Computational Mechanical Design Method for Achieving Both Aesthetics and Functionality

審美性と機能性を両立させる計算的機構設計手法の提案

February 2021

Takuto TAKAHASHI
髙橋　卓人

Computational Mechanical Design Method for Achieving Both Aesthetics and Functionality

審美性と機能性を両立させる計算的機構設計手法の提案

February 2021

Waseda University

Graduate School of Creative Science and Engineering

Department of Modern Mechanical Engineering,

Research on Computational Mechanism Design

Takuto TAKAHASHI

髙橋　卓人

Abstract

It is important to consider not only functionality but also aesthetics when designing a robot that is meant to imitate the way humans and animals are. However, this design process is usually difficult even for experienced designers. In addition, this requires a lot of trial and error, which is costly and time-consuming.

If we formulate the design process to achieve both functionality and aesthetics as an optimization process, it can be expressed in the following three stages. First, the parameters related to the design are set. Next, the objective function to evaluate the functionality is set. Finally, if there is an objective function to evaluate the aesthetics, the minimum value of the objective function of the aesthetics is searched under the constraint that the objective function to evaluate the functionality is zero.

In each of the above three stages, there are problems that create design difficulties and time costs. Depending on the parameter settings, it is not possible to guarantee the solution, fine tuning is necessary after manufacturing, and trial-and-error is difficult to be carried out while satisfying the constraint conditions.

Therefore, three concepts are proposed in this paper to design a mechanism to achieve both aesthetics and functionality. The first concept is “Constructability”, which enables us to reconfigure parameters without technical knowledge. The second concept is “Explorability,” which makes it possible to interactively search for a solution to the aesthetic objective function while satisfying the constraint that the objective function of functionality is zero.The third concept is “Programmability”, which enables us to adjust functions by simply replacing some parts after manufacture.

Furthermore, an implementation method using the optimization calculation of each concept has been devised for basic mechanisms such as a counterweight mechanism, a cam mechanism and a spring-loaded self-weight compensation mechanism.

Dedication

The continued support of my mother and father helped me to carry out this research to the end. Without their encouragement and financial support, I would not have been able to continue my research. I would like to live my life so that this experience will not be in vain, so that I can give something back to my mother and father in the future.

Declaration

I declare that this paper has been edited by me, the research has been carried out by me and has not been submitted to any other academic organization, with the exception of previous papers done by reference, acknowledgements or self.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Prof. Shigeki Sugano, for providing me with a research environment and machines. I would like to thank him for accepting me into his laboratory and giving me the best environment for my research. I would like to thank Prof. Okuno for his detailed guidance, how to write a real paper, and for pushing me to start research in this new field. I would also like to thank Prof. Tetsuya Ogata for giving me great guidance in the research environment. Thanks to my professors, I have been able to continue my research on an international scale through internships abroad (at the University of Montreal), presentations at IROS, submissions to journals (RA-L), etc. I would like to express my gratitude to them.

I would like to thank Prof. Bernhard Tomaszewski for accepting me as an intern and for his guidance afterwards; he was especially good at teaching me the technical aspects and always taught me about the new world. When I read Prof. Bernhard Tomaszewski's and Prof. Stelian Coros' paper "Computational Design of Mechanical Characters" when I was an undergraduate student, I could not imagine the current situation at all. It has been an invaluable experience and is truly a dream come true.

I believe that research should be an act of sublimation, not a small community of complacency, but an act of working internationally, checking one's standing, and receiving points for peer review, etc. However, the presence of the right advisors and environment is essential for this to happen. I am truly grateful for this fortunate opportunity.

Contents

Chapter 1

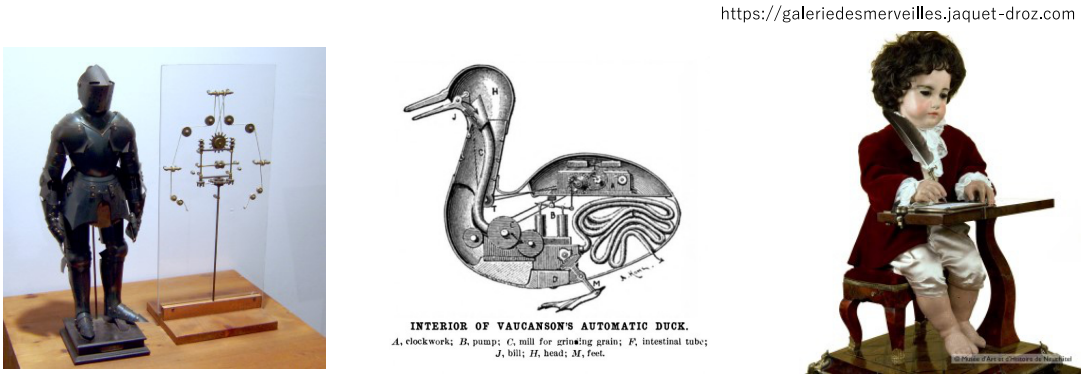
Introduction

1.1 Motivation

The word "Robot" was coined in Karel Čapek's play "R.U.R.", published in 1920, as a term for an artificial person performing labor (robota in Czech). Later, the word robotics was used by science fiction novelist Isaac Asimov for his own science fiction novels with robots (e.g., I, Robot, published in 1950), combining robot with the word -ics for learning. Although it is still difficult to define the term robot, it is often used in two main senses: first, a machine that performs tasks somewhat autonomously to contribute to the secondary industry, so-called industrial robots, and second, a machine that operates somewhat autonomously, imitating the form of a living organism. In particular, the second, a robot as a machine that imitates the form of a living organism, is designed so that the mechanism can not only function with limited energy, but at the same time looks acceptable to people.

The act of designing a mechanism that looks and functions in a way that is acceptable to people has a history older than the term "Robot" was first used and has fascinated people. For example, Leonardo da Vinci designed a machine called the "Automaton knight" around 1495 that acted like a human being; in 1739, inventor Jacques de Vaucanson designed a duck that consisted of about 400 moving parts and behaved like

a living thing. He presented the "Canard digérateur." The watchmaker Pierre Jaquet-Droz produced "The writer," a 40-character programmable writing automaton consisting of some 6,000 parts in 1772.



Leonardo's mechanical knight (1495) Digesting Duck (1739) The writer (1768-1774)

(a) (b) (c)

Figure 1.1: The history of automata. (a) is an automata said to have been designed by Leonardo da Vinci. It is said to have been able to move in imitation of a knight. (b) is an imaginary drawing of a duck automata by Jacques de Vaucanson. This automata is said to have performed a series of movements from eating to excretion. (c) is a hand-drawn automata "The writer" by Pierre Jaquet-Droz. This automata can be programmed with text by changing the shape of the cams for each block.

With the development of electronics since the mid-20th century, the act of designing a functioning mechanism while maintaining the aesthetic appeal of the above has led to the use of electronics technology, and in 1949 Walt Disney's introduction of a singing, moving bird automaton led to the creation of The Walt With the help of Roger E. Broggie and Ken Anderson of the Disney Company, he developed "dancing man," a machine that could replicate the tap dancing of Buddy Ebsen (see Fig. 1.2). In 1969, Tetsuro Mori of Yaskawa Electric Co. The term "Mechatronics," a combination of the words "Mechanism" and "Electronics," was applied for as a trademark. Furthermore, in 1973, Ichiro Kato and his colleagues at Waseda University developed the world's first full-fledged intelligent humanoid robot, WABOT, which was capable of bipedal locomotion and grasping and moving objects with its tactile two hands (see Fig. 1.3).

(source Wikimedia Commons)
https://thewonderofminiatures.home.blog/2017/07/11/disney-world-miniatures/
https://allears.net/2020/03/30/taking-a-look-back-at-the-history-of-animatronics-in-the-disney-parks/

(a) (b) (c)

Figure 1.2: History of Walt Disney's audio-animatronics. (a) is a prototype of the audio animatronics "Dancing Man" or "Project Little Man." The animatronics are the size of a 9-inch doll and mimic the movements of actor Buddy Ebsen. (b) is a full-size audio-animatronic of President Lincoln. The audio-animatronics became popular when they were displayed at the New York World's Fair. (c) An interior view of President Lincoln's audio animatronics. The machines are cleverly designed to function properly while maintaining a high level of aesthetics.

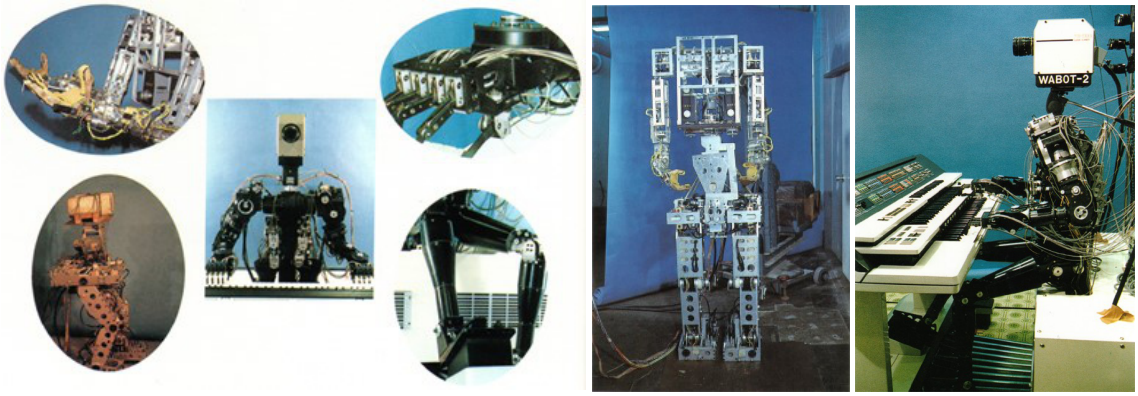


Figure 1.3: History of humanoid robots at Waseda University. The robot, called WABOT, was not only shaped like a human, but also equipped with the necessary intelligence to perform physical tasks, and it became the world's first full-fledged humanoid robot.

Behind these creative activities, experienced designers often achieve these goals through mindless trial and error. In other words, the designer incorporates the mechanism into the blueprint, manufactures the actual product based on the blueprint, and then verifies whether it meets the target requirements, and if it does not meet the target, the designer goes back to the point of redesigning the blueprint. Even if the mechanism works well, if it is determined that it does not look good, the designer has to redesign the design as well. Not only is it difficult for a novice designer, but it is also a time-consuming and costly process for even an experienced designer (see Fig. 1.4).

In order to solve such time-consuming design problems, there is a new field of Computational Design, which makes use of the computational power of computers to significantly reduce the time required for design. Specifically, the results of manufacturing are simulated and evaluated in the virtual space of the computer before actual manufacturing, and the design is automatically remade until the results meet the requirements (see Fig. 1.5). The computer's ability to rapidly find the best parameters based on constraints greatly reduces designer error and increases the probability of success for the final evaluation at the time of manufacture. This field is not only used in robotics manufacturing, but also in all areas of design.

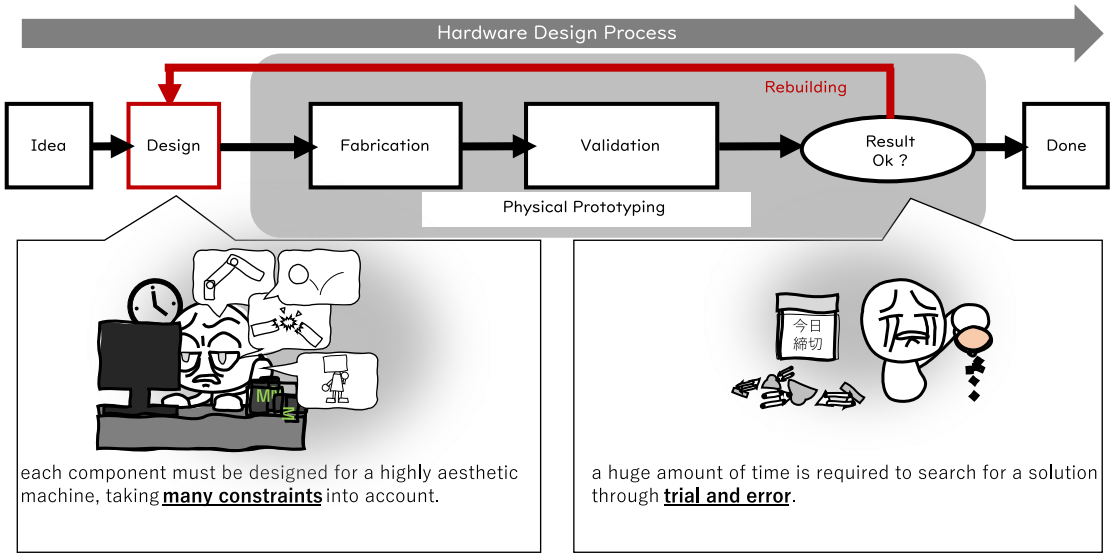


Figure 1.4: The process in machine design and the problems associated with it. The designer decides on the geometry of the machine part by part, in order to achieve an aesthetically pleasing overall structure, while dealing with too many constraints for a human to consider. In addition, the actual manufacturing process requires a lot of trial and error, which is time consuming and costly.

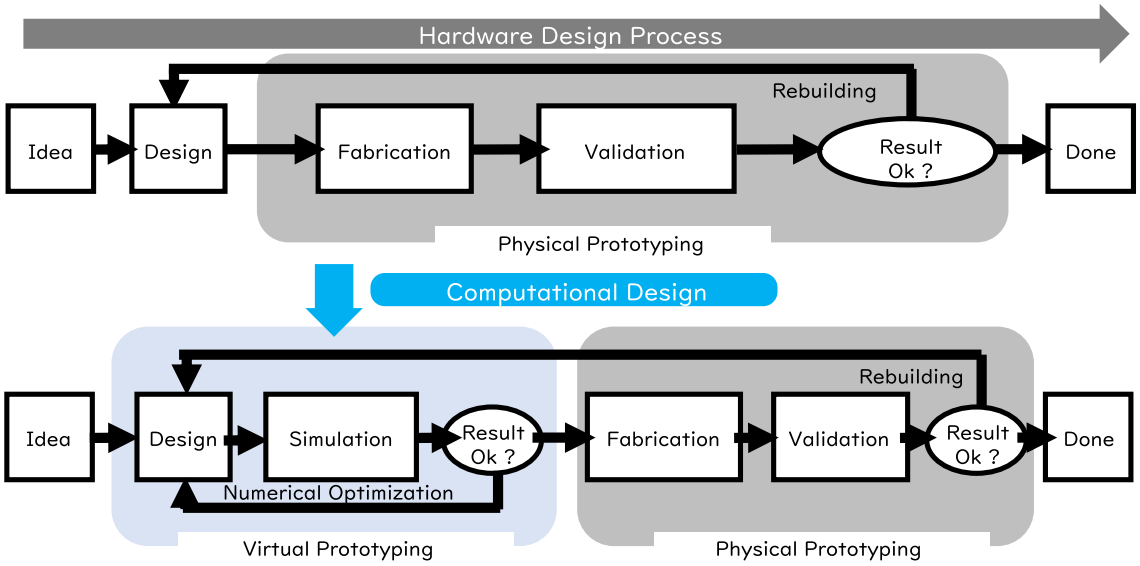


Figure 1.5: The general design flow in Computational Design. Machine manufacturing is typically a time-consuming process, and time wasted due to trial and error can be critical in projects with limited schedules and budgets. Computational Design pre-simulates and evaluates prototypes in virtual space prior to manufacturing and automatically modifies the design until the proper evaluation is achieved. This saves time during the entire design process.

1.2 Problem Formulation

In order to utilize computational design in the field of robotics, it is essential to put the design process of the mechanism into the context of numerical optimization to ensure that it is aesthetically pleasing and functionally correct. If the problem of design is reduced to a problem of determining numerical parameters, the design which has been dependent on implicit and empirical aspects can be made explicit, and the numerical calculation using computers can support the discovery of the optimal values. The method we have focused on in this study is a design support method by numerical optimization. From here, we will explain the process of designing a mechanism while maintaining its aesthetics, following the context of design support by numerical optimization.

1.2.1 Design Parameters

First, in order to treat the design problem as a numerical optimization, the parameters involved in that design must be properly set by the designer. There is an n-order parameter space required to determine the design, and let the vector $\mathbf{p}$ be the parameter.

$$\mathbf{p} = (p_1, p_2, \cdots, p_n)^T \in \mathbb{R}^n \tag{1.1}$$

This parameter has to be set appropriately so that it can be used as a variable in the objective function described below.

1.2.2 Objective Functions

In order to determine the parameters described above, it is necessary to set up an objective function. In this task, there are two kinds of objective functions: one to evaluate whether the mechanism works well and the other to evaluate the aesthetics. We will now explain the two types of objective functions.

First, let's discuss the objective function for evaluating the performance of the

mechanism, which is necessary for the mechanism to operate properly. There is a objective function

$$f : \mathbb{R}^n \to \mathbb{R} \tag{1.2}$$

such that $f(\mathbf{p}) = 0$ when the mechanism is energy efficient and operates normally. We call it mechanical objective function $f(\cdot)$. Since this objective function can often be set explicitly by appropriate modeling of the mechanism, there is already a large amount of research supporting it using continuous optimization techniques.

Next, let's discuss the objective function for evaluating aesthetics. The function to evaluate aesthetics is expressed as a function

$$g : \mathbb{R}^n \to \mathbb{R} \tag{1.3}$$

that indicates the acceptability of the mechanism to people, that is, aesthetics [1]. We call it aesthetic objective function $g(\cdot)$. This objective function for evaluating aesthetics depends on the subjectivity of the designer and is more difficult to set explicitly than the objective function for the mechanism.

In this case, the designer needs to determine the parameter $\mathbf{p}$, where aesthetic objective function $g(\cdot)$ is the smallest in that mechanical objective function $f(\mathbf{p})$ is 0 (see Fig. 1.6). Through these operations, the design of a mechanism that is both functional and aesthetic is achieved.

$$\min_{\mathbf{p}} \quad g(\mathbf{p}) \quad \text{s.t.} \quad f(\mathbf{p}) = 0, \tag{1.4}$$

1.3 Challenges

However, it is time consuming for an experienced designer to determine the parameter $\mathbf{p}$ related to the structure of the mechanism and its shape with the appearance (aesthetic objective function $g(\cdot)$) and function (mechanism objective function $f(\cdot)$) as design requirements (see Fig. 1.18). There are three reasons for this.

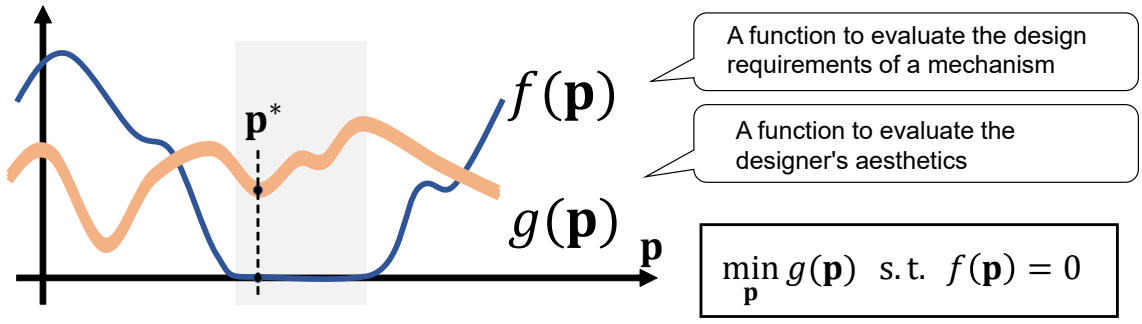


Figure 1.6: Conceptual diagram of the design of a machine to improve aesthetics while maintaining functionality. The process of searching for the optimal solution of the objective function for aesthetics in a space where the objective function for functionality is constrained to be optimal, i.e., zero.

The first challenge is that it is not possible to guarantee that a suitable solution that satisfies the equation can be found for the set design parameters. If the wrong design parameters are found to have been set, the design parameters need to be re-set, which requires a lot of skill and experience, and therefore costs the designer a lot of time and money.

The second challenge is that the setting of the mechanism objective function does not always match the actual performance of the manufactured mechanism and requires fine tuning after manufacture. Repeated trial and error in the manufacturing process itself requires a lot of time.

The third challenge is that the aesthetic function is difficult to set explicitly and must be interactively tried and tested by the designer, and the constraint that the mechanism objective function is zero must be met. Normally, the constraint that the mechanism objective function is zero is not met when the designer freely specifies the shape of the system.

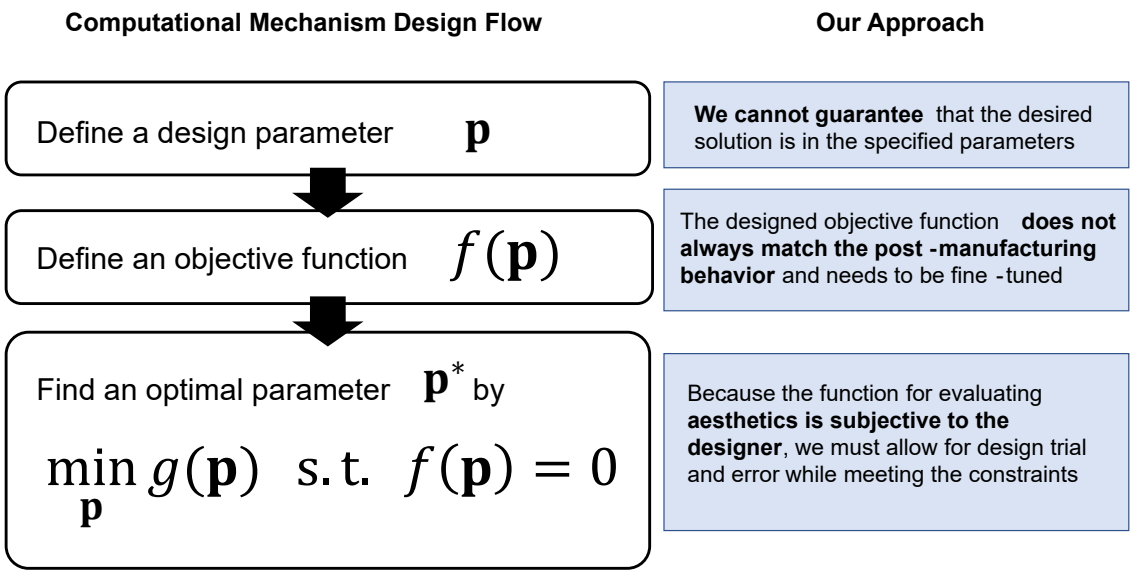


Figure 1.7: In computational mechanism design, the three main problems that prevent you from achieving your design.

1.4 Our Approach

In order to effectively solve the three main problems listed above, this paper proposes three new approaches: “Programmability,” “Explorability” and “Constructability.”

1. *Constructability :* A concept that allows a design novice to reconfigure design parameters without expertise. They do not need to be set up in advance and can specify the overall picture of the mechanism using only touch controls or pointing devices such as a mouse, and then design using optimization. This thesis describes a method for appropriately setting the parameters necessary for optimization by providing the user with an interface to specify the geometry and decompose the joints, using a mechanism called a counterweight to maintain the postural balance of the mechanism as the subject.

2. *Explorability :* A concept that allows the user to interactively change parameters while still satisfying the constraint that the objective function of the mechanism is zero. This allows the user to design a mechanism that is both aesthetically pleasing and functional. This thesis describes a design method for an interface that allows

the user to easily search for aesthetically appropriate parameters while holding on to the objective function of 0, which has been difficult to achieve with a special spring-assisted self-weight guarantee mechanism.

3. *Programmability :* A concept that allows the performance of a mechanism to be changed by simply manufacturing or replacing some special parts after manufacture. The parts in question are called programs, and the designer has the flexibility to change the programs. In this thesis, we focus on a special mechanism called a cam mechanism, which allows for programmability by changing the design of some of its parts, and describe a method for creating a mechanism that allows the function of the machine to be adjusted after it has been manufactured.

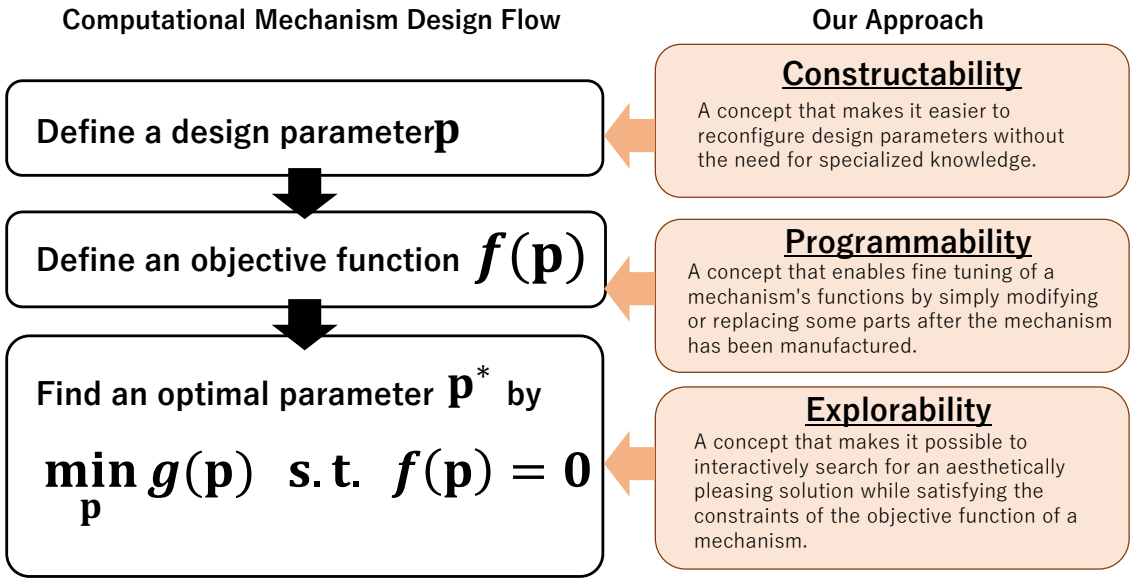


Figure 1.8: Our approach

In order to establish a new mechanism design method using these three concepts, we have developed a new method by treating the basic components of the mechanism, i.e., a cam mechanism, a spring mechanism and a counterweight mechanism.

1.5 Organization of the Thesis

The organization of this PhD thesis is shown in Fig. 1.9. The introductory chapter describes the background knowledge for designing aesthetically appropriate mechanisms, and then introduces the software and research related to them.

In Chapter 2, on the subject of counterweight mechanisms and stability, a method is proposed to design a counterweight mechanism in which the position of the joint is stabilized and returned to the user's sketching input only. This allows the designer to set design parameters interactively without the need for expert knowledge, and approaches the problem of re-setting parameters, which has been difficult.

In Chapter 3, an exploratory design method for a self-weight assurance mechanism is proposed, with the theme of spring mechanisms and balance. Null-space exploration" here allows the designer to interactively search for aesthetically pleasing parameters while still meeting the constraints of the mechanism's objective function. This approach addresses the problem of balancing aesthetics and functionality, which has been difficult to achieve in the past.

In Chapter 4, we focus on cam mechanisms and trajectories, and proposes a framework in which the motion of the mechanism can be programmed by designing the cam geometry. These methods establish the basic concept of programmability in mechanism design and enable us to approach issues that require fine tuning after manufacture.

