] Mehran Mojarad and Mohsen Shahinpoor. Biomimetic robotic propulsion using polymeric artificial muscles. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 3, pages 2152–2157. IEEE, 1997. [4.6.1](#)
- [39] Yuki Mori and Takeo Igarashi. Plushie: an interactive design system for plush toys. In *ACM Transactions on Graphics (TOG)*, volume 26, page 45. ACM, 2007. [2.0.1](#)
- [40] Cagdas D Onal, Xin Chen, George M Whitesides, and Daniela Rus. Soft mobile robots with on-board chemical pressure generation. In *Robotics Research*, pages 525–540. Springer, 2017. [4.6.1](#)
- [41] Jesús Pérez, Bernhard Thomaszewski, Stelian Coros, Bernd Bickel, José A Canabal, Robert Sumner, and Miguel A Otaduy. Design and fabrication of flexible rod meshes. *ACM Transactions on Graphics (TOG)*, 34(4):138, 2015. [2.0.1](#)
- [42] Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker,

- Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chocie, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018. [5.1](#), [5.4.1](#), [5.4.2](#), [5.4.2](#), [5.4.3](#), [5.13](#), [5.4.3](#), [5.4.5](#)
- [43] KC Rusch. Load–compression behavior of flexible foams. *Journal of Applied Polymer Science*, 13(11):2297–2311, 1969. [4.6.2](#)
- [44] Siddharth Sanan. *Soft inflatable robots for safe physical human interaction*. PhD thesis, Carnegie Mellon University, 2013. [4.6.2](#)
- [45] Frank Saunders, Barry A Trimmer, and Jason Rife. Modeling locomotion of a soft-bodied arthropod using inverse dynamics. *Bioinspiration & biomimetics*, 6(1):016001, 2010. [2.0.3](#)
- [46] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International Conference on Machine Learning*, pages 1312–1320, 2015. [5.4.1](#)
- [47] L McDonald Schetky. Shape-memory alloys. *Kirk-Othmer Encyclopedia of Chemical Technology*, 1982. [4.6.1](#)
- [48] Sangok Seok, Cagdas Denizel Onal, Kyu-Jin Cho, Robert J Wood, Daniela Rus, and Sangbae Kim. Meshworm: a peristaltic soft robot with antagonistic nickel titanium coil actuators. *IEEE/ASME Transactions on mechatronics*, 18(5):1485–1497, 2013. [4.6.1](#)
- [49] Robert F Shepherd, Filip Ilievski, Wonjae Choi, Stephen A Morin, Adam A Stokes, Aaron D Mazzeo, Xin Chen, Michael Wang, and George M Whitesides. Multigait soft robot. *Proceedings of the national academy of sciences*, 108(51):20400–20403, 2011. [1.2](#), [4.6.2](#), [4.6.3](#)
- [50] Mélina Skouras, Bernhard Thomaszewski, Bernd Bickel, and Markus Gross. Computational design of rubber balloons. In *Computer Graphics Forum*, volume 31, pages 835–844. Wiley Online Library, 2012. [2.0.1](#)
- [51] Mélina Skouras, Bernhard Thomaszewski, Stelian Coros, Bernd Bickel, and Markus Gross. Computational design of actuated deformable characters. *ACM Transactions on Graphics (TOG)*, 32(4):82, 2013. [2.0.1](#)
- [52] Mélina Skouras, Bernhard Thomaszewski, Peter Kaufmann, Akash Garg, Bernd Bickel, Eitan Grinspun, and Markus Gross. Designing inflatable structures. *ACM Trans. Graph.*, 33(4):63:1–63:10, July 2014. ISSN 0730-0301. doi: 10.1145/2601097.2601166. URL <http://doi.acm.org/10.1145/2601097.2601166>. [2.0.1](#)
- [53] Mélina Skouras, Stelian Coros, Eitan Grinspun, and Bernhard Thomaszewski. Interactive surface design with interlocking elements. *ACM Transactions on*

- Graphics (TOG)*, 34(6):224, 2015. [2.0.1](#)
- [54] Peng Song, Chi-Wing Fu, Prashant Goswami, Jianmin Zheng, Niloy J Mitra, and Daniel Cohen-Or. Reciprocal frame structures made easy. *ACM Transactions on Graphics (TOG)*, 32(4):94, 2013. [2.0.1](#)
- [55] Bernhard Thomaszewski, Stelian Coros, Damien Gauge, Vittorio Megaro, Eitan Grinspun, and Markus Gross. Computational design of linkage-based characters. *ACM Transactions on Graphics (TOG)*, 33(4):64, 2014. [2.0.1](#)
- [56] Michael T Tolley, Robert F Shepherd, Michael Karpelson, Nicholas W Bartlett, Kevin C Galloway, Michael Wehner, Rui Nunes, George M Whitesides, and Robert J Wood. An untethered jumping soft robot. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 561–566. IEEE, 2014. [4.6.1](#)
- [57] Michael T Tolley, Robert F Shepherd, Bobak Mosadegh, Kevin C Galloway, Michael Wehner, Michael Karpelson, Robert J Wood, and George M Whitesides. A resilient, untethered soft robot. *Soft robotics*, 1(3):213–223, 2014. [1.2](#), [4.6.2](#), [4.6.3](#)
- [58] Takuya Umedachi, Vishesh Vikas, and Barry A Trimmer. Highly deformable 3-d printed soft robot generating inching and crawling locomotions with variable friction legs. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 4590–4595. IEEE, 2013. [4.6.3](#)
- [59] Etienne Vouga, Mathias Höbinger, Johannes Wallner, and Helmut Pottmann. Design of self-supporting surfaces. *ACM Transactions on Graphics (TOG)*, 31(4):87, 2012. [2.0.1](#)
- [60] Younan Xia and George M Whitesides. Soft lithography. *Annual review of materials science*, 28(1):153–184, 1998. [4.6.3](#)
- [61] Yohei Yamashita, Tatsuya Ishikawa, Hironori Mitake, Yutaka Takase, Fumihiro Kato, Ikumi Susa, Shoichi Hasegawa, and Makoto Sato. Stuffed toys alive!: cuddly robots from fantasy world. In *ACM SIGGRAPH 2012 Posters*, page 80. ACM, 2012. [3.2](#), [4.6.1](#), [4.6.2](#)
- [62] Jonas Zehnder, Stelian Coros, and Bernhard Thomaszewski. Designing structurally-sound ornamental curve networks. *ACM Transactions on Graphics (TOG)*, 35(4):99, 2016. [2.0.1](#)
- [63] Lifeng Zhu, Weiwei Xu, John Snyder, Yang Liu, Guoping Wang, and Baining Guo. Motion-guided mechanical toy modeling. *ACM Trans. Graph.*, 31(6):127–1, 2012. [2.0.1](#)