Finally, the concluding chapter discusses the validity of the methods presented so far and explains the direction of future research.

1.6 Related Work

In particular, there are studies that combine additive manufacturing of mechanisms: Lifeng Zhu et al. have developed a technique to automatically manufacture automata from a specified motion[2]; Vittorio Megaro et al. [3] and Gen Nishida et al. [4] have developed a

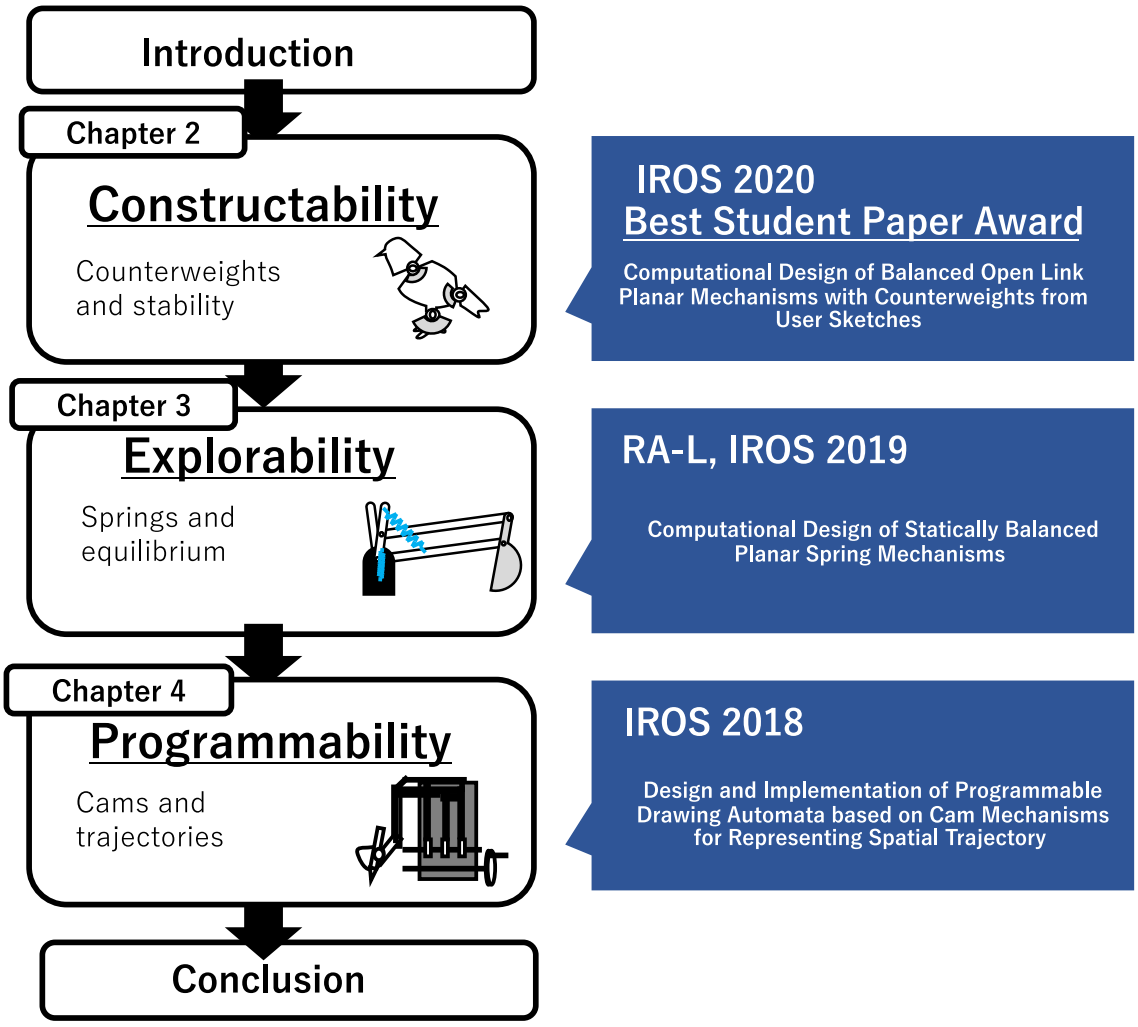


Figure 1.9: Organization of this doctoral thesis. The proposed new approach is explained in each chapter with the basic mechanism as the subject, and how to implement a specific design support system. The results of manufacturing using the system are also presented.

technique to manufacture an arbitrary two-dimensional. Stelian Coros et al. and Bernhard Thomaszewski et al. have developed design tools and optimization techniques for character automata[5, 6], which specify the start and end states of a link to generate a link mechanism to achieve this. However, most of them have focused on the mechanistic aspects, and few of them effectively solve the energy efficiency as well as the appearance problem.

1.6.1 Additive Manufacturing and Fabrication Oriented Design

The widespread use of additive manufacturing technologies, such as 3D printers, has brought about a major change in the act of designing real-world objects [7]. In addition, because complex three-dimensional mesh information can be directly materialized, it is compatible with the materialization of optimization results in the CG field of research. For this reason, research has been conducted on how to materialize three-dimensional characters that could only exist in a virtual space. For example, in the game ”Spore” developed by Maxis [8], the game provides an interface that allows the player to easily design a complex character, while at the same time ensuring that the character is animated without fail. There are studies that realize this as This research uses optimization techniques to estimate from mesh information with appropriate joints [9] shown in Fig. 1.10. There is also research on outputting characters without disassembling the joints in a 3D printer by appropriately combining soft and hard materials [10] shown in Fig. 1.11.

1.6.2 Computational Design of Mechanisms

One of the reasons for the widespread use of optimization techniques to support toys with machines and mechanisms is the research on toy design at Microsoft Laboratories [2]. The study introduces a method for generating mechanisms that take cyclically animated input information input by the designer and reproduce it as the motion of an automata driven by a single rotating axis shown in Fig. 1.12. Collision avoidance between the parts and the designer's preferred layout can also be found, and the annealing method is mainly used to optimize the parameters. In addition, technologies such as a 3D printer,

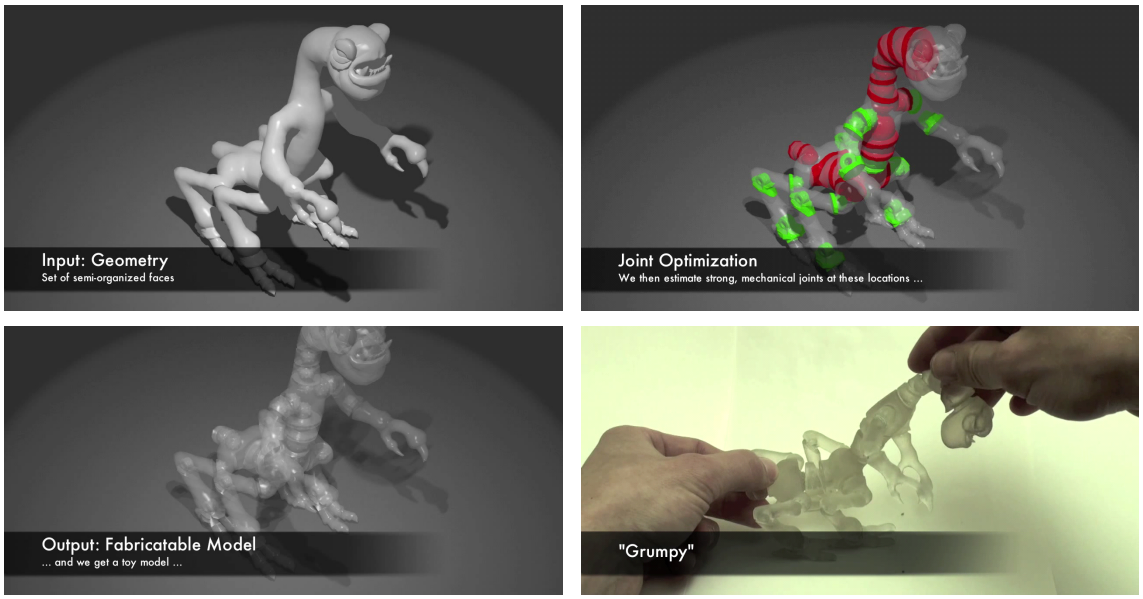


Figure 1.10: The system automatically divides the mesh data of an arbitrary 3D character. If the 3D printer outputs the results of the calculation, it will be possible to deform the posture of the 3D character by moving the angles of each joint by hand.

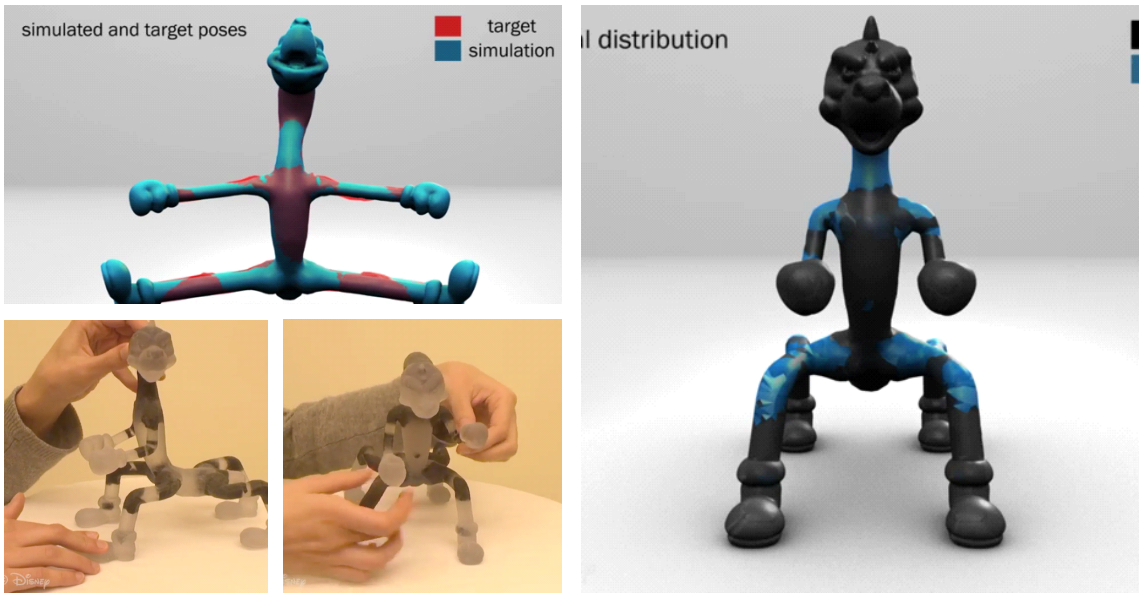


Figure 1.11: In 3D printing of the shape of a 3D character, the system allows to change the posture of a 3D character without articulation breakage by calculating the distribution of materials of different softness to be printed.

which can output the design results directly as a three-dimensional object, can be used to easily materialize automata. This technique provides the first step toward realizing a toy with the intended movement, even for beginners who have no knowledge or experience in mechanical design.

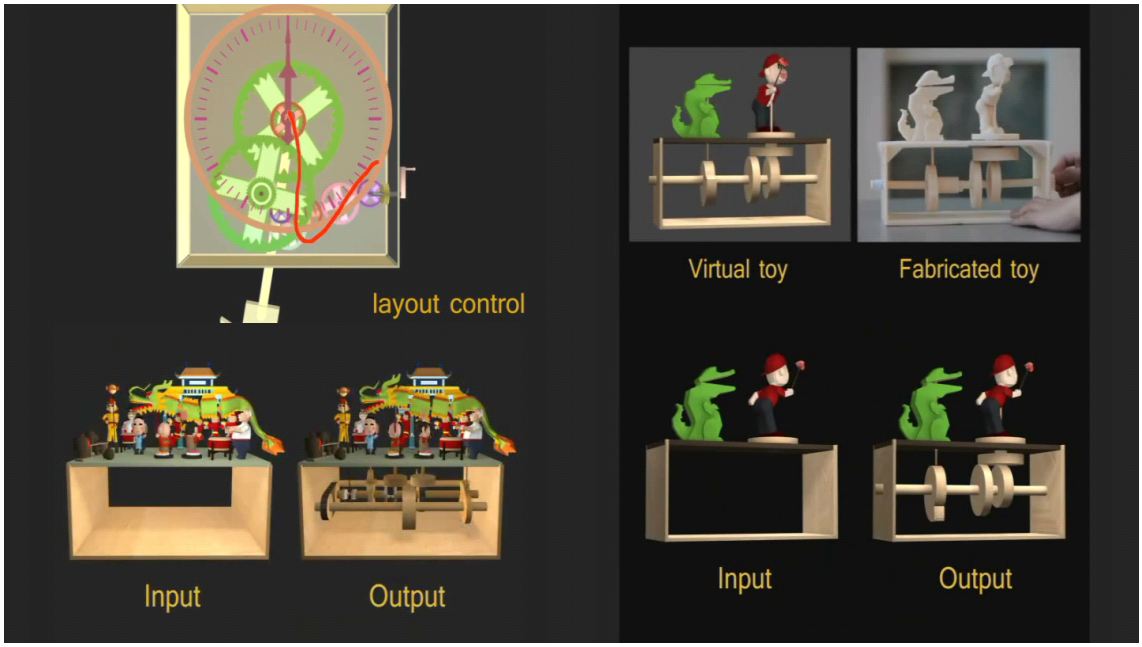


Figure 1.12: The system generates a mechanism that reproduces the movements of the recursive characters specified by the artist and fine-tunes their arrangement so that they can be output to a 3D printer and be moved directly.

The research conducted by Disney research to support the design of a series of character mechanisms has also had a significant impact. For example, it is difficult to design a linkage mechanism that makes the entire machine come alive while rotating a part of the mechanism because the motion of the linkage mechanism is difficult to predict. For such a task, there is research to interactively adjust parameters such as the length and position of the linkage mechanism by giving the desired trajectory to the designer [5, 6]. This research has also been made possible by the development of optimization and additive manufacturing technologies and interfaces shown in Fig. 1.12.

Some research has also been done to automatically generate a linking mechanism that will convert from one desired posture to another desired posture for some shapes drawn by the user's hand [3, 4]. These are not cyclical motions and are therefore suitable

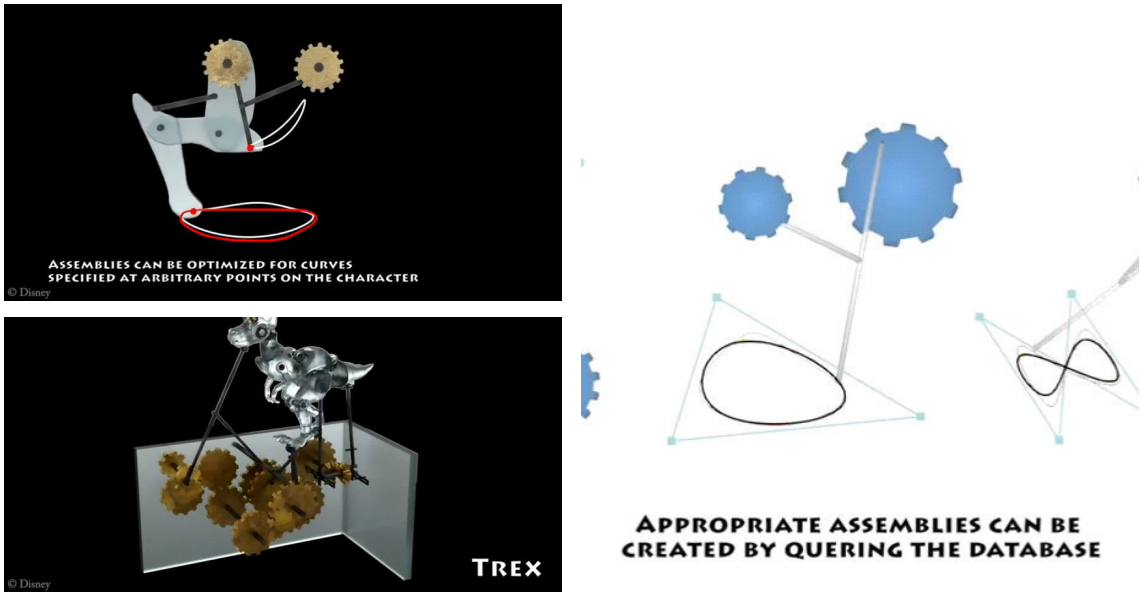


Figure 1.13: The system calculates the parameters and gearing of the linkage mechanism to achieve the trajectory to be followed by the user-specified linkage mechanism endpoint. It simulates the complex motion of the link mechanism and solves the inverse problem.

for designing objects with bending structures, such as furniture or heavy machinery shown in Fig. 1.14.

Methods have also been developed to divide meshes so that they move at joints using 3D printing technology, and to generate mechanisms that contain elasticity [11] shown in Fig. 1.15. There is also a technology that allows elastic objects to be connected with strings and pulled by actuators to achieve animation [10, 12, 13].

1.6.3 Mechanism Design Interface

Computer-aided design (CAD) originated with Sketchpad, developed by Ivan Sutherland in 1963, and is a design aid still used today for machine design shown in Fig. 1.16. It enabled the creation of drawings. Nowadays, CATIA and Solidworks from Dassault Systèmes and Autodesk Inventor and Fusion360 from Autodesk are widely used to follow Sketchpad's intent. Robert and McNeel Associates' Rhinoceros 3D is particularly focused on freeform NURBS modeling. OpenSCAD, which is developed and maintained

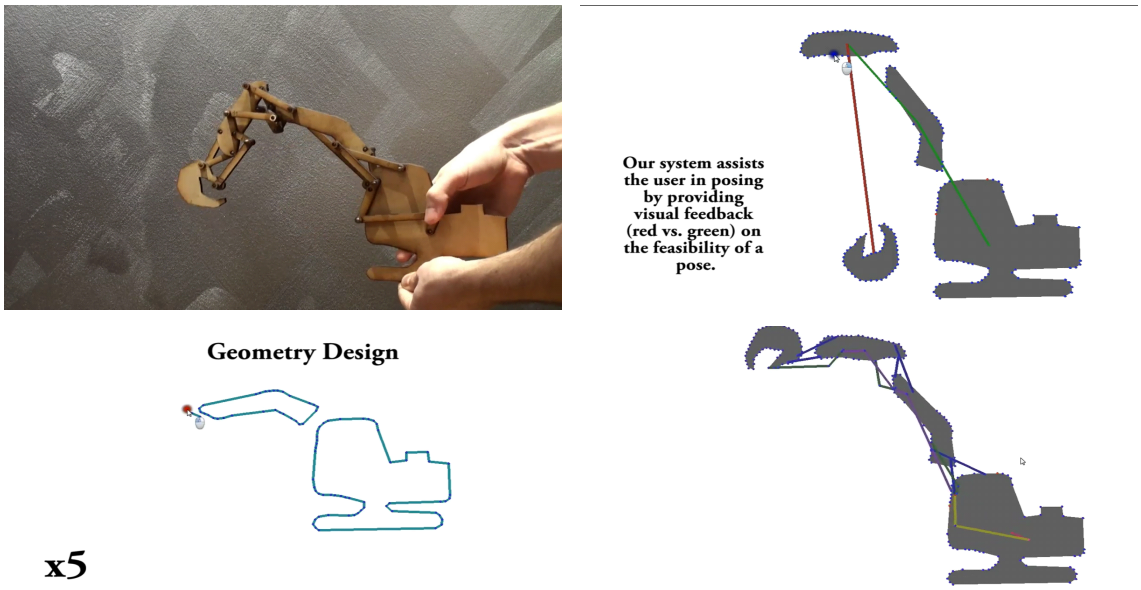


Figure 1.14: A system that generates a linking mechanism capable of moving from a user-specified two-dimensional sketch and multiple postures to interpolate between those postures.

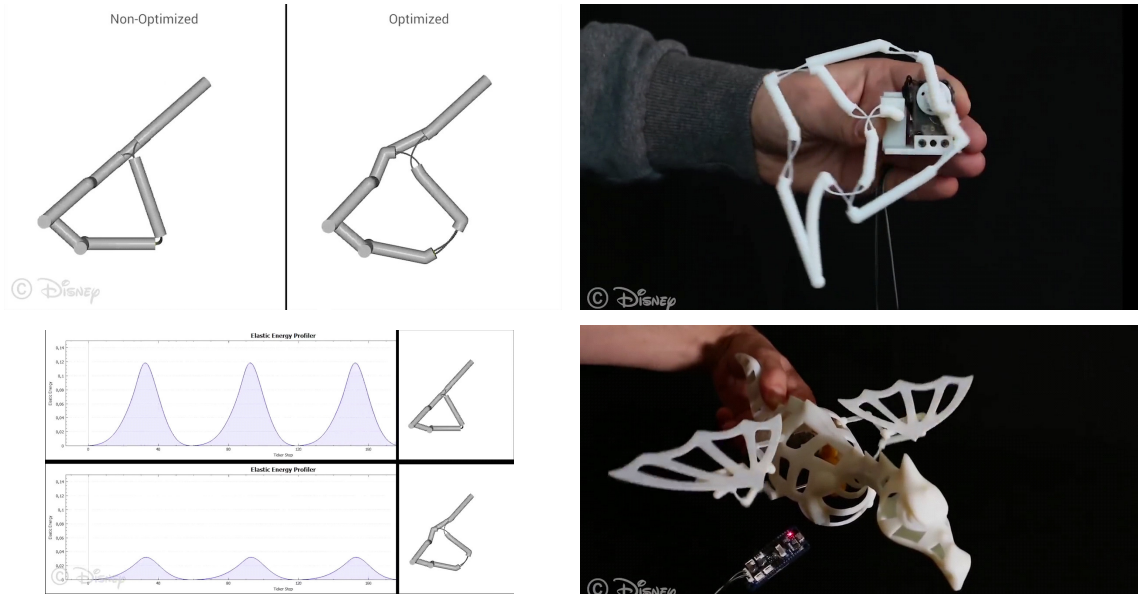


Figure 1.15: A system that generates a soft continuum, such as a linking mechanism, that achieves a similar motion. Complex motion can be achieved in a single 3D printing session without using components such as joints.

by Marius Kintel et al. is a free source software with structural solid geometry (CSG) modeling as its source code. Autodesk's TinkerCAD is also an educational CAD tool that makes it easy to create parts in the browser. Some of the software described here is used to create a machine by creating a three-dimensional shape of each part, and then adding constraints, called joints, to combine the parts into a mechanism.

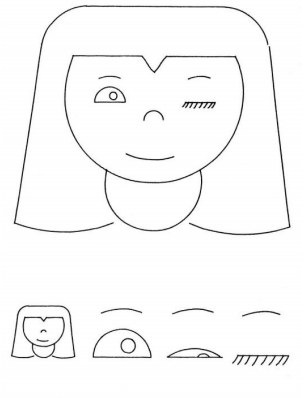


FIGURE 9.8.
WINKING GIRL AND COMPONENTS

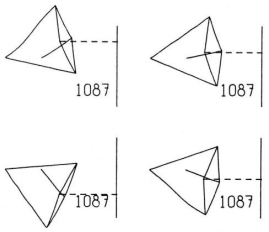


FIGURE 1.6.
FOUR POSITIONS OF LINKAGE
NUMBER SHOWS LENGTH OF DOTTED LINE

Figure 1.16: An early computer-based design support system called Sketchpad. Illustrations, figures and parts can be drawn with various constraints.

Although the design software called CAD is widely used, it does not aim to solve the above design difficulties directly [14].

Looking beyond CAD, educational physics simulators can help create machines: the Algodoo, developed by Emil Ernerfeldt, is a 2D physics simulator for education, specifically allowing mechanisms to be created interactively in virtual space. So far, many user works have been created and published on the Internet. Gazebo and PyBullet are examples of simulations in three-dimensional space, but they do not make designing machines as easy as Algodoo.

The development of mechanisms in games is also worth mentioning. Since the hardware and PCs have been developed with specs that allow for 3D physics simulation inside games, games that allow for the creation of machines in virtual space have also been developed, such as RigidChips and Laputan Blueprints, Garry's Mod developed by Facepunch Studios; LBP and Dreams by Media Molecule; Scrap Mchanic by

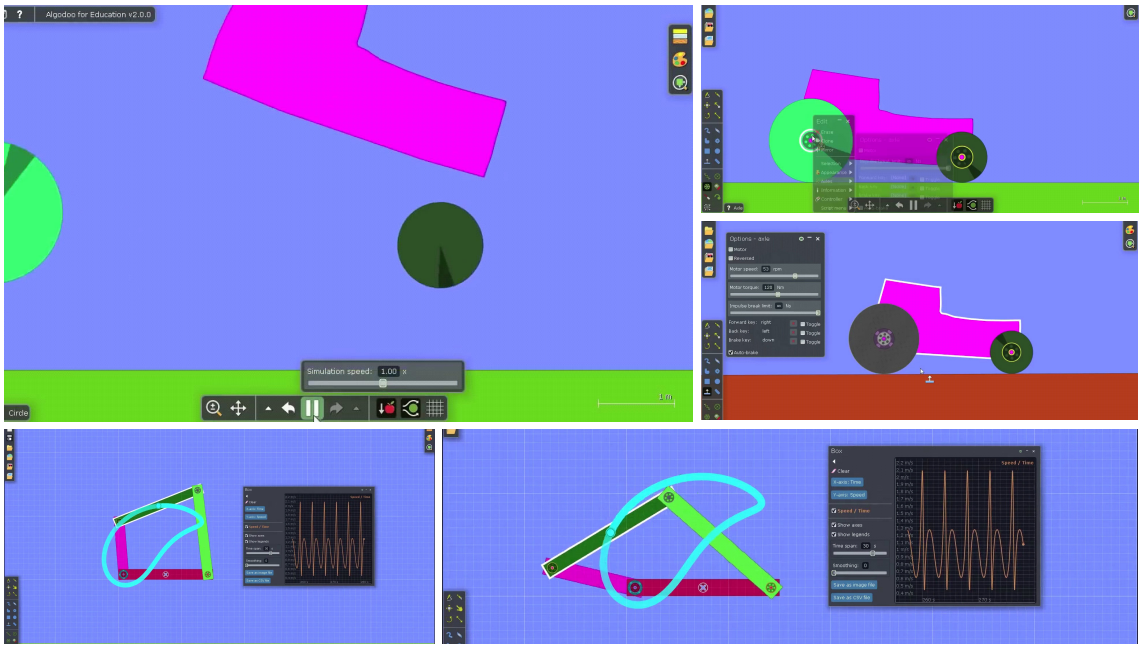


Figure 1.17: An interactive 2D physics simulation called Algodoo is an educational software that can be enjoyed like a game. It can also be used to simulate a mechanism such as a linkage mechanism to verify its operation.

Axolot Games AB; Besiege by Spiderling Studios; and See also Trailmakers from Flashbulb Games. Most of these consist of defining rigid bodies and joints and piecing them together.

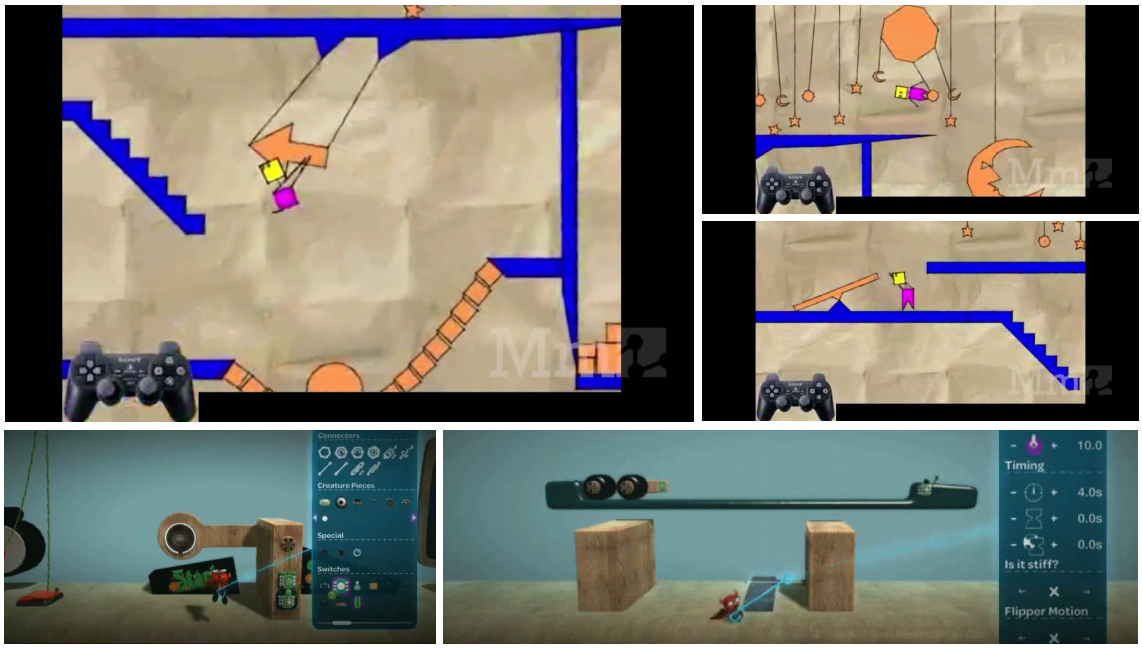


Figure 1.18: Little Big Planet, a PS3 game developed by Media Molecule, features a 2D physics simulator that allows users to generate mechanisms using controllers. Users can upload the items they create to a platform that is run on the Internet and share them with other users.

There is no such thing as working backwards to create or modify each part based on the final function or look of the game, which is all about putting the basic parts together and trying to achieve the desired function through trial and error, while seeing how they behave in the simulator. If we can establish a design method that achieves both appearance and function, we will be able to contribute to the field of CAD and games.

Chapter 2

Computational Design of Balanced Open Link Planar Mechanisms with Counterweights from User Sketches

In this study, the design of a low-vibration multi-joint mechanism is investigated to maintain a stable static balance. Our method augments the user-supplied design with a counterweight whose mass and mounting position are automatically calculated. The optimized counterweight adjusts the center of gravity so that the mechanism returns to its original shape for limited external perturbations. In this study, we present several examples that show how a wide range of user-provided designs can be transformed into statically balanced mechanisms using a sketch-based system. Additionally, validate the results with a series of physical prototypes.

2.1 INTRODUCTION

Designing an articulated mechanism that is both functionally and aesthetically pleasing is a challenging problem at the intersection of art and engineering. A prime example of

this is the 18th-century Edo-era Japanese automata called "Karakuri Ningyo[1]." One of the most notable of these designs is the "Chahakobi Ningyo," a mechanical character that uses the weight of tea to balance itself. In addition, Tanaka Hisashige's "Yumihiki Doji" can shoot a bow by moving each joint; see [15].

When driving these articulated mechanisms with actuators, it is important to consider the weight of the various mechanical components. If poorly designed, the torque required to support the structure can cause the mechanism to vibrate or move incorrectly. For this reason, dead weight compensation strategies to reduce the static loads that must be supported by the actuators have been extensively researched [16, 17].

On the other hand, an under-actuation mechanism such as the "Roly-Poly Toy" allows you to balance effortlessly, even if your posture changes due to external forces [18]. This stable balance can be achieved by adjusting the center of gravity according to the posture. One commonly used means for this purpose is a counterweight. While there is a solution in the case of a single component [18], designing a counterweight for balancing in the case of multiple components is a problem for which no effective solution has been shown.

We propose a method to automatically determine the counterweight parameters so that the corresponding underacting mechanism maintains its balance and returns to this configuration during restricted perturbations by adding to the input design provided by the user. To address this problem, we are guided by three main requirements.

- *Multi-component mechanism:* the mechanism is composed of multiple rigid bodies and joints.
- *Sketch-based input:* the initial design is provided by the user through a sketch-based interface.
- *Stable balancing:* the final mechanism maintains its balance in the target pose, and automatically returns to this pose after bounded perturbation.

As a first step towards this goal, we develop a system that generates an open-

[1] "Karakuri Puppet" on Wikipedia

linked planar balancing mechanism from a user-specified sketch and semi-automatically discovers the counterweight shape using an evolutionary optimization strategy. We demonstrate our method on a set of virtual designs created entirely interactively within our system. To validate the feasibility of our design, we have further built several physical mechanisms.

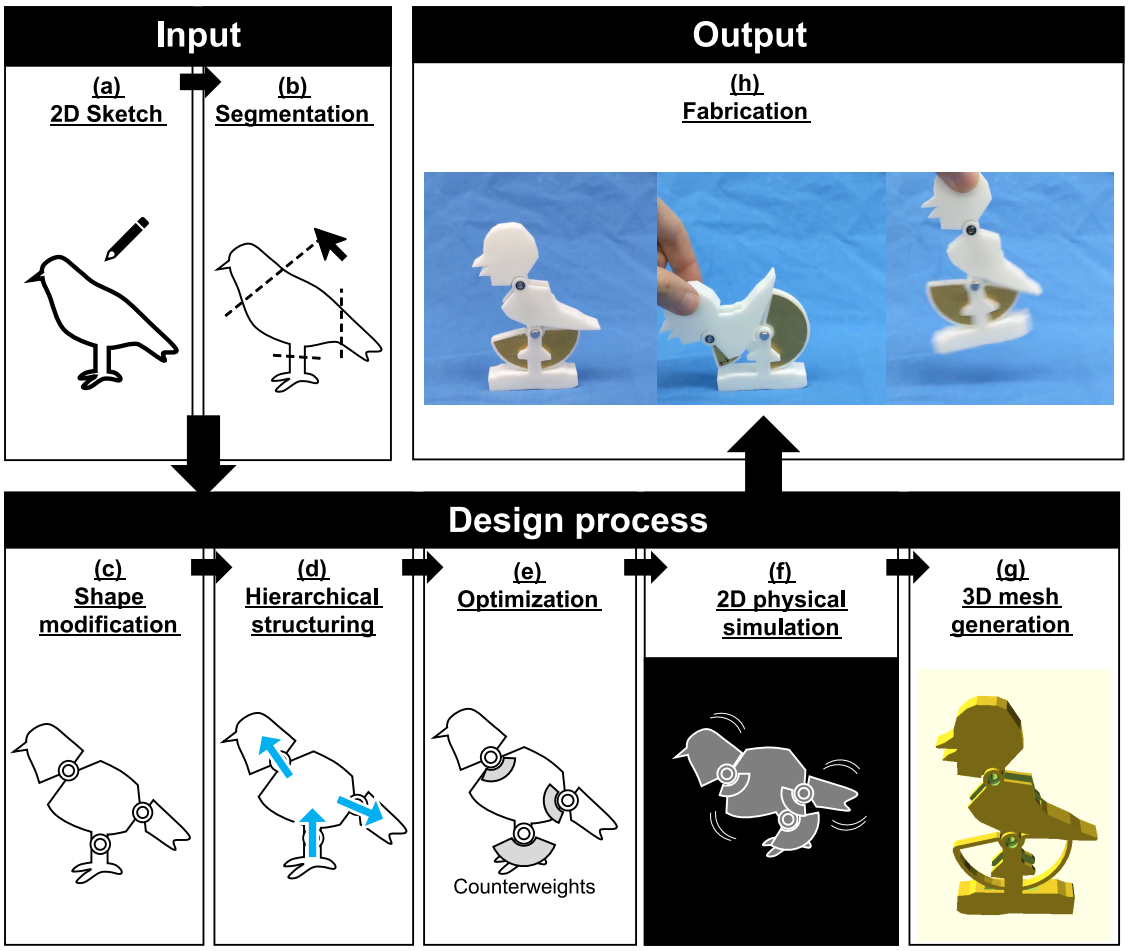


Figure 2.1: Design System Overview. The user draws a sketch and specifies where to divide it. The system then optimizes the parameters of the counterweight based on physical simulations. The final design is converted into a set of 3D printable meshes, ready for fabrication.

2.2 RELATED WORK

Our method builds on previous research in three major research areas: design of mechanical properties, statically balanced mechanisms, and optimization-based design of statically stable objects. These three research areas are described in the following subsections.

2.2.1 Mechanical Character Design

With the increasing availability of additive manufacturing technologies, the computer graphics community is beginning to embrace the design of 3D printable objects that contain functional mechanisms with desired aesthetics [3, 19]. Our work is particularly inspired by the work of Baecher et al. [20] who automatically converted digital letters into components that, once printed, allowed the physical letters to be posed within a desired range of motion. This work has implications for recent methods of creating mechanically animated characters [21, 22], and we aim to extend it to the problem of static balance.

2.2.2 Statically Balanced Mechanism

In the field of robotics and mechanical engineering, there have been many studies on gravity compensation using compliant elements such as counterweights and springs [16, 17]. For example, several studies have proposed methods to design counterweights for specific mechanisms [23, 24, 25]. Takahashi et al. [26] proposed an interactive design system that automatically suggests spring parameters to achieve static balance throughout the space of user-specified configurations, but they did not consider the design of counterweights. In this study, we aim to develop a system that automatically adjusts the parameters of the counterweight to achieve static balance, even for a new articulated mechanism specified in the user's sketch.

2.2.3 Optimization Based Stable Object Design

In the context of design for additive manufacturing, a recent stream of research has addressed the problem of optimizing the center of gravity for a given input shape [27, 28, 29] (see Fig. 2.2). Similarly, Zhao et al. [18] have proposed a design tool for a so-called *roly-poly* toy that returns to its correct posture by itself when pushed down. However, the problem of adjusting the center of gravity to achieve static balance in articulated mechanisms is, to the best of our knowledge, still an unsolved problem.

Figure 2.2: Optimizing the center of gravity of a given input shape.

2.3 METHOD

As shown in Fig. 4.2, the workflow of this system consists of eight steps. First, user-supplied sketches are converted to two-dimensional polygons and the joint positions are automatically determined from user-specified cutting lines. Then, we modify the link shape and create a hierarchical structure as described in Section 2.3.4. The optimization is then performed and the counterweight shape is computed. Physical simulations

are performed using the optimized shapes to confirm that they are balanced. Once the static balance is confirmed, the system generates a 3D shape by Boolean operations, as described in section 2.3.5. Finally, the output shape is 3D printed and connected via a shaft with an off-the-shelf bearing.

2.3.1 Mechanism

We represent an articulated mechanism using a hierarchical structure consisting of n_l links and n_j joints (see Fig. 2.3). A link represents a rigid body and a joint represents a constraint on the relative position of two links. One of the links in the mechanism must be connected to a fixed root link, which represents the vertex of the hierarchical structure. We only consider designs that can be described in this way, which excludes mechanisms with closed loops.

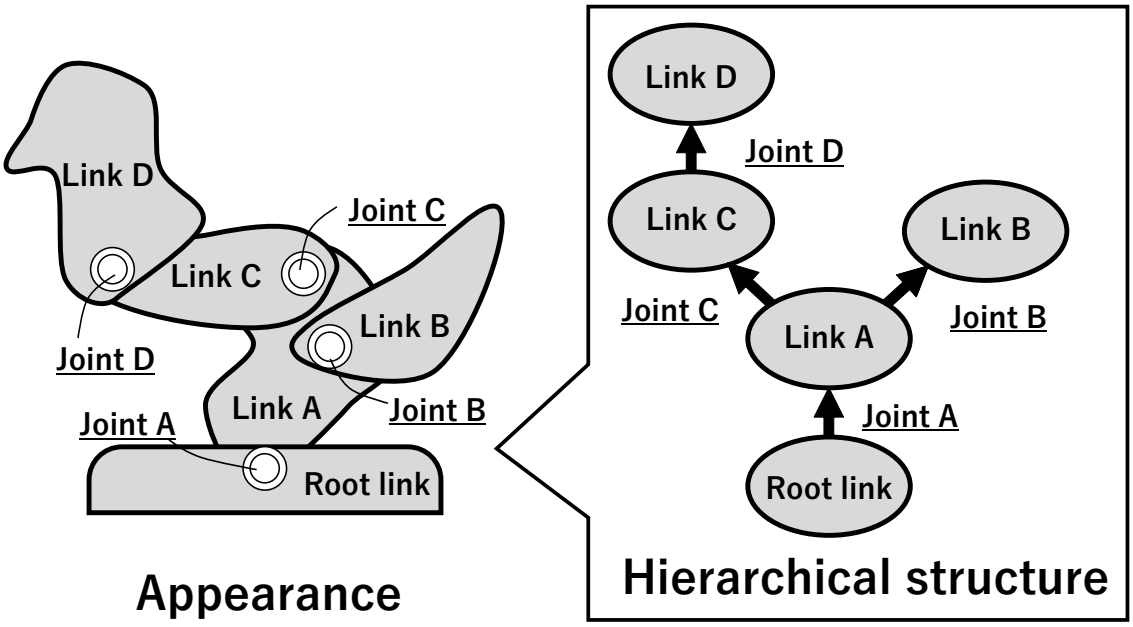


Figure 2.3: A diagram of the hierarchical structure used to represent the mechanism. The mechanism consists of links and joints. A joint defines the positional relationship between a given child link and its parent link.

The link constitutes an articulated mechanism and can change its position and

angle in the planar state. In addition, a link has 2D shape information itself as a rigid body; 2D shape information is a vector of vertices in local coordinates, which is called a 2D polygon. Therefore, a link can be written as

$$\mathbf{l}_i = \left(\mathbf{p}^T, \theta, \hat{\mathbf{s}}^T\right)^T \ , \tag{2.1}$$

where $\mathbf{p}$ is the 2D position in global coordinates, *theta* is the angle, and $\hat{\mathbf{s}}$ is the vector representing the 2D polygon.

Joints are responsible for limiting the positional relationship between two links. Here, we focus only on the so-called pin joints and do not consider other joints, such as sliding mechanisms. A joint is formulated as

$$\mathbf{j}_i = \left(\hat{\mathbf{p}}^T_{\text{parent}}, \hat{\mathbf{p}}^T_{\text{child}}, \hat{\theta}_{\text{range}}\right)^T \ , \tag{2.2}$$

where $\hat{\theta}_{\text{range}}$ represents the range of relative angular movement between a given child link and its parent link, and $\hat{\mathbf{p}}^T_{\text{parent}}$ and $\hat{\mathbf{p}}^T_{\text{child}}$ represent the joint position in local coordinates of the corresponding link.

2.3.2 Counterweights

The counterweight structure consists of n_w weights, and each weight is fixed to a moving link. In order to reduce the dimensionality of the parameters related to the shape of the counterweight, to the annulus sector defined by two radii (r_{inner}, r_{outer}) and angles (θ_{start}, θ_{width}) The shape of the counterweight is restricted (see Fig. 2.4). The counterweight connected to the i-th link can be defined as

$$\mathbf{w}_i = \left(r_{\text{inner}}, r_{\text{thickness}}, \theta_{\text{start}}, \theta_{\text{width}}\right)^T \ , \tag{2.3}$$

where $r_{\text{thickness}}$ is the difference between the inner radius r_{inner} and the outer radius r_{outer} of the annulus sector.

2.3.3 Optimization

The goal of the optimization phase is to find all appropriate counterweight parameters so that the articulated mechanism generated from the user's sketch will be in static equilib-

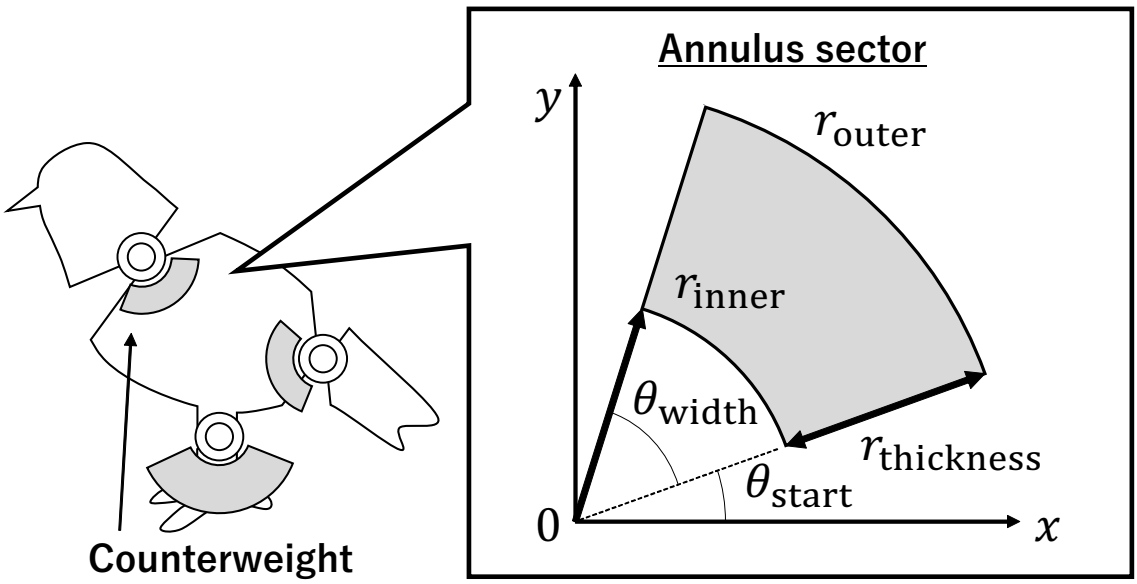


Figure 2.4: We use counterweights in the form of annulus sectors. These shapes are defined by four parameters as indicated.

rium. Here, we summarize the parameters to be optimized $\mathbf{x}$ as in static equilibrium in the target configuration. We summarize the parameters $\mathbf{x}$ to be optimized as

$$\mathbf{x} = \left(\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_{n_w}\right)^T. \tag{2.4}$$

When calculating the center of gravity of a given link in the mechanism, the weights of all its children must be taken into account. For this reason, we recursively compute the center of gravity of all links in the hierarchical structure as follows:

$$\mathbf{p}_i^{\text{CoG}} = \frac{\sum_{j \in \mathscr{I}_i} \left(\rho_{\text{l}} A\left(\mathbf{l}_j\right) \mathbf{c}\left(\mathbf{l}_j\right) + \rho_{\text{w}} A\left(\mathbf{w}_j\right) \mathbf{c}\left(\mathbf{w}_j\right)\right)}{\sum_{j \in \mathscr{I}_i} \left(\rho_{\text{l}} A\left(\mathbf{l}_j\right) + \rho_{\text{w}} A\left(\mathbf{w}_j\right)\right)}, \tag{2.5}$$

where $\mathscr{I}_i$ represents the set of indices of the i-th link and its children, the function A computes the area of a two-dimensional polygon, the function $\mathbf{c}$ calculates the centroid of a two-dimensional polygon, and ρ_{l} and ρ_{w} are the mass densities of the link material and the counterweight, respectively.

If the line connecting the joint position connected to the parent link and its center of gravity (including the influence from all child elements) is not aligned with

the direction of gravity, a non-zero moment will occur and the structure will no longer be statically balanced. Therefore, in order to achieve balance, we define the objective function for calculating the counter-load parameter as follows:

$$f(\mathbf{q}) = \sum_i \| \mathbf{p}_i^{\mathrm{CoG}} - \left(\mathbf{p}_{\mathrm{i,parent}} + \alpha \frac{\mathbf{g}}{\|\mathbf{g}\|} \right) \|^2 , \tag{2.6}$$

$$\min_{\mathbf{q}} \quad f(\mathbf{q}) \quad \text{s.t.} \quad r_{\mathrm{thickness},i} \geq 0,\ \theta_{\mathrm{width},i} \geq 0 \quad \forall i , \tag{2.7}$$

where $\mathbf{q}$ is a vector that accumulates all mechanical information as in

$$\mathbf{q} = \left(\mathbf{l}_1, \mathbf{l}_2, ..., \mathbf{l}_{n_l}, \mathbf{j}_1, \mathbf{j}_2, ..., \mathbf{j}_{n_j}, \mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_{n_w} \right)^T , \tag{2.8}$$

and $\mathbf{p}_{\mathrm{i,parent}}$ is the position of the parent joint, $\mathbf{g}$ is the acceleration of gravity in global coordinates, and α is the target distance from the parent joint (we use $\alpha = 5$ [mm]). This nonlinear minimization problem is solved using CMA-ES [30, 31], a derivative-free method for numerical optimization.

After this optimization calculation, 2D physics simulations are performed to verify the counterweight results and rule out unstable designs. 2D physics simulations are performed using an impulse-based solver [32], which is fast enough for all the examples presented in this study.

2.3.4 2D Sketch Interface

Our system can easily create an open link mechanism from a sketch with a two-dimensional GUI as in Fig. 2.5.

First, contour lines are extracted from a rasterized sketch, or image, drawn by the user, and converted to 2D polygons. A simplification technique known as the Ramer–Douglas–Peucker algorithm [33, 34] is used to convert the rasterized image into 2D polygons.

To divide the 2D polygon converted from the sketch into an articulated mechanism with links and joints, the user draws a line through the GUI to the shape to determine the position of the division (the position of the joint is automatically set to the center of the

divided part). The 2D polygon is then cut based on the desired angular range of motion $\hat{\theta}_{\text{range}}$.

Furthermore, in order to be able to place mechanical joint parts such as shafts and bearings, a circular shape is added to each link with respect to the position where the joint is to be placed, using boolean operations in 2D geometry. This time, we set its radius to $r_{\text{joint}} = 5mm$.

After the user has performed the above partitioning process any number of times and the link shapes have been determined, the hierarchical relationship is defined by determining the joints corresponding to the parent and child links and their link IDs. Specifically, when the user specifies the root link, the link ID of the joint is automatically set according to the distance from the root link.

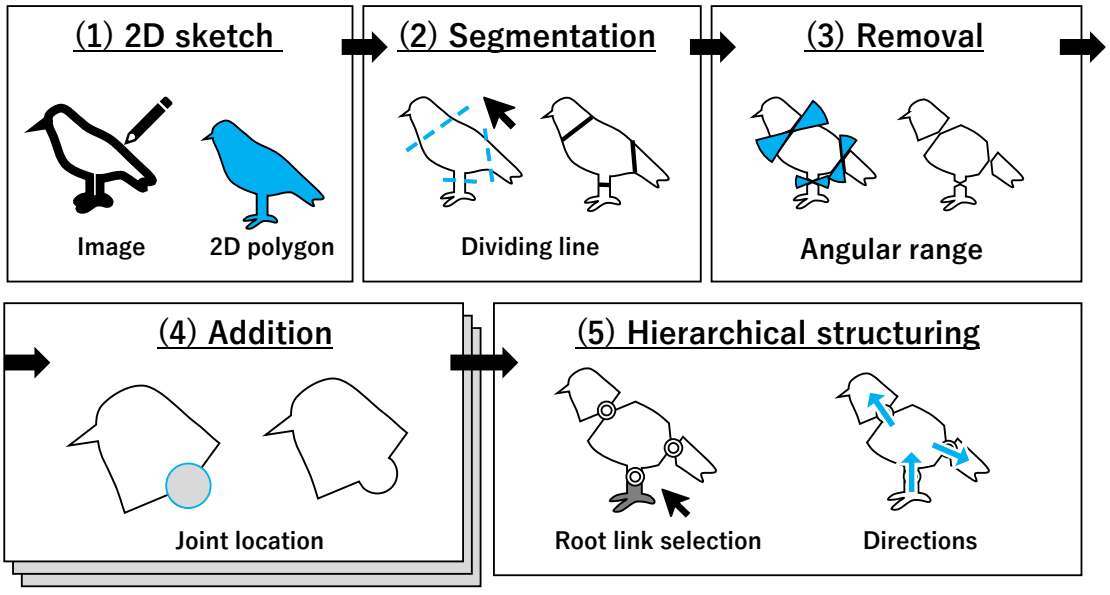


Figure 2.5: Process diagram from 2D sketch to mechanism definition. A user-drawn sketch (1) is polygonized and divided along the division line (2). The links are then cut so that the joints allow for arbitrary angular changes (3) and the geometry for attaching the joints is added (4). We then set up the link hierarchy with the root link selected by the user (5).

2.3.5 3D Mesh Generation

Our system translates optimized 2D shapes into 3D mesh data, as illustrated in Fig. 2.6, using 3D Boolean operations. To avoid collisions between links and counterweights, we design 3D meshes with two layers. The thickness for the inner layer is a_{inner} and the outer layer thickness on each side is $a_{\text{outer}} = a_{\text{inner}}/2$. Furthermore, the clearance between the layers is $a_{\text{clearance}}$ (we use 1 [mm]).

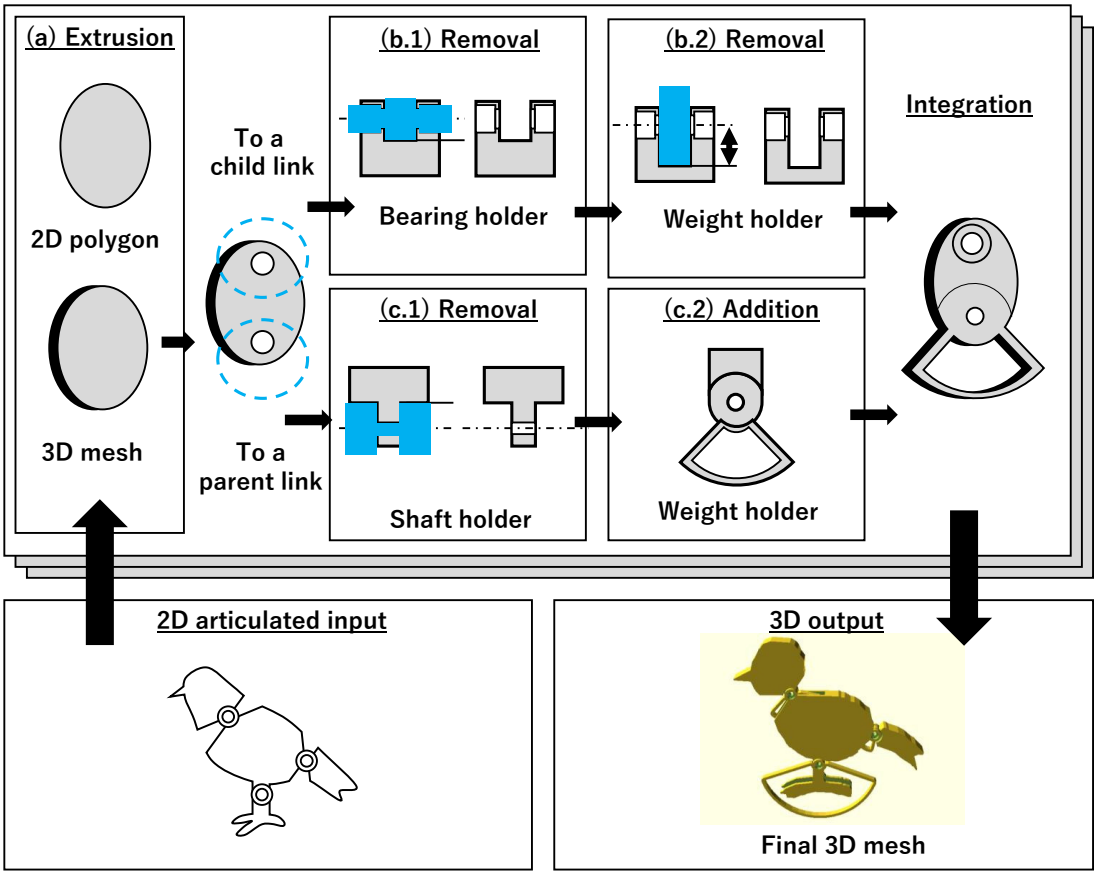


Figure 2.6: Process from mechanism definition to 3D mesh output. The system performs 3D Boolean operations.

First, we extrude the 2D polygons of the link with the thickness $a_{\text{extrude}} = a_{\text{inner}} + 2a_{\text{outer}} + 2a_{\text{clearance}}$, as indicated in Fig. 2.6(a). Around the joint connected to the child side (Fig. 2.6(b)), we cut the inner layer with a radius of $r_{\text{cut}} = r_{\text{joint}} + a_{\text{clearance}}$

or the corresponding counterweight's outer radius $r_{\text{outer}} + a_{\text{clearance}}$. To fit the bearings, we also create a hole with diameter d_{bearing} in the outer layer. However, around the joint connected to the parent side (Fig. 2.6(c)), we cut the outer layer with a radius of r_{cut} and the inner layer with a diameter of d_{shaft}. Thereafter, we add the counterweight fixtures by adding an extruded polygon in the shape of the counterweight. By performing these operations on every link, we obtain a 3D-printable mesh.

2.3.6 Implementation

For numerical optimization using CMA-ES, the linear algebra libraries Eigen[2] and libcmaes[3] are used in C++. For GUI and mechanism visualization, OpenSiv3D[4] is used. For image contour extraction, the contour extraction function of OpenCV[5] is utilized. In addition, Boost geometry[6] is used for simplification, Boolean operations, and center-of-gravity calculation of two-dimensional polygons. In addition, Box2D footnotehttps://box2d.org is used for 2D physics simulation. 3D mesh generation using Boolean operations on 3D geometry is done using OpenSCAD[7] is used to generate 3D meshes using Boolean operations on 3D geometries. All the algorithms used in this study are based on the processed fast enough for interactive operation on a laptop (2.4 GHz, single-threaded).

2.4 FABRICATION AND EVALUATION

In order to ensure that the system developed in this research works properly, several tests were conducted. These tests included both simple and relatively complex examples of the linkage mechanism. For further validation, we also tested several physical prototypes. For this evaluation, a 3D printer (Prusa MK3S) and PLA filament were used to fabricate the links, and stainless steel shafts (diameter $d_{\text{shaft}} = 3$ [mm]) and bearings ($d_{\text{bearing}} = 6$ [mm])

[2]http://eigen.tuxfamily.org
[3]https://github.com/beniz/libcmaes
[4]https://github.com/Siv3D/OpenSiv3D
[5]https://opencv.org/
[6]https://www.boost.org
[7]https://www.openscad.org

were used for the joints. For the counterweight material, brass (thickness $a_{\text{extrude}} = 6$ [mm]) cut on a CNC mill (KitMill RZ420) was used. To calculate the density, $\rho_{\text{l}} = 0.6$ [g/cm^2] and $\rho_{\text{c}} = 5.1$ [g/cm^2] were used. During optimization, the inner radius of the counterweight, r_{inner}, was fixed at 7[mm] for ease of assembly after manufacturing.

2.4.1 Simple mechanisms

For a simple test case, we used three different geometries: a single joint, two serial links, and three branching links. as shown in Fig. 2.7, the proper division was successfully achieved for the geometry drawn by the user. In addition, each mechanism was able to maintain its balance at the specified posture after optimizing the counterweight.

2.4.2 Complex mechanisms

We then adopted three relatively complex shapes: the arm, the cat, and the monster. For the simple shapes, we succeeded in stabilizing the target mechanism shape as shown in Fig. 2.8. However, we also encountered several solutions involving collisions between counterweights, a problem that made them unsuitable for manufacturing. If such collision problems occur, the final results may be improved by adding the sum of the overlapping counterweight shapes to the objective function during optimization.

2.4.3 Physical mechanisms

For experimental validation, we used the generated 3D mesh to create three physical prototypes: a bird, a human, and an arm. The prototypes successfully balanced in the desired shape and returned to their original posture when moderate fluctuations were applied, as can be seen in the video (see Fig. 2.9).

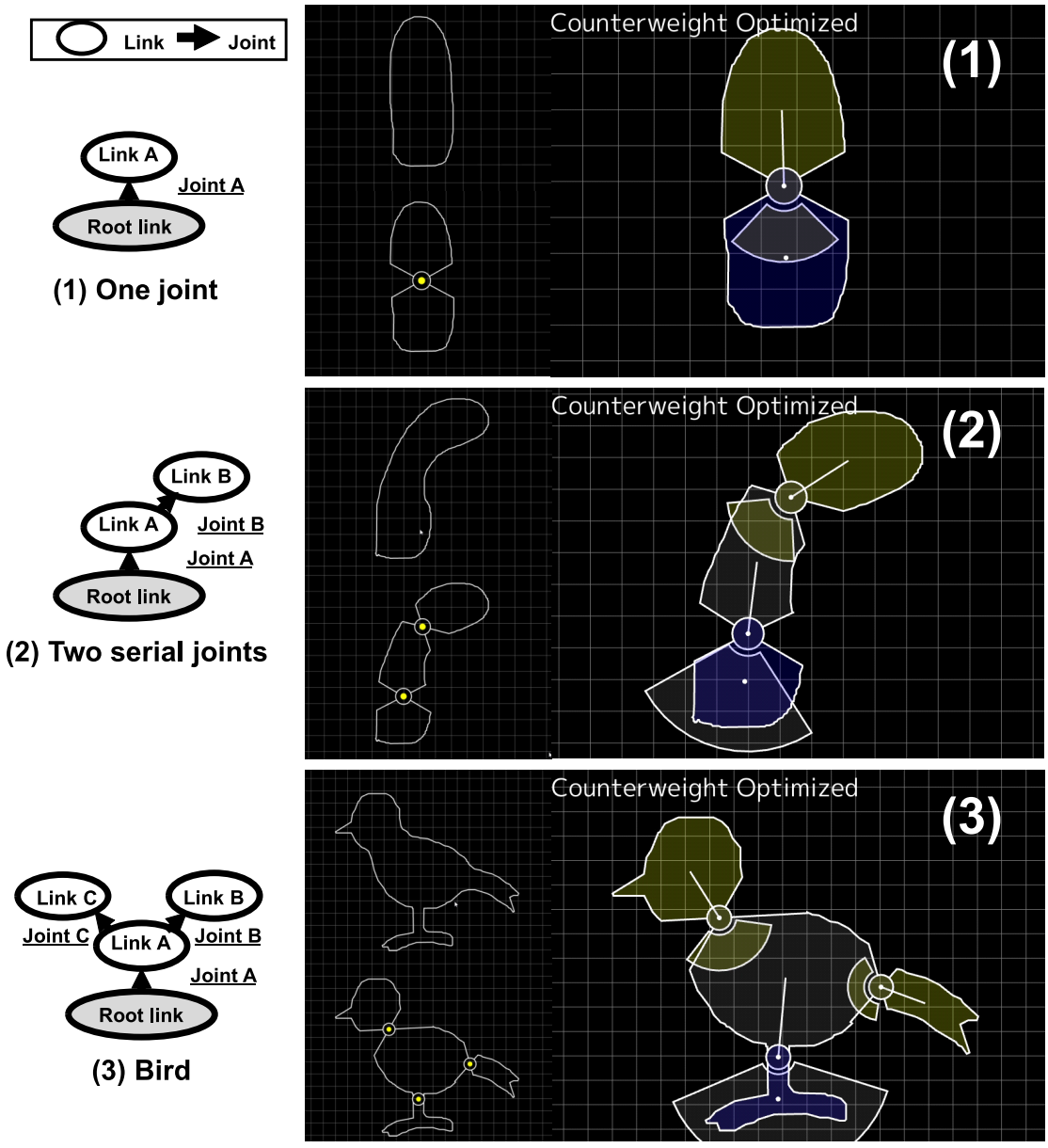


Figure 2.7: Results for simple examples. The counterweights attached the joints allowed for stable balance in all cases.

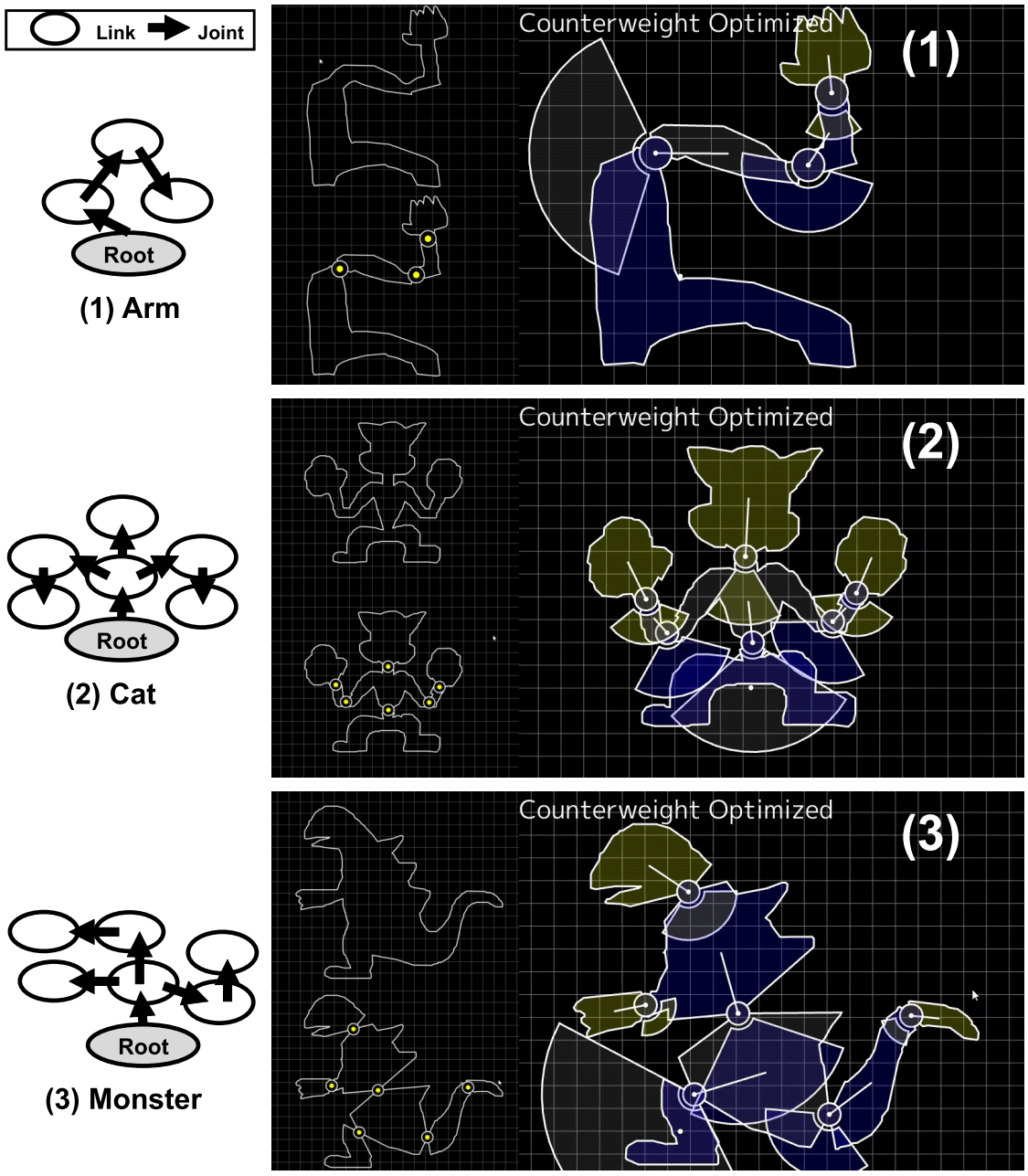


Figure 2.8: Results for complex shapes For simple shapes, the results show a stable balance after parameter optimization of the counterweight. It can be seen that some of the weights are relatively large. Depending on the application, it may be desirable to use thicker or more dense materials.

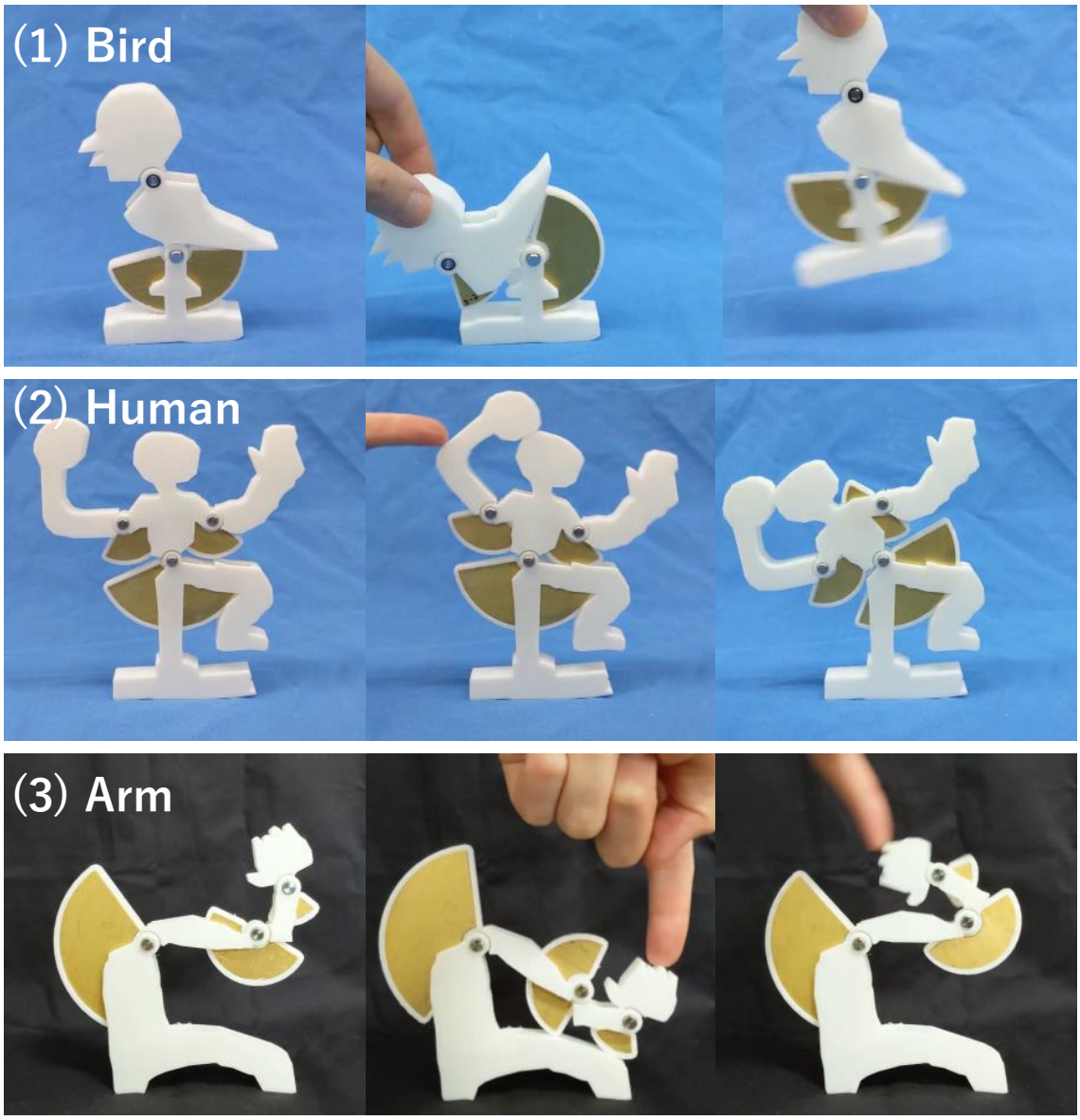


Figure 2.9: Physical prototypes of articulated mechanisms designed using the proposed system. These mechanisms each return to a stable target shape when subjected to moderate perturbations.

2.5 DISCUSSION AND FUTURE WORK

We presented an approach to create a statically balanced mechanism by optimizing the counterweight parameters. Our sketch-based interface allows users to express their design intent in a simple and efficient way. We have demonstrated our method in several virtual designs and physical prototypes. As originally intended, we found that optimizing the counterweight by focusing on the center of gravity was effective in creating a statically stable mechanism. However, the method currently has various limitations and is described below.

First, the system cannot guarantee that the optimization can find a solution within the space defined by the user's input. One solution would be to widen the search space by allowing the optimization process to manage the division of the shape, similar to traditional research [20].

Second, the optimization problem shows multiple optimal solutions, and a given local solution is generally not globally optimal. In our system, the user cannot change the parameters after optimization. However, an iterative process that allows the user to fine-tune the parameters may help to guide the system to a particular solution. Interactive solution space exploration, such as the null-space search scheme described in the previous work [26], can also be useful.

Third, even if the mechanism is statically balanced, the links and joints may collide and the articulated mechanism may not be assembled. For example, the mechanisms in Figs. 2.8(2) and 2.8(3) are well balanced, but the links, joints, and counterweights are in conflict. When designing printable layouts of articulated mechanisms, a constraint-aware modeling approach can solve this problem [21, 35].

In this study, we focused only on the open link mechanism, which is a mechanism that can be expressed in a hierarchical structure. Closed link mechanisms are not currently supported.

Finally, our system does not take into account the weight of the joints; the center of gravity in 2D does not always match the 3D printed one. As a result, some manual

modifications are required before manufacturing the physical prototype. In the future, these aspects should be considered during the design process so that the system can adjust the parameters of the counterweight accordingly.

Chapter 3

Computational Design of Statically Balanced Spring Mechanisms

Statically balanced spring mechanisms are used in many applications that support our daily lives. However, designing a new one is a challenging problem because the designer has to simultaneously determine parameters such as the appropriate number of springs, connectivity, and attachment points. We propose a new optimization-driven approach for designing statically balanced mechanisms in an interactive and semi-automatic manner. In particular, we describe an efficient method for design optimization based on the constant potential energy principle, an automatic sparsification method for spring topology design, and a null-space search scheme that allows the user to navigate the local space of design choices. Simulation examples and several prototypes will be used to demonstrate our design system.

3.1 INTRODUCTION

Pixar's short film "Luxo Jr." shown in Fig. 3.1 is a masterpiece of computer animation. The centerpiece of the story is a lamp placed on a desk, which is animated with the ability to easily switch and balance between any poses [36], and the real-world inspiration for

Luxo Jr. is the well-known Anglepoise lamp [37]. Like angle-poise lamps, many types of balancing mechanisms use counterweights or springs to maintain a static equilibrium in various configurations.

Figure 3.1: Pixar short film "Luxo Jr." Pixar applied the principles of Disney's 2D animation to 3DCG, allowing for realistic movement of characters that was considered impossible in 3DCG at the time.

Many applications that support our daily lives employ statically balanced mechanisms. In consumer electronics, spring-loaded mechanisms make it easier to maneuver heavy doors; in robotics, gravity-compensating mechanisms facilitate the movement of robotic arms, making actuators smaller, lighter, and less expensive; and in medical engineering, passively balanced braces make it possible for patients to stand and walk can be. What these applications have in common is energy efficiency.

There are two ways to design a static equilibrium mechanism: using counterweights or using an elastic body such as a spring [16]. Although the use of counterweights is relatively easy to implement, the use of counterweights increases the total inertia of the mechanism and may reduce the dynamic performance of the mechanism and its resistance

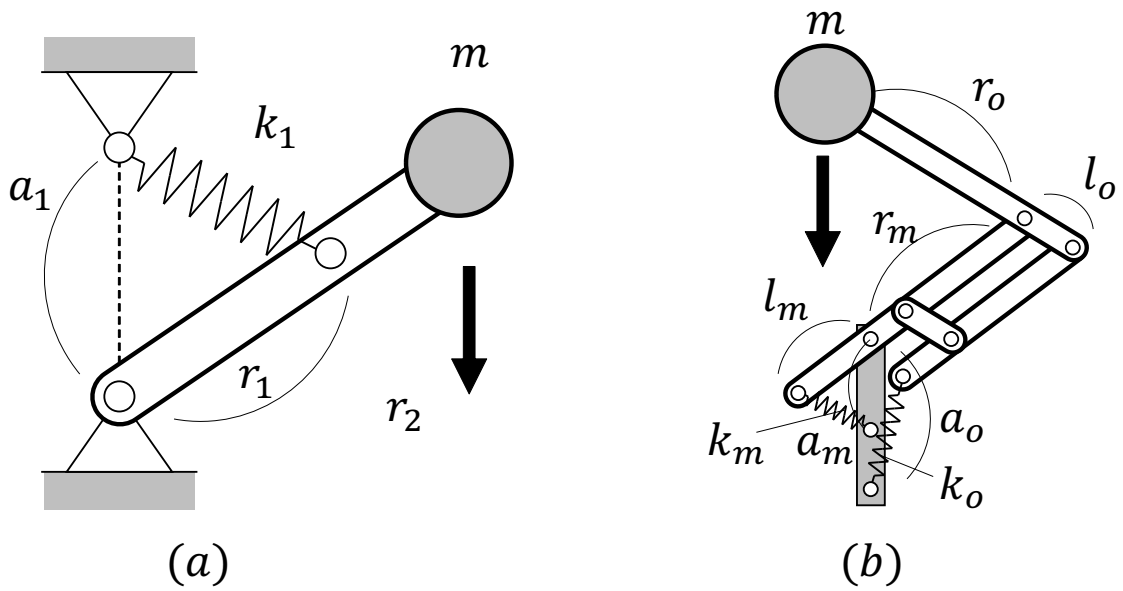


Figure 3.2: Template for a statically balanced spring mechanism. (a) Basic gravity balance mechanism with one spring. (b) Anglepoise mechanism with two springs.

to external forces.

In this study, we focus on achieving static equilibrium using elastic bodies. A well-known example of this class is the basic gravity equilibrium shown in Fig. 3.2(a). The static equilibrium conditions for this single spring mechanism are given by

$$mgr_2 = r_1 k_1 a_1 \ . \tag{3.1}$$

Similarly, the conditions for the double-spring Anglepoise mechanism shown in Fig. 3.2(b) are as follows

$$\begin{aligned} mgr_m &= l_m k_m a_m \ , \\ mgr_o &= l_o k_o a_o \ . \end{aligned} \tag{3.2}$$

The equilibrium conditions in the above examples can be derived using simple geometric reasoning. However, there are few mechanisms for which static equilibrium conditions are explicitly known.

Instead of using explicit balancing conditions, an alternative approach is to require that the total potential energy must be constant across all acceptable configurations

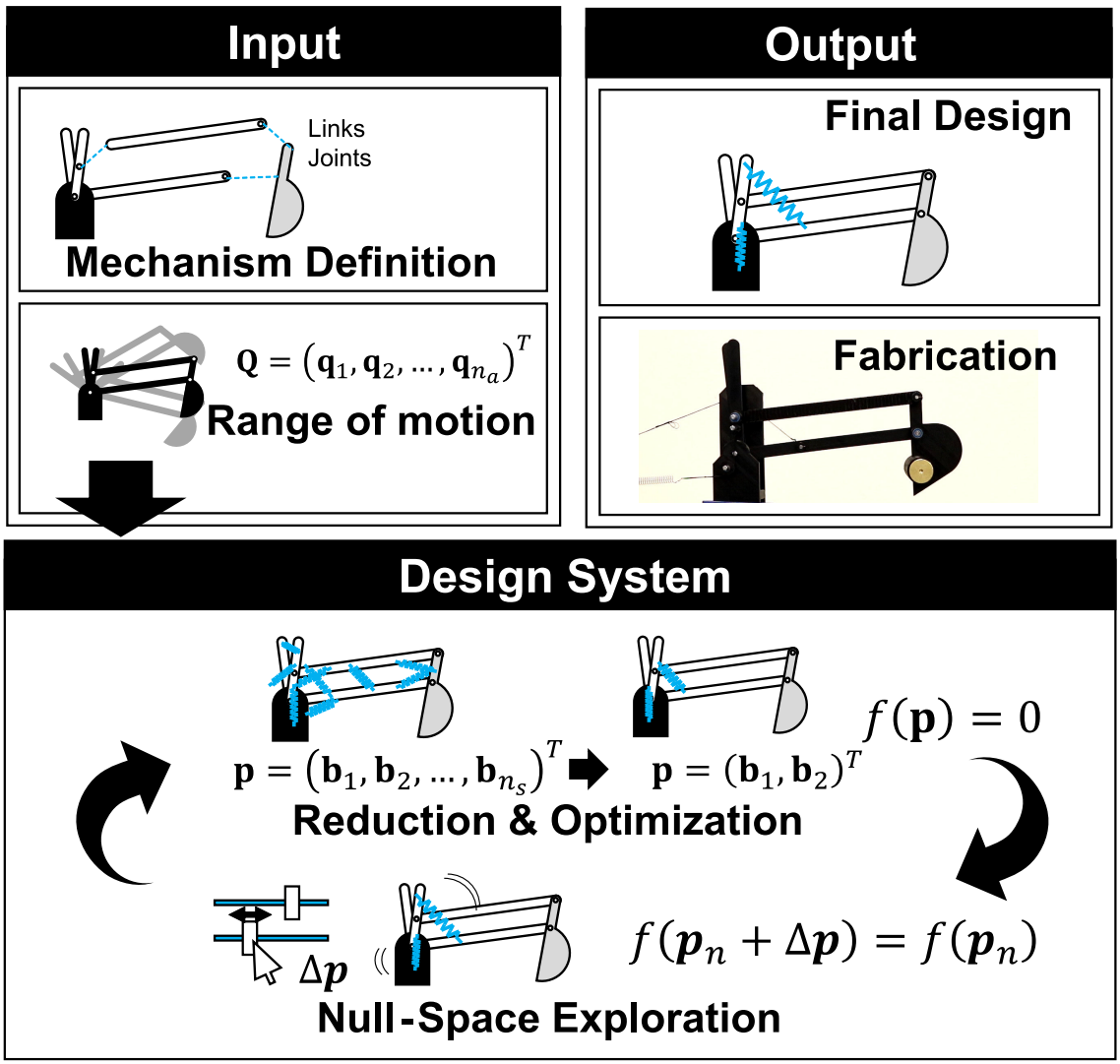


Figure 3.3: Overview of the proposed design system. The method accepts as input the configuration of a rigid joint mechanism and the desired range of motion. Candidate springs are automatically proposed and parameters are optimized for static balancing. Navigating the desired null-space of the static balance allows the user to interactively explore different design options.

of the mechanism [38, 39]. Some studies have proposed design methods based on this principle. For example, the stiffness of a torsional spring can be determined in a graphical manner [40]. As an alternative to rigid joint mechanisms, previous research has examined the shape of the links of compliant mechanisms to achieve static balance [41]. However, a limitation of many of the existing methods of designing static-balance mechanisms is that they assume that the topology of the mechanism (e.g., the number of elastic elements and how they are connected between the links) is known in advance.

Our goal is to establish a computational approach to discover a new statically balanced spring mechanism. Based on the steady-state energy principle, we have developed a fast, optimization-driven method that allows for an interactive search for the spring topology and parameters. To achieve this goal, our method relies on the following technical contributions.

- A formulation for automatically optimizing the design of statically balanced spring mechanisms in the space of a user-defined configuration.
- The topology simplification method using a sparse regularizer can reduce the number of springs needed.
- A null-space exploration method for navigating the space of first-order feasible design variations.

3.2 RELATED WORK

The design of statically balanced spring mechanisms is a fundamental challenge in many fields. For example, in robotics, gravity compensation mechanisms play an important role in improving efficiency [16, 42, 43, 44], and the development of a balanced exoskeleton is another challenging research task [45, 46]. In addition, combining the concepts of static balance and reconfigurability [47, 48] can open the door to many new applications.

With the widespread use of additive manufacturing equipment, there has been a lot of research on optimization-driven tools to simplify the design of rigid joint mech-

anisms [5, 6, 49], compliant mechanisms [11], and cable drive mechanisms [50]. Our work falls into the category of interactive design tools that take a high-level, purpose-guided, optimization-driven approach to simplify design tasks, instead of allowing users to manually specify parameters.

Designing mechanisms optimized for peak torque, joint force and actuator requirements, as well as pure kinematics, is an ongoing research topic [51, 52, 53, 54, 55]. In order to simplify the design of statically balanced compliant mechanisms, several studies have been conducted that focused on adding torsional springs [39, 40, 56, 57] to pin joints in conventional mechanisms. Our approach differs from these studies in that we use a linear spring whose free length and attachment points are interactive and optimized.

3.3 METHOD

We propose a computational method for designing semi-automated and balanced mechanisms. In this method, the input is a conventional mechanism consisting of rigid links and joints. The user defines the desired range of static balance of the mechanism using the set of postures $\mathbf{Q}$. In interactive design, the user explores the space of a feasible, i.e. statically balanced design, using high-level commands and objectives in the context of a graphical user interface (GUI). For example, the user can receive suggestions from the system for inserting springs between specific links or for which springs to remove. Our system then interactively computes the optimized design parameters $\mathbf{p}$ to get as close as possible to the user-defined objective while satisfying the static balance constraints.

3.3.1 Mechanical Model

We model a planar mechanism with rigid links, elastic springs, and mechanical joints. The mechanism consists of n_c links, where each link is a rigid body with three degrees of freedom as

$$\mathbf{a}_i = \left(\theta, \mathbf{z}^T\right)^T \ , \tag{3.3}$$

where θ is an angle, and $\mathbf{z}$ is a two-dimensional center position in global coordinates. For notational convenience, we concatenate the variables of all n_c links into a single vector

$$\mathbf{q} = (\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_{n_c})^T . \tag{3.4}$$

We model elastic elements as linear Hookean springs defined by six parameters

$$\mathbf{b}_i = (l, k, \hat{\mathbf{x}}_A, \hat{\mathbf{x}}_B)^T \ , \tag{3.5}$$

where k is the spring stiffness, l its initial length, and $\hat{\mathbf{x}}_A$ and $\hat{\mathbf{x}}_B$ are attachment locations in local coordinates on the two links, A and B, that the spring connects; see Fig. 3.4. We likewise concatenate the parameters of all n_s springs into a single parameter vector

$$\mathbf{p} = (\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_{n_s})^T \ . \tag{3.6}$$

Joints are modeled as constraints on the geometric relation between a pair of links. We restrict considerations to pin joints and sliding joints in this work and define corresponding constraint functions as

$$\begin{aligned} c_{\text{pin}}(\mathbf{q}) &= \mathbf{x}_A(\mathbf{q}) - \mathbf{x}_B(\mathbf{q}), \\ c_{\text{slider}}(\mathbf{q}) &= (\mathbf{x}_A(\mathbf{q}) - \mathbf{x}_B(\mathbf{q}))^T \mathbf{v}(\mathbf{q}) \ , \end{aligned}$$

where $\mathbf{x}_A(\mathbf{q})$ is the position of the joint on link A in world space (and analogously for $\mathbf{x}_B$), and $\mathbf{v}(\mathbf{q})$ is the world-space version of the unit vector $\mathbf{v} = (-\sin\phi_A, \cos\phi_A)^T$, indicating the sliding direction on link A through a local angle ϕ_A. We collect the constraint functions of the n_j joints into a global vector

$$\mathbf{C} = (c_1, c_2, ..., c_{n_j})^T. \tag{3.7}$$

In order to eliminate global rigid-body motion, we introduce additional constraints to fix the degrees of freedom of individual links.

Given a fully-constrained mechanism, we find the configuration $\mathbf{q}$ that satisfies all constraints by solving the unconstrained minimization problem

$$\min_{\mathbf{q}} \quad \frac{1}{2}\mathbf{C}(\mathbf{q})^T \mathbf{C}(\mathbf{q}) \tag{3.8}$$

using Newton's method.

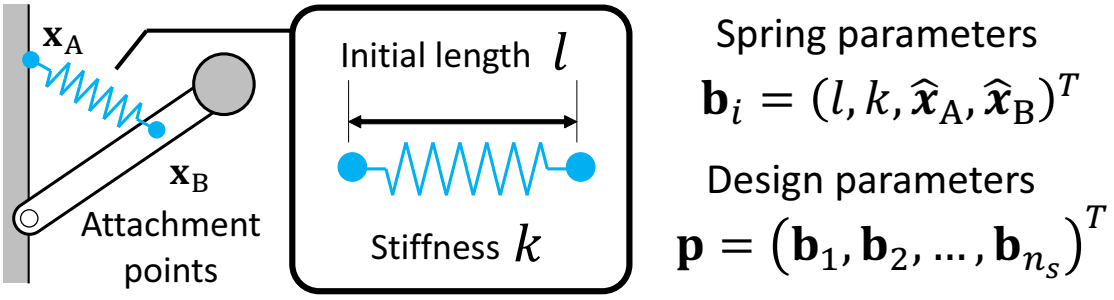


Figure 3.4: The parameters of a spring are its initial length l, stiffness k, and attachment locations $\hat{\mathbf{x}}_A$ and $\hat{\mathbf{x}}_B$; the parameters of all springs are concatenated into the design parameter vector $\mathbf{p}$.

3.3.2 Target Space

The target space is the set of configurations in which the mechanism should be statically balanced. In the discrete setting, we represent this space using n_a individual target configurations $\mathbf{q}_i$ that we stack as

$$\mathbf{Q} = (\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_{n_a})^T. \tag{3.9}$$

To facilitate the creation of $\mathbf{Q}$, we allow the user to specify target angles for one or several joints per target configuration and compute the remainder of $\mathbf{q}_i$ automatically. To this end, we introduce angular constraints

$$c_{\text{angular}}(\mathbf{q}) = (\theta_B(\mathbf{q}) - \theta_A(\mathbf{q})) - \hat{\theta}_\mathbf{i}\,, \tag{3.10}$$

where $\hat{\theta}_\mathbf{i}$ denotes the target angle specified by the user. We add these angular constraints to the joint constraints $\mathbf{C}$ and compute the target configurations $\mathbf{q}_i$ by minimizing (3.8) as before. If the resulting minimum is nonzero, we notify the user that the target angles are conflicting with mechanical constraints. Once the target configurations have been computed, they remain fixed throughout the remainder of the design process.

3.3.3 Optimization

With the target configurations defined, the goal is now to find spring parameters $\mathbf{p}$ such that the mechanism is statically balanced. To this end, we start by defining the total

potential energy for a given pose as

$$U(\mathbf{p}, \mathbf{q}_i) = E_{\text{gravity}}(\mathbf{q}_i) + E_{\text{spring}}(\mathbf{p}, \mathbf{q}_i) \ . \tag{3.11}$$

The gravitational and elastic components are defined as

$$E_{\text{gravity}}(\mathbf{q}_i) = -\sum_{j}^{n_c} m_j \mathbf{z}_j \cdot \mathbf{g} \ , \quad \text{and} \tag{3.12}$$

$$E_{\text{spring}}(\mathbf{p}, \mathbf{q}_i) = \frac{1}{2} \sum_{j}^{n_s} k_j \left(\|\mathbf{x}_{B,j} - \mathbf{x}_{A,j}\|_2 - l_j \right)^2 \ , \tag{3.13}$$

where $\mathbf{g}$ is the gravitational acceleration, m_j is the total mass of link $\mathbf{a}_j$, and $\mathbf{z}_j$ its position.

A direct approach to static balance would be to ask that the net generalized forces vanish for every target pose, i.e.,

$$\mathbf{F}(\mathbf{p}, \mathbf{q}_i) = -\frac{\partial U(\mathbf{p}, \mathbf{q}_i)}{\partial \mathbf{q}_i} = \mathbf{0} \quad \forall i \ . \tag{3.14}$$

Solving these nonlinear equations directly is, however, not an attractive option from an algorithmic point of view: measuring progress is difficult in the absence of a proper objective function, and the over-determined nature of the system leads to additional overhead.

In order to avoid these problems, we leverage the observation that, if the total potential energy is constant throughout target space, then its gradient must vanish at every point; see also Gallego et al. [39]. Instead of requiring force equilibrium, we therefore ask that the total potential energy be the same for all target poses, i.e.,

$$U(\mathbf{p}, \mathbf{q}_i) = \text{const.} \quad \forall i \ . \tag{3.15}$$

We note that the conditions for force equilibrium and constant energy are only equivalent if they hold point-wise, i.e., for every configuration in target space. While enforcing constant energy for each target pose $\mathbf{q}_i$ does not guarantee force balance in the strict sense, we have never observed a case of constant potential with force imbalance during our experiments.

To implement the constant-energy approach, we must know the value of the energy. However, since this value is not known *a priori*, we construct an objective function

that attracts the potential of the individual poses $U(\mathbf{p}, \mathbf{q}_i)$ to their mean $\overline{U}(\mathbf{p})$ as

$$f_{\text{balance}}(\mathbf{p}) = \frac{1}{2}\sum_{i}^{n_a}\left(U(\mathbf{p}, \mathbf{q}_i) - \overline{U}(\mathbf{p})\right)^2 , \tag{3.16}$$

$$\overline{U}(\mathbf{p}) = \frac{1}{n_a}\sum_{i}^{n_a} U(\mathbf{p}, \mathbf{q}_i) . \tag{3.17}$$

This leads to an unconstrained minimization problem for the unknown spring parameters $\mathbf{p}$—the configurations $\mathbf{q}_i$ are fixed—that we solve using Newton's method. A zero value for the solution indicates a valid, statically balanced design. However, such a solution does not necessarily exist as explained next.

3.3.4 Spring Reduction

The existence of the zero-balance objective $f_{\text{balance}}(\mathbf{p})$ depends on the topology of the selected springs, i.e., the number of springs and the links they connect. Finding a sparse and effective topology that can reduce the balance energy to zero with a small number of springs is, in general, not an easy task. To support users in this process, we start with an exhaustive set of springs, including all paired link connections, and then introduce a sparse objective to gradually remove unneeded springs. For this purpose, we define an L_1-norm sparsity objective

$$f_{\text{sparsity}}(\mathbf{p}) = \|\mathbf{k}\|_1 = \sum_{i}^{n_s} |k_i| , \tag{3.18}$$

where $\mathbf{k} = (k_1, k_2, ..., k_{n_s})^T$ gathers per-spring stiffness coefficients k_i. We then combine sparsity and static balance into a single objective function

$$f(\mathbf{p}) = \alpha f_{\text{balance}}(\mathbf{p}) + (1-\alpha) f_{\text{sparsity}}(\mathbf{p}) , \tag{3.19}$$

where α is a user-provided weight (we use $\alpha = 0.5$).

The combined objective is augmented with a set of bound constraints that prevent initial lengths and the stiffness values from becoming negative. We solve the corresponding optimization problem

$$\min_{\mathbf{p}} \quad f(\mathbf{p}) \quad \text{s.t.} \quad l_i \geq 0,\ k_i \geq 0 \quad \forall i , \tag{3.20}$$

using sequential quadratic programming (SQP). If we detect a sufficiently small k_i when $f(\mathbf{p}) = 0$, we remove the corresponding spring and solve again. Once no further spring can be removed, we compute the final design parameters by solving (3.20) again using only the balance objective.

3.3.5 Null-Space Exploration

For a given spring topology, there is a space of parameters that generally satisfies the static balance condition. Some solutions from this space may be preferable to others in terms of functional or aesthetic goals. For example, a user may want to avoid a design in which the spring attachment point is far from the link, as shown in Fig. 3.5. In such cases, instead of introducing regularization to favor a particular solution, we allow the user to interactively explore the space of feasible designs and choose the one they prefer. We refer to this process as null-space exploration.

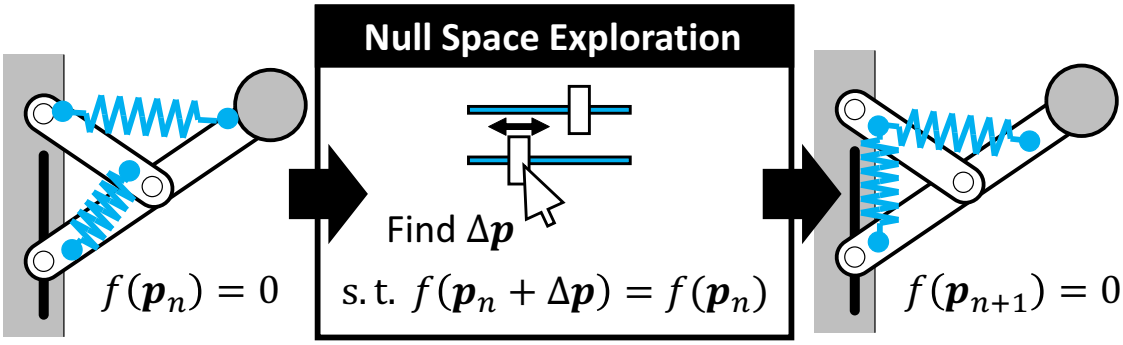


Figure 3.5: Null-space exploration. The user manipulates a specific spring parameter by dragging the slider. All other parameters are automatically adjusted so that the mechanism remains statically balanced.

To ensure the feasibility of the design during the null space search, the parameter change $\Delta\mathbf{p}$ requires that the balance objective and its gradient remain unchanged, i.e.

$$f(\mathbf{p}+\Delta\mathbf{p}) = 0 \quad \text{, and} \quad \frac{\partial}{\partial \mathbf{p}} f(\mathbf{p}+\Delta\mathbf{p}) = 0\,. \tag{3.21}$$

Expanding the second equation around $\mathbf{H}$ yields the first-order condition for maintaining

optimality,

$$\frac{\partial^2 f(\mathbf{p})}{\partial \mathbf{p}^2}\Delta\mathbf{p} = \mathbf{H}\Delta\mathbf{p} = \mathbf{0}\ . \tag{3.22}$$

The above condition is equivalent to requiring that any parameter change in $\Delta\mathbf{p}$ must reside in the null space of $\mathbf{H}$. To enforce this condition, we use eigenvalue decomposition to construct a null-space basis $\mathbf{Z}$ for $\mathbf{H}$ and require that $\Delta\mathbf{p} = \mathbf{Z}\mathbf{w}$ exists for some $\mathbf{w} \in \mathbb{R}^m$ where m is the number of zero eigenvalues.

During the null-space exploration, the user interactively adjusts the spring parameters using a series of sliders as shown in Fig. 3.5. These adjustments produce the target change $\Delta\bar{p}_i$ for a given parameter p_i. Then, the desired change is projected into the null-space of $\mathbf{H}$ by solving the minimization problem

$$\min_{\mathbf{w}}\ \beta\|\mathbf{Z}\mathbf{w}\|_2^2 + (1-\beta)\left[(\mathbf{Z}\mathbf{w})_i - \Delta\bar{p}_i\right]^2\ . \tag{3.23}$$

In the above equation, β balances between the conflicting goals of achieving the desired change in parameters and minimizing the change in the remaining parameters.

3.3.6 Implementation

It was implemented in C++ using Eigen [1] for linear algebra and OpenSiv3D [2] for visualization. In all of the examples presented in this study, all of our algorithms are fast enough to run at interactive speeds on a standard desktop PC (2.8 GHz, single thread).

3.4 RESULTS

In this study, we evaluated our method using a series of simulation examples and several physical prototypes. The first experimental example was aimed at confirming the basic properties, while the next example demonstrates the potential of our method for constructing a new, statically balanced, complex mechanism. The design time for the examples presented here is a few minutes for the simplest mechanisms (e.g., Fig. 3.6) and

[1] https://eigen.tuxfamily.org

[2] https://github.com/Siv3D/OpenSiv3D

less than 30 minutes for the most complex ones (Fig. 3.9, (b)). The largest proportion of this time was spent exploring design variations of different spring topologies.

3.4.1 Validation

In order to verify that our method is capable of the designs known in the literature, we applied it to the gravity equilibrator shown in Fig. 3.2(a). The parameters computed using our method correspond to the analytical solution predicted in Eq. 3.1. The null space search also allows the user to quickly create design variations with different spring attachment positions, as shown in Fig. 3.6. Furthermore, the null-space exploration allows us to numerically verify that the initial length of the spring is zero.

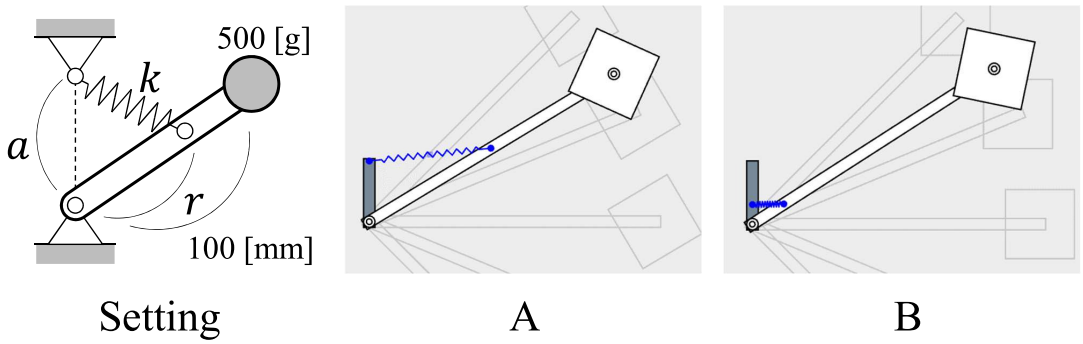


Figure 3.6: Two design variations (*middle*, *right*) for the gravity equilibrator (*left*) generated with our method.

To evaluate the effectiveness of sparsity, we revisit the well-known Anglepoise mechanism presented in Fig. 3.7. We start with an exhaustive set of springs as a first guess and run several iterations of the reduction process described in Sec. 3.3.5. The result is a functional and statically balanced mechanism, as shown in Fig. 3.7 (b), which is an alternative solution to the original design shown in Fig. 3.7 (a).

Fig. 3.8 shows a simple-example illustrating the null space exploration method proposed in this study. In the first design, the springs are mounted far from the links, which is not convenient for manufacturing. Therefore, the user was able to obtain a more compact design by interactively adjusting the parameters of the spring using the null-

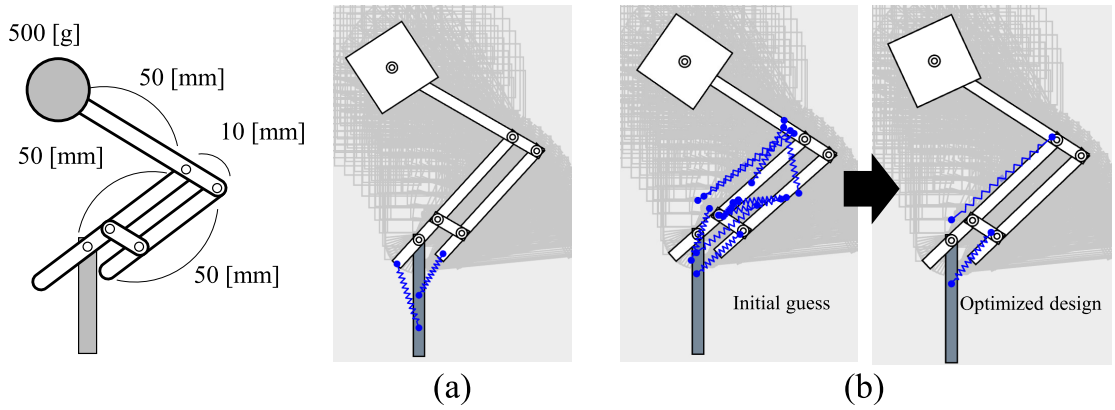


Figure 3.7: Automatic spring topology design shown for an anglepoise mechanism (*left*). Starting from an exhaustive set of springs, our method finds an alternative solution with two springs by automatic sparsification.

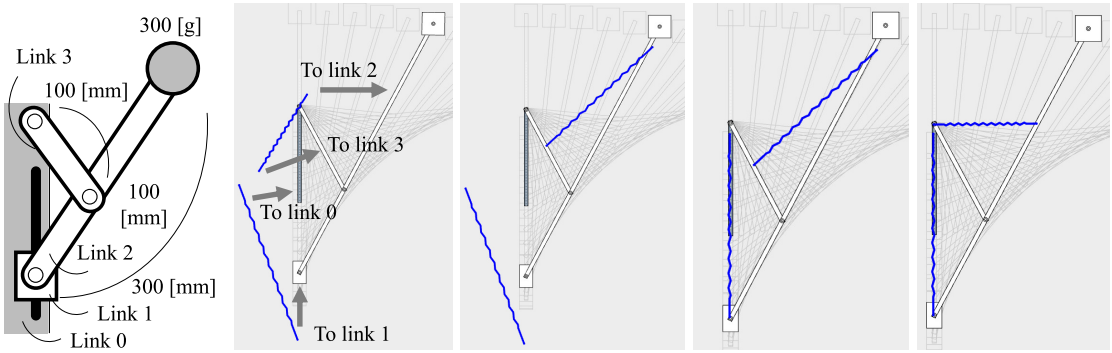


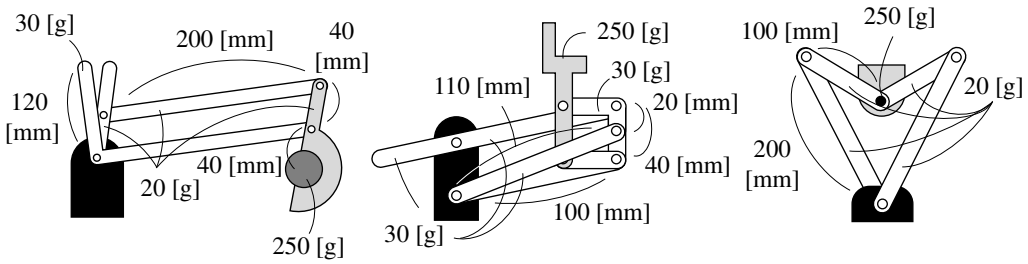
Figure 3.8: Using null-space exploration, the user creates a more compact design through a sequence of three parameter adjustment steps.

space exploration method.

3.4.2 Design Examples

While the method can rapidly generate variations of existing statically balanced mechanisms, its real strength lies in its ability to be applied to mechanisms for which no statically balanced solution exists. To demonstrate the potential of our method for this task, we have selected three relatively complex mechanism designs, as shown in Fig. 3.9.

The links of the mechanism were 3D printed using PLA filament. For the joints, industrial ball bearings and steel shafts were used to reduce friction. A load of 2.5 [N] was applied to obtain the desired load capacity (Figure 3.13). Additionally, to facilitate manufacturing, off-the-shelf springs were used in the manufactured prototype. A pulley and a cable were used to adjust the effective free length of the spring [42].



(a) Excavator (b) Weightlifter (c) Symmetric five-bar linkage

Figure 3.9: Three rigidly-articulated mechanisms that were used as input for our method to create statically balanced versions.

You can see the one we made in Fig. 3.12. As you can see in the video, our mechanism is easy to move and can maintain static equilibrium in a variety of shapes. Please also see Fig. 3.13.

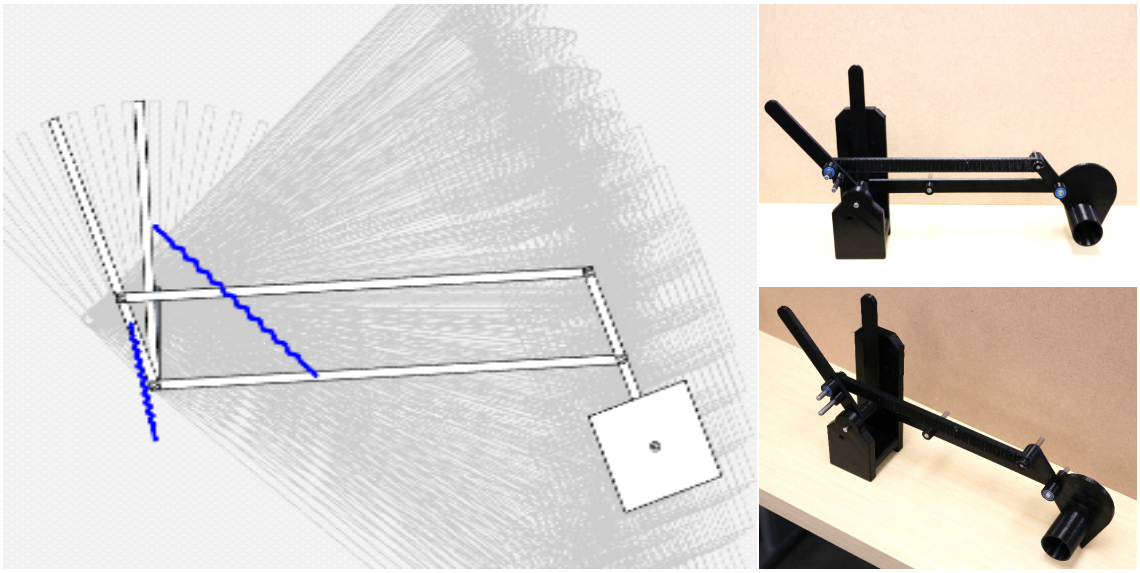


Figure 3.10: Optimized designs (*left*) and manufactured mechanisms (*middle*, *right*).

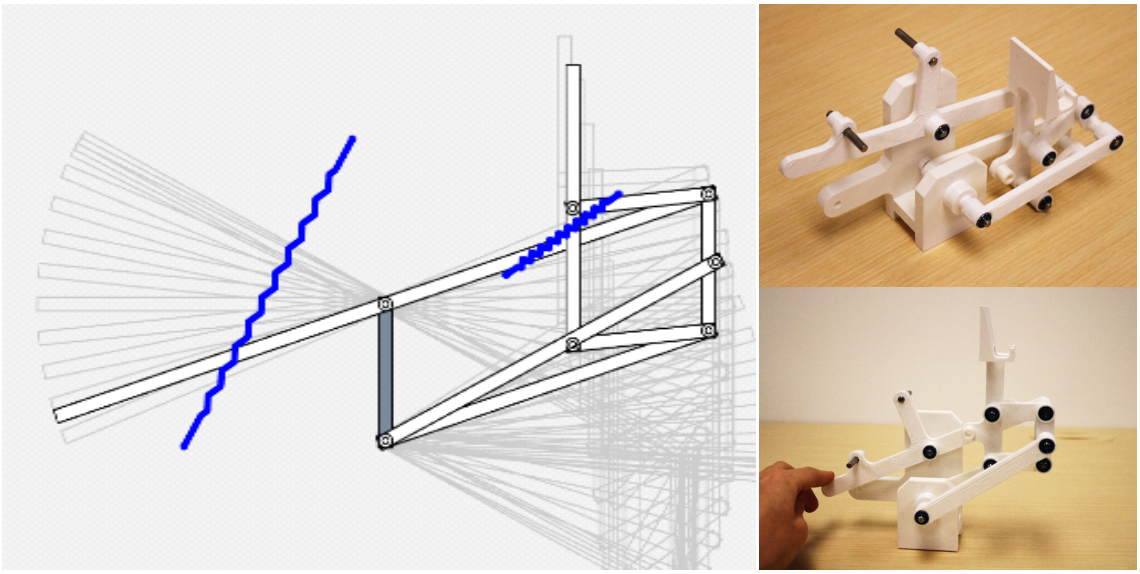


Figure 3.11: Optimized designs (*left*) and manufactured mechanisms (*middle*, *right*).

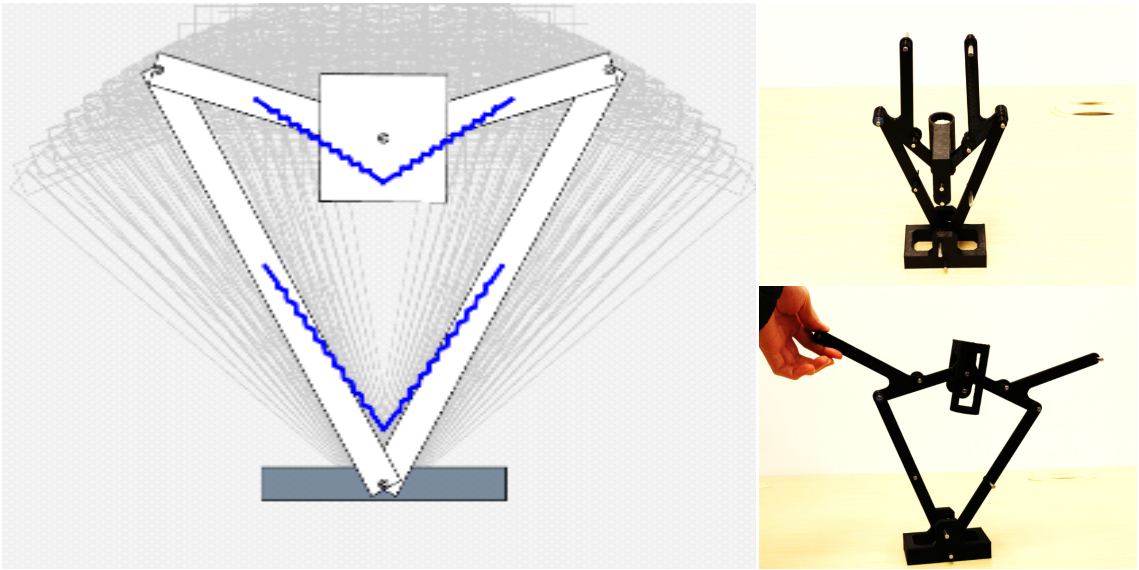
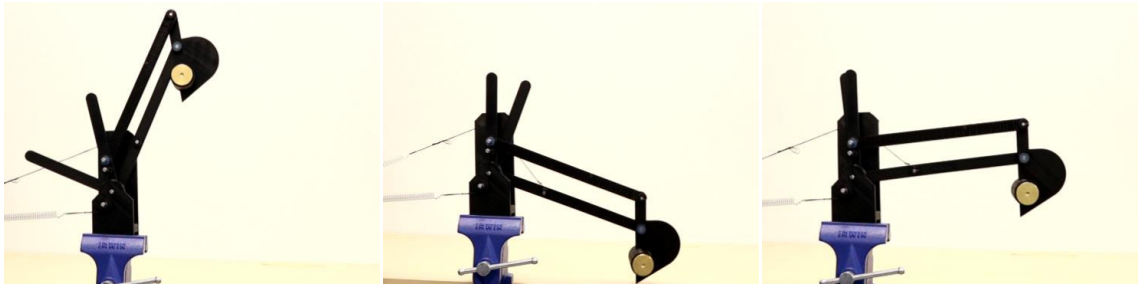
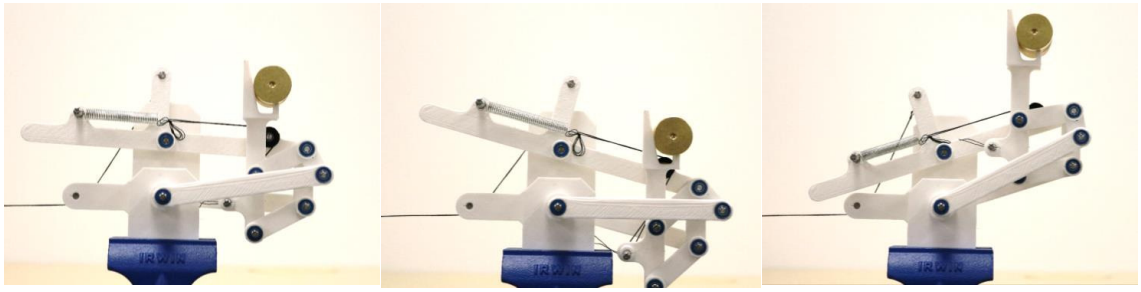


Figure 3.12: Optimized designs (*left*) and manufactured mechanisms (*middle*, *right*).



(a) Excavator



(b) Weightlifter

(c) Symmetric five-bar linkage

Figure 3.13: Snapshots of manufactured mechanisms showing static equilibrium for different poses.

3.5 DISCUSSION AND LIMITATIONS

In this study, we proposed an interactive optimization-driven approach to design a balanced mechanism. Our method calculates the number of springs required, their attachment points, and initial length in a semi-automated manner that integrates user input. Our null space search method is an efficient and effective tool for navigating the local space of design options. Our results suggest that our method is a flexible and powerful approach for creating new statically balanced mechanisms, but it has some limitations, which are described below.

Even with an exhaustive set of springs, there is no guarantee that a mechanism will admit to a statically balanced design. Nor does our method formally guarantee that the resulting number of springs is the minimum required to achieve static balance. Similarly, although our system converges to a locally optimal design, there may be remote solutions that cannot be reached during the null space search. While these limitations are inherent in non-convex optimization, our interactive system mitigates these problems by allowing the user to guide the optimization to new regions of the solution space as needed.

Although this study considers only planar mechanisms, it will be interesting to explore the design of statically balanced spatial mechanisms in the future. This setting brings new challenges, especially in terms of mechanical singularities and collisions between elements [58].

When manufacturing our example, we aimed to minimize friction at the joints. Minimizing friction means minimizing effort during transitions between configurations, but it may be interesting to allow a finite amount of friction in the design to increase the area of static balance.

Finally, our method does not support changes in the underlying mechanisms during optimization. Using changes such as degrees of freedom during optimization may allow for more flexibility and ultimately better designs.

Chapter 4

Design and Implementation of Programmable Drawing Automata based on Cam Mechanisms for Representing Spatial Trajectory

This paper presents the design and implementation of a preliminary version of a programmable drawing automaton (PDA-0) that draws user-specified three-dimensional trajectories. The automaton is strongly influenced by Jaquet Droz's 6,000 moving parts automaton from the 1770s. The authors developed a system that achieves programmability by encoding the trajectory specified by the user in the GUI as a shape on three cams. The author's automaton consists of a cam mechanism with a Revolute-Spherical-S-R (RSSR) linkage and three interchangeable cams, and utilizes inverse kinematics for the user-specified trajectory to calculate the geometry of the three cams. In this paper, we show how simple trajectories such as letters and symbols are drawn using automata and cams designed using our system.

4.1 Introduction

The animation of articulated characters plays an essential role in enhancing the viewer's experience without relying on a specific language. Sophisticated commercial computer graphics (CG) software has been advancing in movies and games for decades, allowing casual users to create rich motion sets of articulated characters. Meanwhile, in the physical world, Jaquet Droz invented a variety of automata in the 18th century, one of which was "The writer," which consisted of 6,000 parts; in the 19th century, a number of craftsmen created musical automata featuring the "Singing bird box"; and in the 19th century, a number of other musical automata were created by a number of craftsmen. These automata influenced the audio-animatronics developed by Disney in the 20th century. Although there is a long and rich history of experimenting with character animation in real objects, such as automata, creating mechanisms and automata that work like CG characters is still a difficult task today.

The problem is that it is difficult to predict the movement of an automaton when it is designed. Manufacturing and assembly of an automaton is generally a time-consuming process since it consists of many parts such as a linkage mechanism, and it takes a lot of time and money to check and correct the results of the movement, i.e., to repeat the trial and error process. This problem could be solved if the movement of the automaton could be predicted before production and if fine tuning could be done easily after production.

On the other hand, let's look at how to program the motion of an automaton using a cam mechanism like Jaquet Droz's automata. The cam mechanism used in this automata consists of two mechanical components called a disc cam and a cam follower. By touching the tips of the followers to the contours of the rotating cam disk, the angle of the followers can be moved periodically. This feature of the cam mechanism can be used to encode, or program, three-dimensional motion into multiple disc cam shapes. Furthermore, Jaquet Droz's automaton dynamically changes the motion of the automaton's arm for writing by introducing a mechanism that automatically changes a set of cam discs with different shapes (see Fig. 4.1).

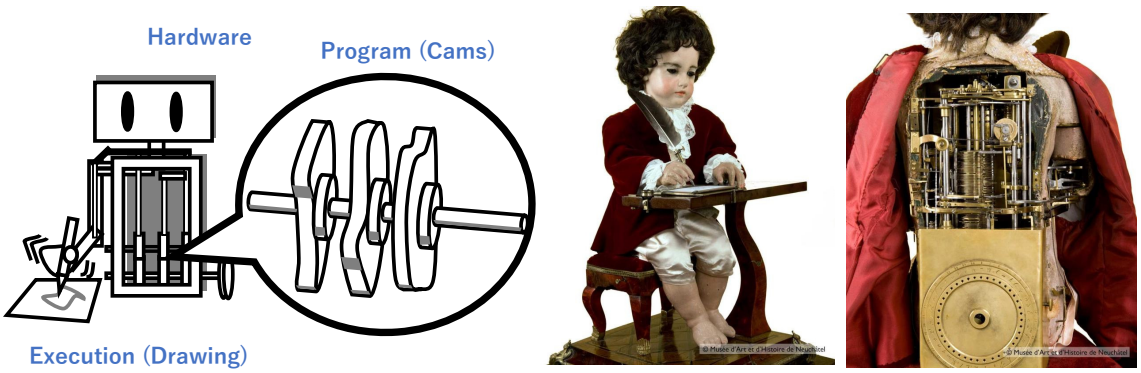


Figure 4.1: The concept of a drawing automaton based on cam and linkage mechanisms with three interchangeable cams. The pose of the arm linkage is controlled by three cams to trace a user-specified spatial trajectory.

Focusing on the programmability of the cam mechanism, the authors manufactured a programmable drawing automaton (PDA-0) and developed a system to interactively and automatically calculate the shape of the disc cams. PDA-0 consists of an RSSR (Revolute-Spherical-S-R) link and an interchangeable cam mechanism. In this paper, we present a method for calculating the geometry of the cam disk such that the automata can follow a user-specified spatial trajectory. Furthermore, we verify the validity of our method by actually manufacturing a cam disk using this system and verifying how PDA-0 reproduces the pre-defined motion.

4.2 Related work

For fabrication-oriented design, many studies have developed optimization-based methods which allow a user to design mechanical characters or automata [7]. Optimization targets can be classified mainly into three approaches: *trajectory*, *posture*, and *motion*.

Continuous endpoint movements of an articulated character can be traced. In fact, mechanical characters are fabricated by mechanical parts like gears and linkages. Some examples include mechanical automata from motion capture sequences [59], linkage based characters [5, 6, 49], compliant mechanism characters [11], and walking robots [60]. Complex movements, however, need more degrees of freedom, that is, more gears

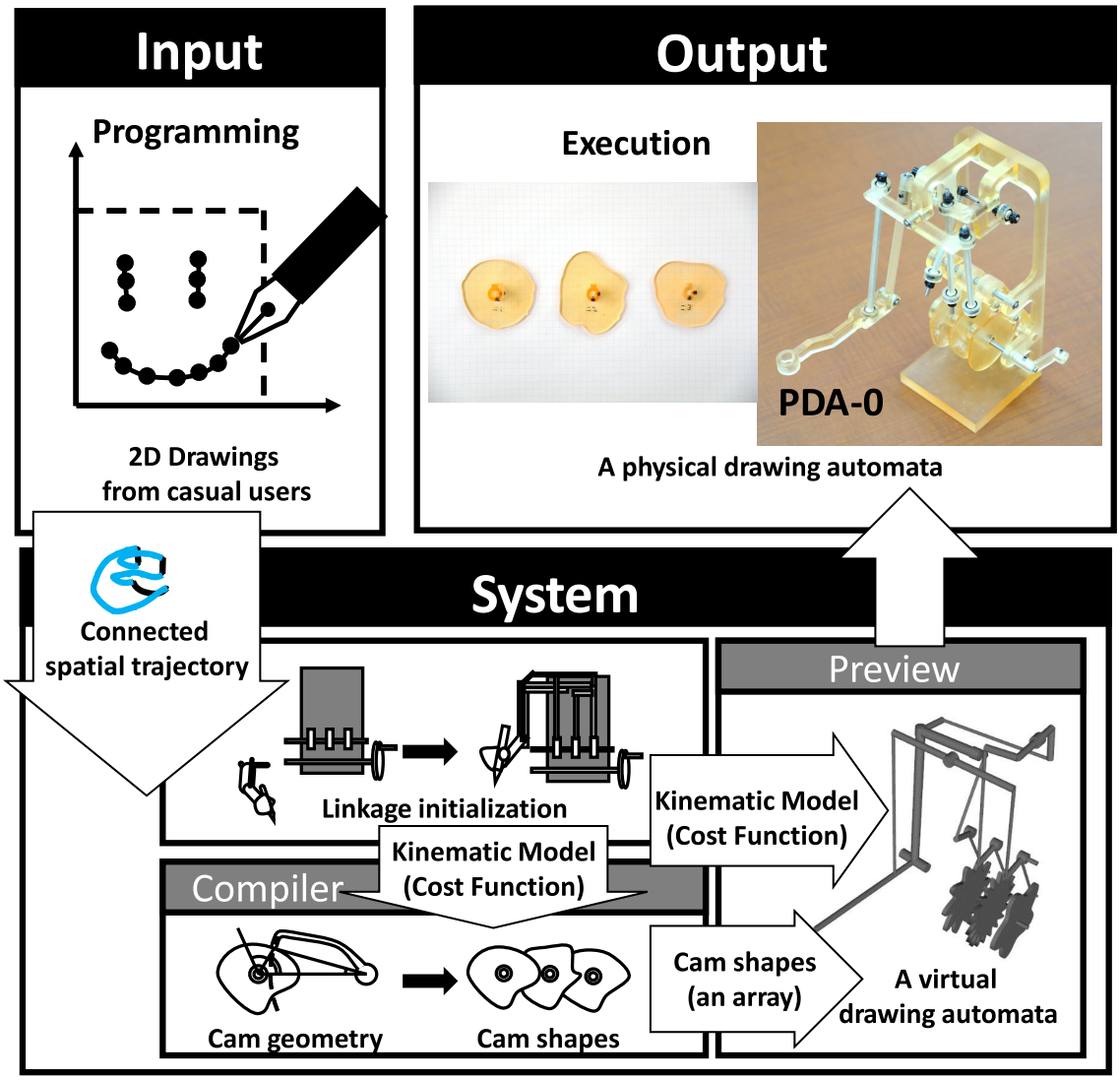


Figure 4.2: The flow of programming, compilation, and execution of the programmable drawing automaton, PDA-0.

and linkages.

Character animation can be expressed by detailed joint angles. One standard method is to use wires because the direction of wire tension can be bent to reduce the number of actuators. Some examples include flexible materials [10, 61, 62] and pulleys [50, 12]. Smooth movements, however, need more degrees of freedom, that is, more actuators.

Motion can be specified symbolically like "rotating continuously," "waving linearly," and "waving rationally." This approach is suitable for utilizing data structures to build mechanisms into the small character body such as printable automata [2], wind-up toys [63], and functional toys [64]. Non-predefined movements, however, may be difficult to trace symbolically.

Some other studies adopt either an exploratory design method [65] or mathematical functions [66]. However, the mechanical design for realizing a user-specified detailed three-dimensional (spatial) trajectory has not been implemented yet.

4.3 METHOD

As a research platform, we developed a programmable drawing automaton shown in Fig. 4.2. Its drawing process has three phases: *programming*, *compilation*, and *execution*. In a compilation phase, since the mechanical linkage may contain a closed loop, we used a constrained based simulation method like previous studies [5, 6] rather than a forward kinematics method. For programming, we developed a GUI so that the user can program a motion by specifying 2D trajectories via the GUI.

4.3.1 Cam Mechanism

A cam mechanism is one type of mechanism which can translate rotational motion to repetitive motions. It is consists of two parts: a disc cam and a cam follower. Since the endpoint of a cam follower always touches the curved surface of the opponent disc cam,

the angle of the cam follower oscillates according to the rotating curved surface of the disc cam.

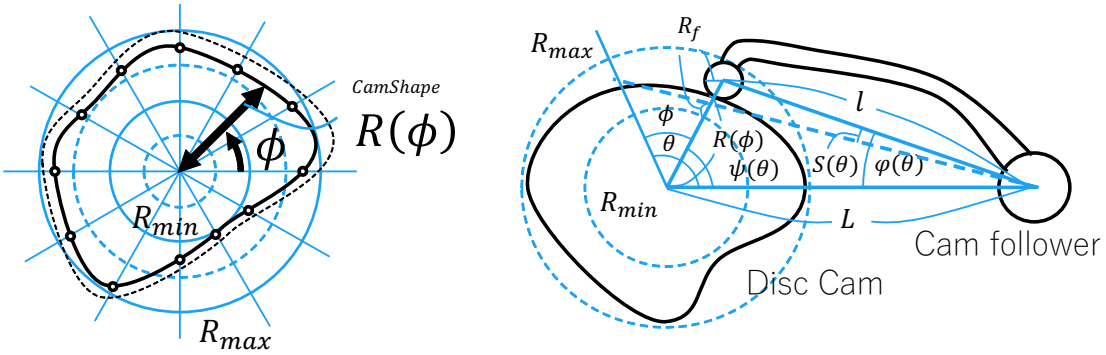


Figure 4.3: The design of the shape of a cam and the geometric relationship of the cam and cam mechanism in our system. L is the distance from the center of the disc, while l is the radius of the cam follower.

We formulate the shape of a disc cam in Fig. 4.3. A shape of the disc cam $R(\phi)$ is described by the radius function

$$R(\phi) = r(\phi)(R_{\max} - R_{\min}) + R_{\min} \tag{4.1}$$

where $r(\phi)$ is a normalized radius function which is linear interpolated from a n number of set $\mathscr{R}$ in the polar coordinate as

$$\mathscr{R} = (\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_n)^{\mathrm{T}} \tag{4.2}$$

$$\mathbf{r}_i = (r_i, \phi_i)^{\mathrm{T}}. \tag{4.3}$$

4.3.2 Trajectory and Motion Follower Program

To make the automaton draw desirable letters or symbols, we set a user-specified target path as a three-dimensional trajectory. A cyclic trajectory is a discrete function $\mathbf{X}(\theta) \in \mathbb{R}^3$ represented by a set of spatial positions

$$\mathscr{X} = (\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_N)^{\mathrm{T}} \tag{4.4}$$

$$\mathbf{X}_i = (\mathbf{x}_i, \theta_i)^{\mathrm{T}}. \tag{4.5}$$

A motion description called a motion program can be encoded into the shape of a disc cam as cyclic and continuous data [67]. A follower motion program is an angular function $S(\theta)$ which is represented by a sequence of angular data

$$\mathscr{S} = (\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n)^{\mathrm{T}} \tag{4.6}$$

$$\mathbf{s}_i = (s_i, \theta_i)^{\mathrm{T}} \tag{4.7}$$

The contact point of disc cam and cam follower makes a triangle with the axis position of disc cam and cam follower. When we set $\psi(\theta)$ and $\varphi(\theta)$ as interior angles of the triangle described in Fig. 4.3-b, we can derive the following equation about the follower motion program as follows:

$$S(\theta) = \varphi(\theta) - \varphi_{\min} \tag{4.8}$$

$$\varphi(\theta) = \arccos\left(\frac{l^2 + L^2 - (R(\phi(\theta)) + R_{\mathrm{f}})^2}{2lL}\right) \tag{4.9}$$

$$\varphi_{\min} = \arccos\left(\frac{l^2 + L^2 - (R_{\min} + R_{\mathrm{f}})^2}{2lL}\right) \tag{4.10}$$

$$\phi(\theta) = \theta - \psi(\theta) \tag{4.11}$$

$$\psi(\theta) = \arcsin\left(\frac{l}{R(\phi(\theta)) + R_f}\sin(\varphi(\theta))\right) \tag{4.12}$$

where R_{f} is the roller radius on the cam follower tip. To get equations above, we used the law of cosine $(R(\phi(\theta)) + R_f)^2 = l^2 + L^2 - 2lL\cos(\varphi(\theta))$ and the law of sine $(R(\phi(\theta)) + R_f)\sin(\psi(\theta)) = l\sin(\varphi(\theta))$.

4.3.3 Kinematics

In our system, linkages are mechanical structures composed of links **l** and joints **j**. Links are rigid bodies that translate cam follower angular movements to the endpoint of the automaton. We set a link **l** as

$$\mathbf{l}_i = (\mathbf{p}, \mathbf{R})^{\mathrm{T}} \tag{4.13}$$

where **p** is a position $\in \mathbb{R}^3$ and **R** is an orientation matrix. Only one of the links, the root link $\mathbf{a}_{\mathrm{root}}$, is provided to fix the position of the mechanism. The position and orientation of the root link is a constant.

The cam follower is a special link whose angle is defined by the shape of the cam and its rotation. The rotation of the cam follower is connected to the root link by a joint in the X-axis direction. In addition, because of the presence of three cam mechanisms, three cams and a cam follower are arranged in parallel in the X-axis direction.

Joints $\mathbf{j}$ have a role in determining the geometrical relationship between two links $\mathbf{l}_a$ and $\mathbf{l}_b$. There are two types of joints: revolute joints and distance joints. A revolute joint is a joint which restricts both the joint position and the axis of rotation. A distance joint is a joint which constrains the distance between the joint positions of a link. The expression of this type of joint is as follows

$$\mathbf{j}_{\text{distance}} = \left(\hat{\mathbf{p}}_{i_a,j_a}, \hat{\mathbf{p}}_{i_b,j_b}\right)^{\mathrm{T}} \tag{4.14}$$

$$\mathbf{j}_{\text{revolute}} = \left(\hat{\mathbf{p}}_{i_a,j_a}, \hat{\mathbf{p}}_{i_b,j_b}, \hat{\mathbf{v}}_{i_a,j_a}, \hat{\mathbf{v}}_{i_b,j_b}\right)^{\mathrm{T}}. \tag{4.15}$$

where $\hat{\mathbf{p}}_{i,j}$ and $\hat{\mathbf{v}}_{i,j}$ are j-th number of a position and rotation axis $\in \mathbb{R}^3$ of a i-th number of link $\mathbf{l}_i$ in a local coordinate.

The functions of the each joint are represented by the following expressions:

$$\mathbf{C}_{\text{distance}}\left(\mathbf{j}_{\text{distance}}\right) = \|\mathbf{p}_a - \mathbf{p}_b\| - d \tag{4.16}$$

$$\mathbf{C}_{\text{revolute}}\left(\mathbf{j}_{\text{revolute}}\right) = \left((\mathbf{p}_a - \mathbf{p}_b)^{\mathrm{T}}, (\mathbf{v}_a - \mathbf{v}_b)^{\mathrm{T}}\right)^{\mathrm{T}} \tag{4.17}$$

where d is a distance, $\mathbf{p}_a$ and $\mathbf{p}_b$ are positions, $\mathbf{v}_a$ and $\mathbf{v}_b$ are axes in a global coordinate calculated by

$$\mathbf{p}_k = \mathbf{R}\hat{\mathbf{p}}_{ik,jk} + \mathbf{p}_{ik} \tag{4.18}$$

$$\mathbf{v}_k = \mathbf{R}\hat{\mathbf{v}}_{ik,jk}. \tag{4.19}$$

When distance joints are actually manufactured, they are treated as a component "rod" or RSSR joint with both ends of the rod attached at the rod end. However, it should be noted that there is an angular limitation on the rod end when it is connected with a rod.

4.3.4 Disc Cam Shape Compilation

To get the disc cam shape functions $R(\theta)$ described as

$$R(\phi(\theta)) = \sqrt{l^2 + L^2 - 2lL\cos(S(\theta) + \varphi_{\min})} - R_f, \tag{4.20}$$

we need to compute the follower motion program $S(\theta)$ from a predefined trajectory $\mathbf{X}(\theta)$ using inverse kinematics. In the inverse kinematics calculation, we adopted constrained non-linear optimization problem manner [68]. We get a global position and the orientation of all links by minimizing the objective function:

$$f(\theta) = \frac{1}{2}\sum_i \mathbf{C}(\mathbf{j}_i)^{\mathrm{T}}\mathbf{C}(\mathbf{j}_i) + \frac{1}{2}\|\mathbf{X}(\theta) - \mathbf{p}_{\text{endpoint}}\|^2 \tag{4.21}$$

where $\mathbf{X}(\theta)$ is a target position at phase θ and $\mathbf{p}_{\text{endpoint}}$ is a current endpoint position in a global coordinate. For minimizing the function, we used the quasi-newton method with L-BFGS or BFGS [69, 70].

After determining link states at θ, we get the follower motion program $S(\theta)$ from the angular vector of the cam followers. By doing this calculation for all θ_i, we get a set of cam radii $\mathscr{R}$ for all cams, that is, the disc cam shape function $R(\theta)$.

4.3.5 Synthesis

In order to track a specified three-dimensional trajectory, a structure corresponding to the so-called arm is needed. The arm consists of three links: a shoulder link, an upper arm link, and a forearm link. The shoulder link is connected to the root link by a rotational joint on the x-axis rotation axis. The upper arm link is connected to the shoulder link at a slightly displaced position by a rotational joint on the Y-axis (gravity) rotation axis. The forearm link is connected to the upper arm link by a rotation joint on the X axis at a downward position by the length of the upper arm. The point extending forward from the forearm link is the point “end point” to be tracked.

However, the position cannot be determined because one free degree of freedom is left in each link that makes up the arm. For that remaining degree of freedom, the position is determined by connecting it to the links of the cam follower with a joint.

First, set all links to the initial position with the joints connected to each other. Next, select the corresponding cam follower links a_{follower} and arm links a_{arm}. For each link, we focus on its position and axis of rotation in a previously defined rotational joints. We define two planes $\mathbf{Z}_{\text{f}}$ and $\mathbf{Z}_{\text{a}}$ based on this positions $\mathbf{p}_{\text{f}}$ and $\mathbf{p}_{\text{a}}$, and axes $\mathbf{v}_{\text{f}}$ and $\mathbf{v}_{\text{a}}$ of rotations. These two planes can be classified as parallel or not parallel.

If the two planes are not parallel, focus on the line of intersection $\mathbf{L}(k) = \mathbf{p}_{\text{i}} + k\mathbf{v}_{\text{i}}, \forall k \in \mathbb{R}$ between the two planes, which is determined by the following equation.

$$\mathbf{p}_{\text{i}} = \mathbf{e}_{\text{f}}\langle \mathbf{e}_{\text{f}}\mathbf{t}_{\perp} \rangle + \mathbf{p}_{\text{f}} \tag{4.22}$$

$$\mathbf{v}_{\text{i}} = \frac{\mathbf{v}_{\text{f}} \times \mathbf{v}_{\text{a}}}{\|\mathbf{v}_{\text{f}} \times \mathbf{v}_{\text{a}}\|} \tag{4.23}$$

where $\mathbf{t}$, $\mathbf{t}_{\perp}$, $\mathbf{t}_{\|}$ are offsets between joints and $\mathbf{e}_{\text{f}}$, $\mathbf{e}_{\text{a}}$ are normalized perpendicular directions described as

$$\mathbf{t} = \mathbf{p}_{\text{a}} - \mathbf{p}_{\text{f}} \tag{4.24}$$

$$\mathbf{t}_{\perp} = \mathbf{t} - \mathbf{t}_{\|} \tag{4.25}$$

$$\mathbf{t}_{\|} = \mathbf{v}_{\text{i}}\langle \mathbf{v}_{\text{i}}, \mathbf{t} \rangle \tag{4.26}$$

$$\mathbf{e}_{\text{f}} = \frac{\mathbf{v}_{\text{f}} \times \mathbf{v}_{\text{i}}}{\|\mathbf{v}_{\text{f}} \times \mathbf{v}_{\text{i}}\|} \tag{4.27}$$

$$\mathbf{e}_{\text{a}} = \frac{\mathbf{v}_{\text{a}} \times \mathbf{v}_{\text{i}}}{\|\mathbf{v}_{\text{a}} \times \mathbf{v}_{\text{i}}\|}. \tag{4.28}$$

After finding the intersection line, create two points perpendicular to the line obtained from the above equation from the two joint positions. Set these two points as the ends of the distance joint and the distance between the two points as the distance value of the distance joint. If the joint is too long and causes collisions, it may be possible to solve this problem by creating a new relay link connected to the root link by a rotational joint on the intersection axis, and then repeating the above procedure.

If the two planes created are parallel to each other, the distance joint should be created so that the distance joint is a four-joint linkage mechanism, since it is almost the same as a two-dimensional linkage mechanism. If the linkage mechanism is in collision or other inconvenience, create a new relay link at a free position that is connected to the root link with a rotational joint and try the technique explained above.

By performing these tasks on all arm link and cam follower link pairs, the motion of the cam follower can be converted to the motion of the arm's endpoint.

4.3.6 Implementation

To get a desired spatial trajectory $\mathbf{X}(\theta)$, we developed an interface for casual users shown in Fig. 4.4. This interface reads multiple 2D trajectories $\mathscr{W} \in \mathbb{R}^2$ as input and translates them into a connected spatial trajectory as

$$\mathscr{W} = (\mathbf{W}_1, \mathbf{W}_2, ..., \mathbf{W}_n)^{\mathrm{T}}, \tag{4.29}$$

$$\mathbf{W}_i = (\mathbf{w}_i, \theta_i)^{\mathrm{T}}. \tag{4.30}$$

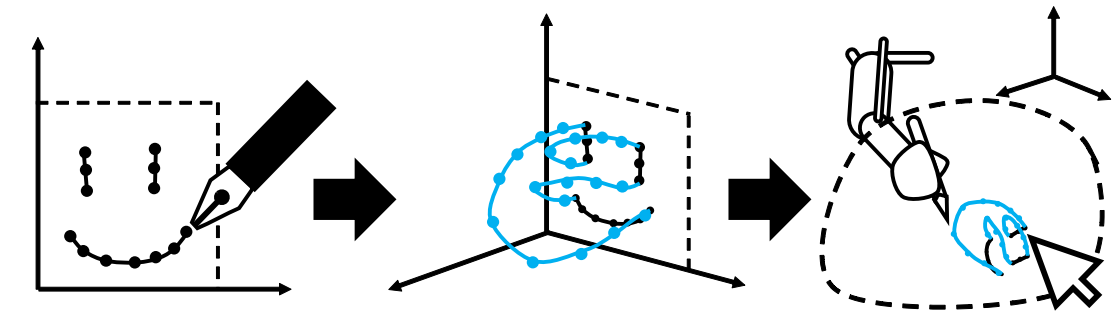


Figure 4.4: Scheme of GUI for programming. This interface reads multiple 2D trajectories as input and estimates its connected spatial trajectory.

We connect and interpolate both ends of all strokes to translate to connected spatial trajectory $\mathbf{x}$ using Bezier curves [71] that represent the motion of lifting the pen and moving it to the start of the next stroke.

We developed software which implemented our method in C++ shown as is shown in Fig. 4.5. This software has two areas: a preview area and a drawing input area. Users can draw using a mouse in the drawing input area to calculate a target trajectory. Once the trajectory is calculated, the optimal cam shapes are automatically estimated by our inverse kinematics method.

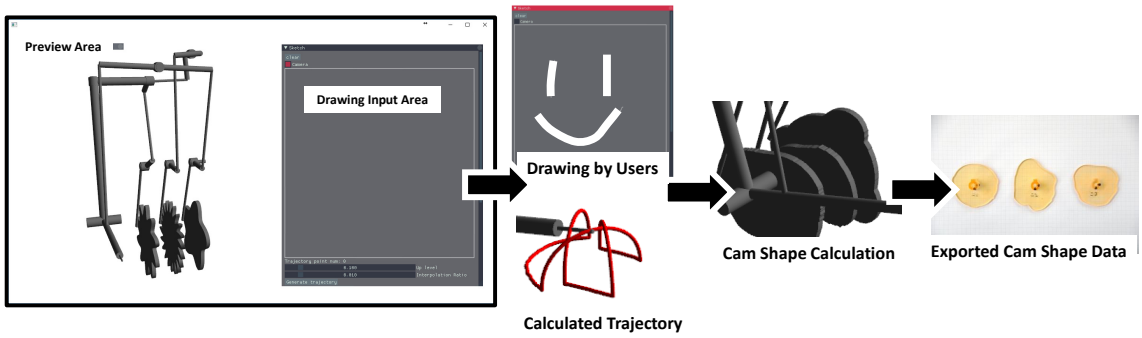


Figure 4.5: Cam shape generation software which inputs user drawings.

4.4 Fabrication

We fabricated the drawing automaton using a 3D printer, Keyence Agilista, with ABS resin. In our initial experience in fabricating all parts by the 3D printer, we had many problems. The three main problems in designing the mechanisms are summarized as follows:

1. *Reducing friction in joints.* Ineligible material friction property of ABS resin may cause high friction between 3D printed joints.

2. *Enhancing accuracy in joints.* An inaccurate 3D-printed joint may cause backlash due to the insufficient accuracy of the 3D printer.

3. *Suppressing deflection of structural objects.* A high stressed point in a 3D-printed object may cause deflection.

To cope with these three problems, we combined the 3D printer material with metal joint parts for fabricating PDA-0, shown in Fig. 4.6. In each revolute joint and spherical joint, we used 4 [mm] diameter metal shafts and bearings or spherical bearings. We also use 3 [mm] diameter metal shafts as rods in RSSR linkages. The size of the assembled automaton is 80 [mm] width, 80 [mm] depth, and 210 [mm] height. Our automaton consists of about 20 parts including three cams. Note that Jaquet Droz's drawing automata used 6,000 parts.

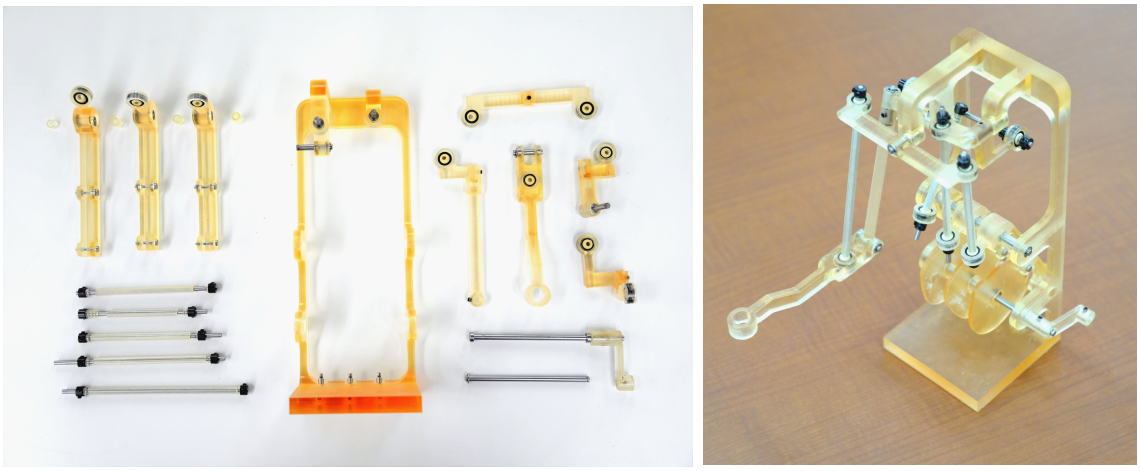


Figure 4.6: Fabricated PDA-0 using 3D printed and metal parts. Metal parts were used as bearings and spherical bearings.

4.5 Evaluation and Discussion

We conducted simple experiments shown in Fig. 4.7 to evaluate the performance of PDA-0 by executing three kinds of trajectories: circle, square, and characters shown in Figs. 4.8, 4.9, 4.10 and 4.11. Fig. 4.9 illustrates circle and square trajectories, the corresponding cams, and drawings. Each drawing is distorted inwards in the horizontal axis. Inaccuracy and backlash in the mechanisms may cause this distortion. Fig. 4.11 demonstrates that our automaton can draw user-specified trajectories in a sufficient accuracy to recognize a hiragana and a smile.

First, despite its simple structure of 16 components and three cams, PDA-0 shows its potential as a programmable drawing automaton: the three cams encode user-specified three-dimensional trajectories into cam shapes, the automaton executes the cam shapes as code, and the and can draw a specified trajectory.

Second, from a programming perspective, we have achieved visual programming for drawing trajectories. In this sense, our method surpasses Jaquet Droz's drawing automata, which used hard-encoded trajectories.

Of course, this method proved to have its limitations. The properties of the material and the rotational speed have a significant effect on the accuracy of the automaton's

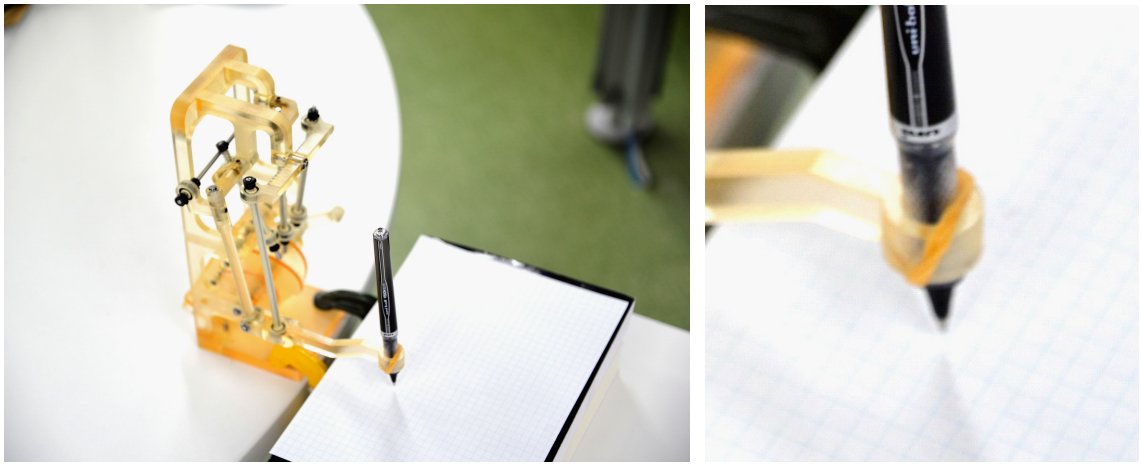


Figure 4.7: The situation of experiment with a programmable drawing automaton.

execution. Since many factors may affect the degree of effectiveness, it is necessary to identify the causes and improve the accuracy in the future.

In addition, the construction of the linking mechanism is not fully automated and still relies heavily on experience. The construction of a three-dimensional linking mechanism needs to be addressed because it is an interesting subject of research, including the singularity and collision problems.

4.6 Conclusion

In this study, we introduced a programmable drawing automaton, PDA-0, which consists of about 20 parts, and a method for calculating the shape of a cam to make the automaton trace a user-specified spatial trajectory. A simple experiment was performed to confirm the feasibility of the method. The results show a feature, programmability, that allows complex spatial motions to be changed by simply modifying or replacing the cam geometry.

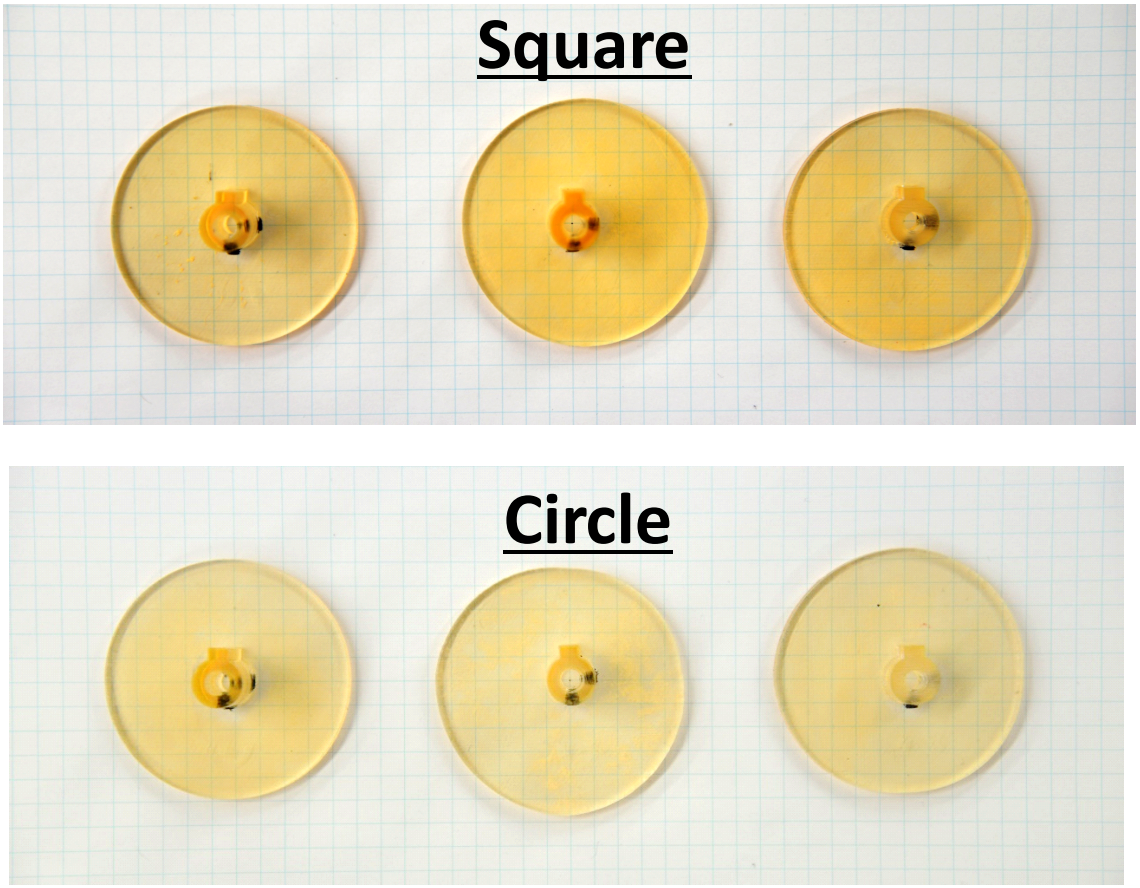


Figure 4.8: Results of letters and drawings. Top of the figure shows a letter of hiragana. Bottom of the figure shows a picture of a"smile". The left side of the graph shows the original target spatial trajectory.

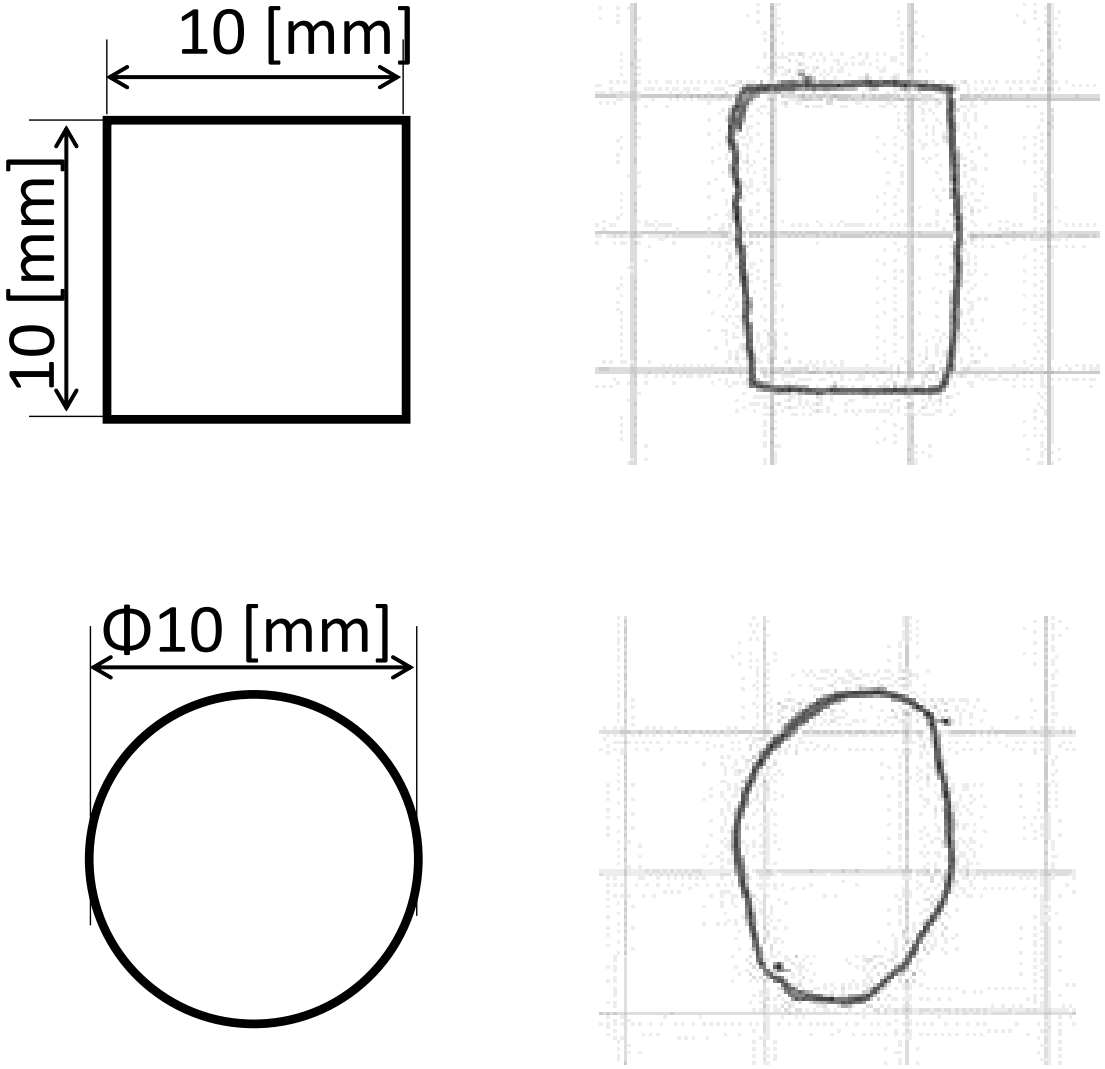


Figure 4.9: Results of simple shape drawings. Top of the figure shows the square shape and bottom of the figure shows circle shape.

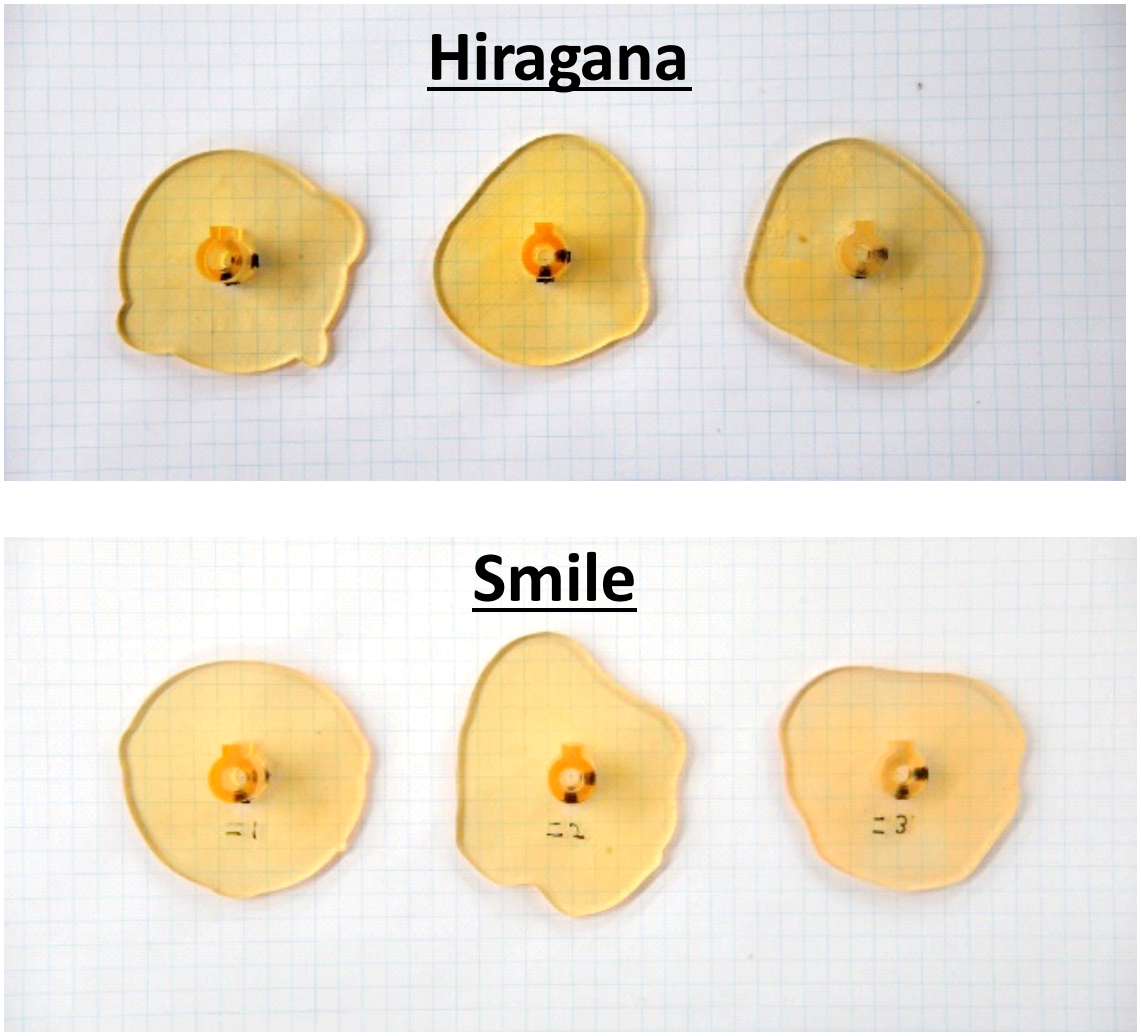


Figure 4.10: Results of letters and drawings. Top of the figure shows a letter of hiragana. Bottom of the figure shows a picture of a"smile". The left side of the graph shows the original target spatial trajectory.

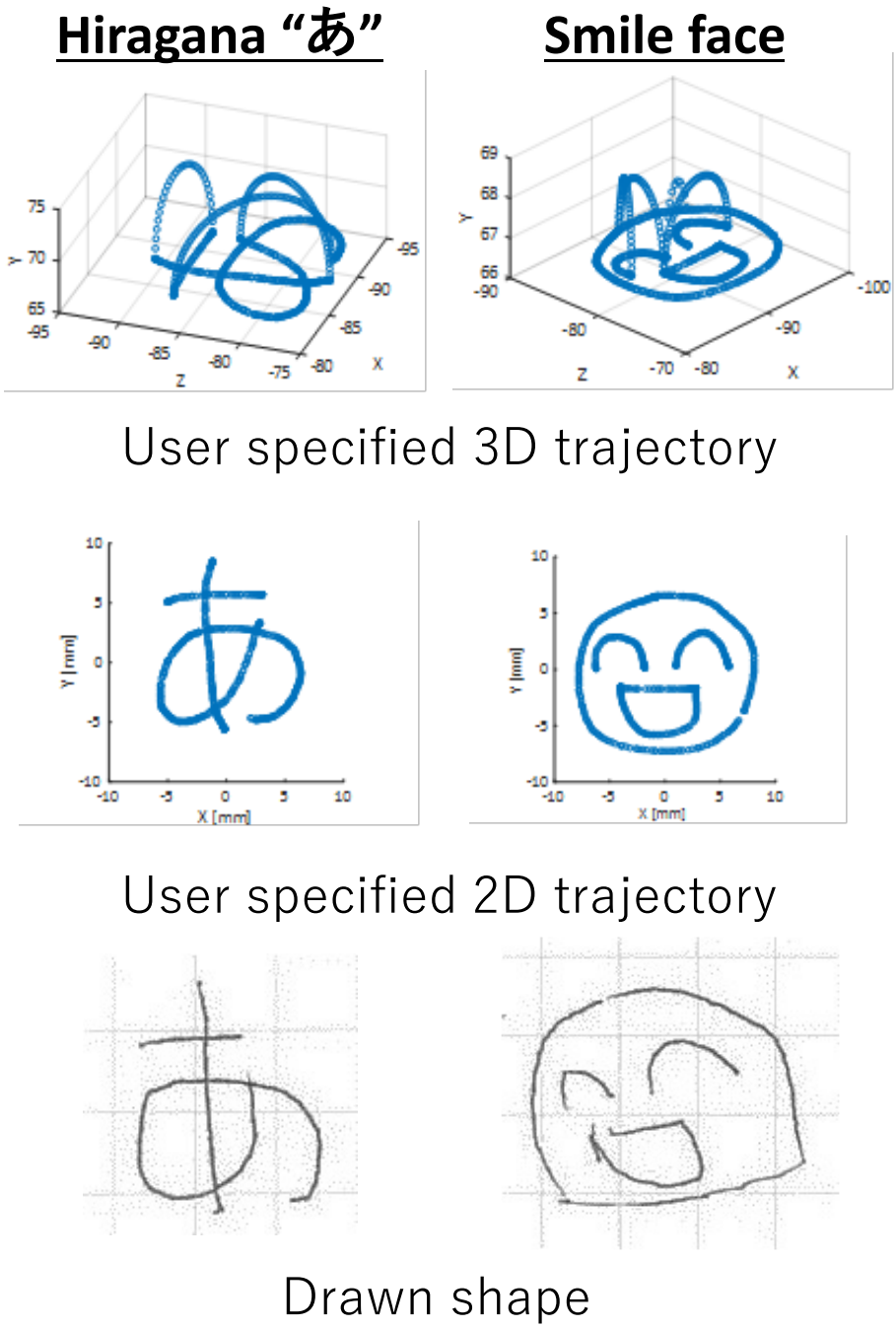


Figure 4.11: Results of letters and drawings. Top of the figure shows a letter of hiragana. Bottom of the figure shows a picture of a”smile”. The left side of the graph shows the original target spatial trajectory.

Chapter 5

Conclusion

5.1 Summary

In this thesis proposes three concepts to solve three main problems that have caused difficulties in the design process of a machine that is both functional and aesthetic, and introduces a concrete implementation. To address the first problem of selecting design parameters, a concept called “Constructability” is proposed, which allows a designer to move seamlessly from hand-drawn sketches to mechanism optimization. To deal with the second problem of fine tuning after manufacture, we propose the concept of “Programmability”, which allows us to adjust the function of the entire machine by replacing only certain parts of the machine. For the third problem, which is difficult to explore aesthetics while satisfying the constraints of the objective function, we propose “Explorability”, a search interface that allows the designer to interactively adjust parameters while satisfying the constraints of functionality.

In Chapter 2, we develop and propose a system to realize the concept of “Constructability”, which allows a designer to seamlessly transition from hand-drawn sketches to the optimization of a mechanism by dealing with counterweights. In this system, the shape of the counterweight and the linkage mechanism are generated by the designer who draws a picture and specifies the joints. After the designer enters the picture through the

interface, the picture is transformed into a polygonal shape. In addition, the designer draws a line on the polygons to specify the joint positions, which are then divided and generated into multiple links and joints. In addition, if the designer selects the link that corresponds to the root, the link mechanism is defined as information in a hierarchical structure. After setting the counterweight as a parameter, the objective function to adjust the position of the center of gravity around the joint is set considering the information of the hierarchical structure. The solution can be found using CMA-ES. The final mechanism is manufactured and assembled using a 3D printer and CNC. As a result, we have successfully manufactured several mechanisms that are stable and return to the original posture based on the geometry and joint structure freely specified by the designer.

In Chapter 3, we develop and propose a system to realize Explorability, a search interface that allows the designer to interactively adjust parameters while satisfying the constraints of functionality. We identify how many, where, and how much elasticity a spring should have to be attached to a complex linkage mechanism in order to establish a self-weight compensation mechanism. In addition, we propose a method of Null-Space Exploration, which allows the designer to interactively adjust the parameters of the self-weight compensation mechanism while keeping the mechanism as a constraint. First, the linkage mechanism is added to a collection of links and joints, and the parameters of the springs are defined. After specifying and calculating the link's range of motion, the potential energy (the sum of potential energy and elastic energy) of the entire link mechanism is calculated to determine how much it is distributed within the range of motion. By including the dispersion of the potential energy in the objective function of the optimization, the potential energy of the entire mechanism does not fluctuate within the range of motion, that is, the potential energy of the entire mechanism moves toward the state of the self-weight compensation mechanism. In addition, a GUI slider is presented on the screen that allows various parameters to be manipulated to adjust the designer's preference with the dead weight compensation mechanism as a constraint. When the designer specifies the direction of the parameter to be moved by the slider, the slider selects the one that is as close as possible to the direction the designer wants to go in the space composed of eigenvectors corresponding to the eigenvalue of the Hesse matrix of the objective function that is zero. By doing this on the interface, which is updated at 60 fps, we can pursue the aesthetics of the designer while keeping the value of the objective function at zero. As

a result, it was confirmed that we were able to design the system in accordance with the designer's aesthetics by using this method. In addition, it was confirmed that the number of springs to achieve the self-weight compensation mechanism and their connection relationships, which had been difficult to find, could be identified. In addition, some new self-weight compensating mechanisms were found by using this method, which had been difficult to find before.

In Chapter 4, we develop and propose a system to realize the concept of "Programmability", in which the function of the entire machine can be adjusted by replacing only specific parts of the machine, using a cam mechanism as the subject matter. The system converts the three-dimensional trajectory specified by the designer into a cam disk shape that can be tracked by the movement of the cam mechanism and linkage mechanism. In other words, it is possible to produce a programmable automaton that can draw programmable character changes. The programmable automata consists of a set of links and joints and a cam mechanism. A cam mechanism consists of two parts, a cam disk and a cam follower, and is characterized by the fact that the periodic motion of the cam follower, which moves in contact with it, can be programmed by changing the shape of the cam disk. Three cam mechanisms are arranged in parallel in the automata, and are connected to the three cam followers through a link mechanism that can be converted to a three-dimensional motion of the automata's arms. The designer inputs the three-dimensional trajectory that the automata should follow via the GUI's handwriting interface. The geometries of the three cam discs were determined by continuously calculating the angles that the cam followers should be at, using a joint that specifies the constraint on the positional relationship of the links as a cost function. The calculated results are manufactured by a 3D printer and assembled as an automaton. As a result, we have successfully manufactured an automaton that can follow any trajectory specified by the designer in real space. Furthermore, by replacing only the cam discs, we were able to see how the behavior was changed to track the trajectory specified separately by the designer.

The problems of parameter setting, fine adjustment after manufacturing, and difficulty in finding aesthetic properties were mitigated through the development of a design support method using the optimization technique, which took three approaches:

Constructability, Programmability, and Explorability. Research on technologies to support the design of aesthetically pleasing machines while ensuring functionality is new and there are still many issues that need to be solved.

5.2 Discussions and Limitations

In order to establish a new design method that balances the functionality and aesthetics of the machine, we have introduced the concrete implementation of the three concepts we have proposed and the manufactured prototypes, but of course there are some limitations and issues that need to be addressed.

5.2.1 Quantitative Evaluation Issues

We have not been able to quantitatively evaluate how well we have solved the time-consuming problem of designing a machine that is both functional and aesthetically pleasing. Specifically, we should evaluate how much better the design time and success rate is compared to existing software.

In order to make a quantitative evaluation of the proposed method, it would be necessary to build a new evaluation system. The system should be crowdsourced and ready to be distributed and experimented with by a large number of people. Participating subjects or users will be given a design task, and the design system will be able to measure the time spent on the design, the success rate, and the number of trials and errors. In addition, the system should be able to collect questionnaires to allow users to make a subjective assessment of the aesthetics of their designs. When we have achieved the development of a platform where these are possible, we will be able to make quantitative evaluations of the design interface.

5.2.2 Limited Scope of Application

In order to establish new methods to support the design of machines that are both functional and aesthetically pleasing, there is a need to move from a limited scope of application to a greater variety of situations. Typical examples are three-dimensional situations, collision problems, many mechanisms, material mechanical constraints, and designs that take into account existing components. In order to be able to deal with those situations, it will be necessary to adopt new approaches in addition to the optimization techniques presented in this paper.

Design Support Methods Using Machine Learning

Depending on the physical and mechanical properties of the design of a mechanism, there may be a tendency for inappropriate parameters. In order to avoid such inappropriate solutions, we can sample the space in advance and utilize the results during interactive design. It will be important to estimate the space of solutions in advance by machine learning to avoid falling into an incorrect value. For example, the walking automata design support method by Gaurav et al. achieves efficient search by changing the probability distribution in the space used for probability optimization using the EM algorithm.

The Problem of Collisions in Three Dimensional Space

The issue of collisions becomes a problem. In order to account for collisions of moving objects, all possible postures must be assumed to determine collisions (see Fig. 5.2). If collision is a problem, it will be necessary to re-calculate the geometry and reconfigure the configuration itself. In particular, solving the problem of collisions in three dimensions will be difficult due to the increase in the computational cost of mesh iterations and collisions, as well as the increase in solution space due to the increase in movement and rotation.

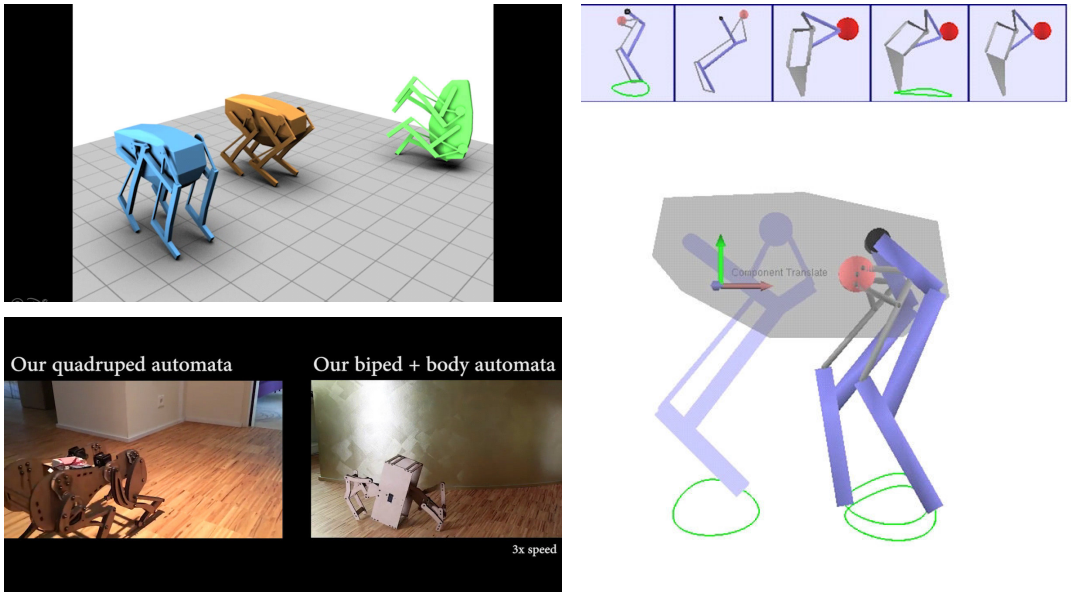


Figure 5.1: A study to find the parameters of the link mechanism that allows the automata to walk correctly. In this study, the parameters are pre-sampled to avoid using incorrect values, allowing for efficient parameter searching.

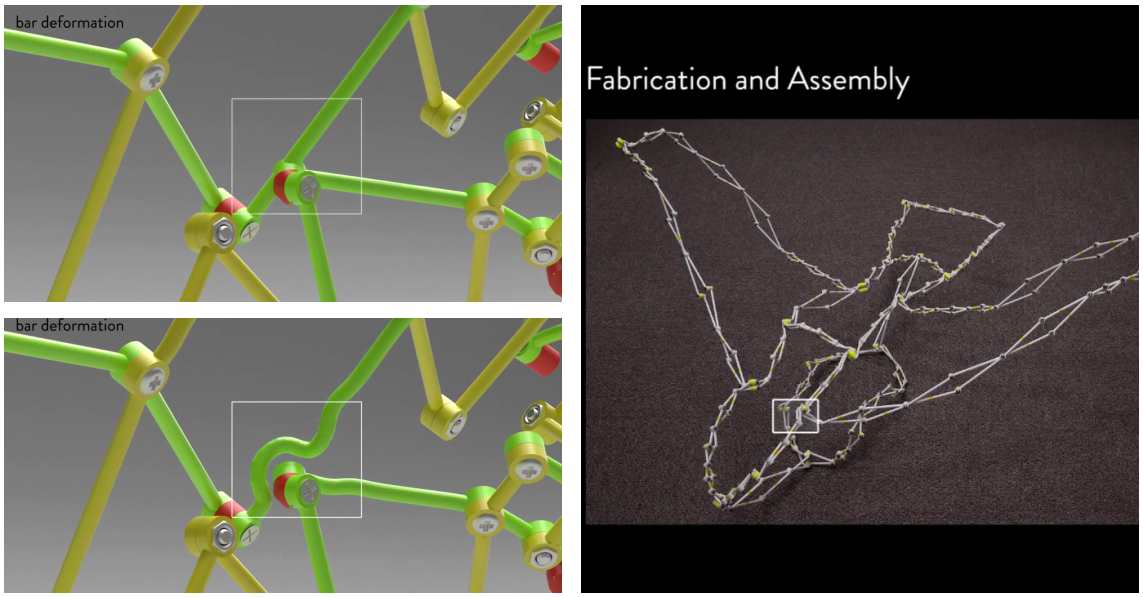


Figure 5.2: Research to generate a three-dimensional linking mechanism that can be folded into a user-specified shape. Addresses the problem of collisions between three-dimensional linking mechanisms.

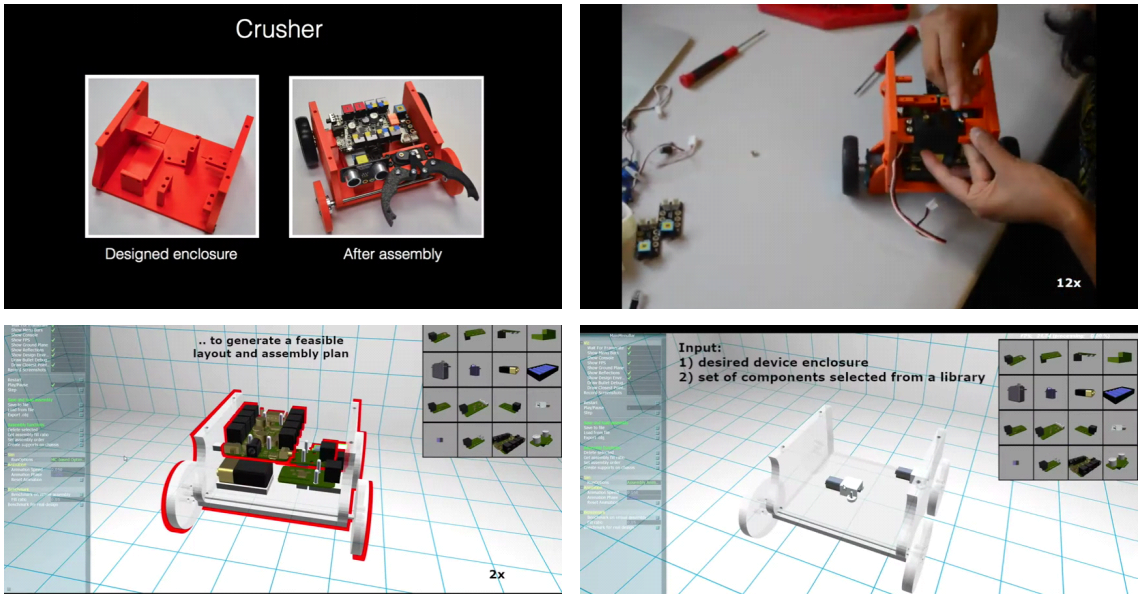


Figure 5.3: A study to generate a blueprint of the parts of a robot, such as the actuators and sensors that make up the robot, taking into account the possibility of assembly. The constraint of assemblability was one of the major factors that took time to design.

Designed for Use with Off-The-Shelf Products

A machine is basically composed of existing parts. It would be of help to the design novice if a system could be developed to determine which components are appropriate to use and take them into account in the design. Combinatorial optimization and machine learning are likely to support this technology.

Machine Movement Design

Another interesting topic is how to design an aesthetically correct motion for a motor or other actuator. There have been several studies on modifying a robot's movement to reproduce as much as possible the user-specified animation (see Fig. 5.4 and Fig. 5.5). Animators' pre-specified animations do not fundamentally follow physical laws, but rather have deformed movements to the extent that viewers can enjoy them. It is a challenging task to design movements that satisfy the animators within the physical and mechanical constraints of a robot. For example, it will be necessary to iteratively optimize the

mechanism itself in accordance with the expected animation.

From the point of view of stability, methods that can be adapted to more general mechanisms may also be necessary. The stability problem is considered to be that the energy function must be convex where it is stable. This can be traced back to the discussion of Hessian positive definiteness. This problem could be exploited in the future in an area called SDP.

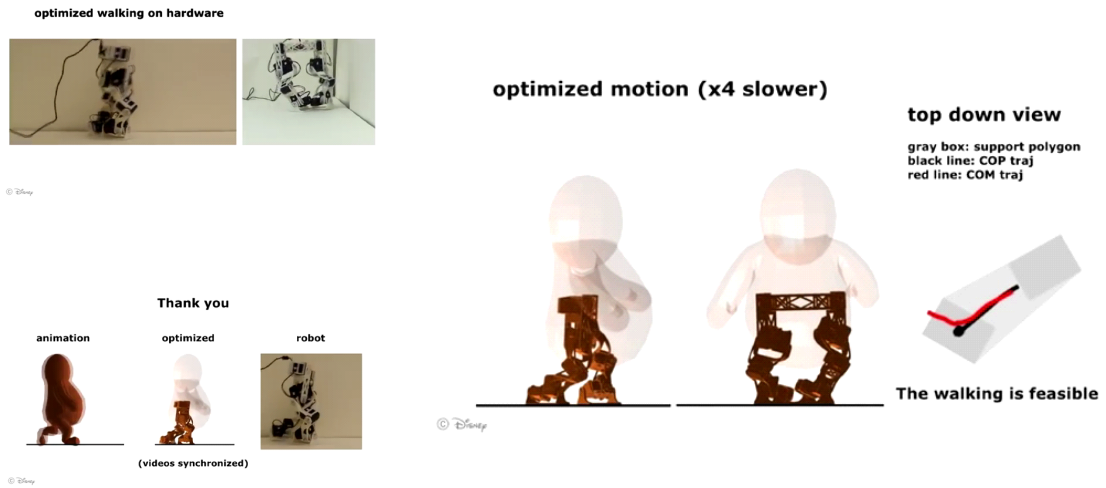


Figure 5.4: Research to achieve a robotic gait that would reproduce as closely as possible the gait animation and gait of the character specified by the artist.

Design that takes into account the Characteristics of the Material

It should also be noted that although in many cases objects have been treated as rigid bodies, in practice there is a risk of deformation and failure when the object is loaded. From the viewpoint of material mechanics and continuum mechanics, it will be necessary to predict the load beforehand by using the finite element method, etc. to maintain an appropriate shape of the design (see Fig. 5.6).

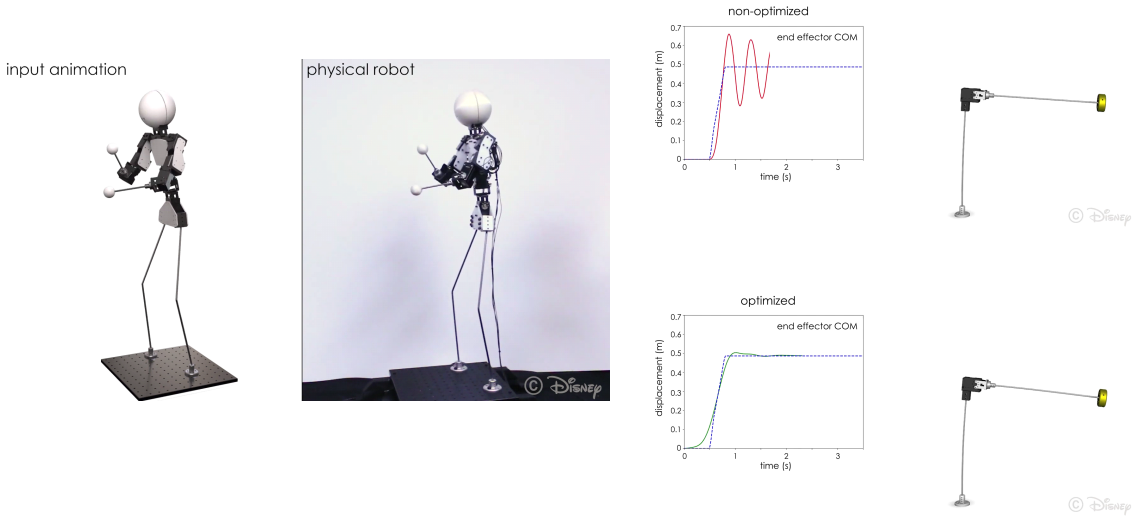


Figure 5.5: Research to control vibration problems caused by physical and material factors when trying to recreate character animation on a robot by pre-simulating and controlling the vibration.

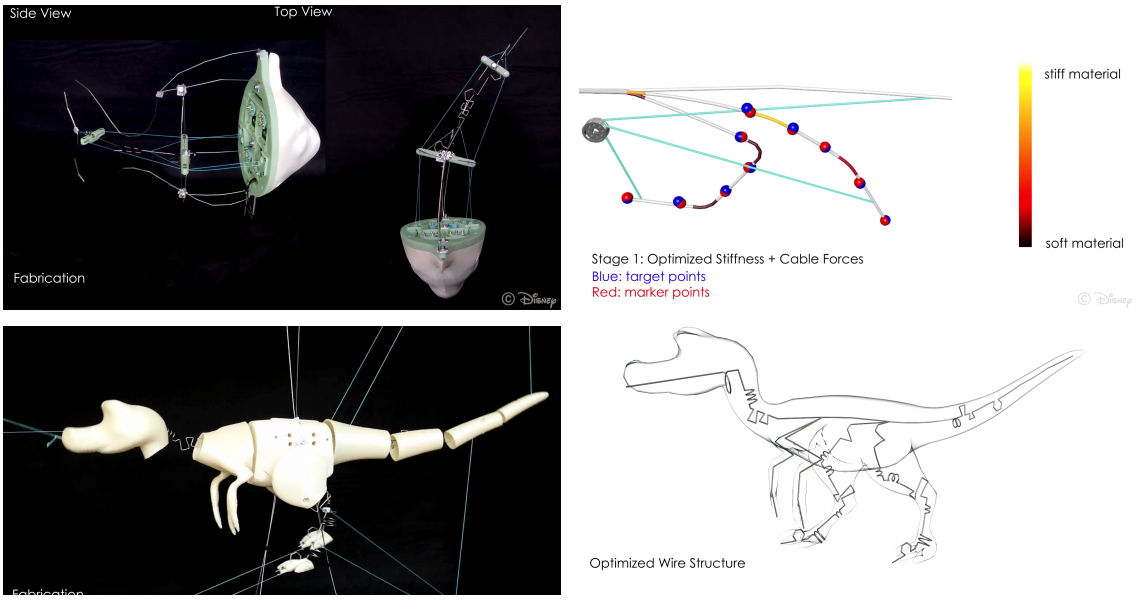


Figure 5.6: Kinetic wire mechanism

5.2.3 Aesthetic Predictability Issues

When questioning the aesthetics of a design, it will also be important to determine whether the aesthetics are universal or personal to a person. If the function of aesthetics has some universal tendency, the tendency can be learned beforehand and used as an initial value when searching for parameters based on personal aesthetics. Machine learning using neural networks could also be used to predict the initial values of aesthetically desirable parameters.

Another problem in using machine learning to predict aesthetic objective functions will be how to efficiently gather enough data for training; a study conducted by Yuki et al. successfully used crowdsourcing to efficiently find aesthetic functions (see Fig. 5.7). In addition, the reinforcement learning platform operated by OpenAI provides users with a python interface and a virtual environment to standardize evaluation methods for efficient development. The future challenge will be how to develop a platform used by a large number of users to adequately estimate the function on aesthetics.

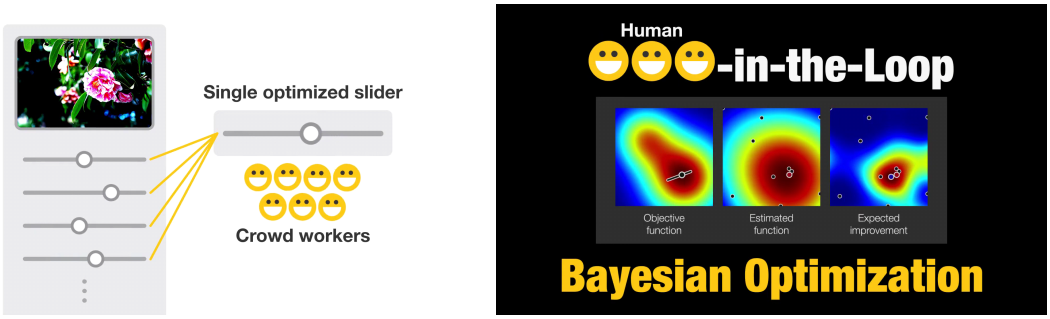


Figure 5.7: A study that uses Bayesian inference to estimate the aesthetics commonly preferred by humans by having crowdsourced workers search for aesthetically pleasing parameters, allowing for an efficient search for parameters.

5.3 Future Directions

In this thesis, we have conducted research to help people design robots and machines in an aesthetically pleasing way, but there is still work to be done. As the number of

people working on these studies increases and the technology matures, it will be possible for the software to be made available to the general public. If this software becomes widespread, tasks that were previously only possible for people with special skills will become creative activities that anyone who wants to express themselves can participate in. This corresponds to the so-called “democratization” activities of special skills that have brought about new innovations in the past. (e.g. computer graphics and PCs, electronics and Arduino, manufacturing and 3D printing technology, 3D games and Unity, etc.) If this democratization is achieved even in the production of beautiful robots and machines, a new creative culture never seen before will emerge. It is hoped that the research in this thesis on supporting design with optimization techniques will help us to move towards such a creative future.

Bibliography

[1] Y. Koyama, "Computational design driven by aesthetic preference," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16 Adjunct, (New York, NY, USA), p. 1–4, Association for Computing Machinery, 2016.

[2] L. Zhu, W. Xu, J. Snyder, Y. Liu, G. Wang, and B. Guo, "Motion-Guided Mechanical Toy Modeling," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 127–1, 2012.

[3] V. Megaro, B. Thomaszewski, D. Gauge, E. Grinspun, S. Coros, and M. Gross, "ChaCra: An Interactive Design System for Rapid Character Crafting," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '14, (Goslar, DEU), p. 123–130, Eurographics Association, 2015.

[4] G. Nishida, A. Bousseau, and D. G. Aliaga, "Multi-Pose Interactive Linkage Design," *Computer Graphics Forum*, vol. 38, no. 2, pp. 277–289, 2019.

[5] S. Coros, B. Thomaszewski, G. Noris, S. Sueda, M. Forberg, R. W. Sumner, W. Matusik, and B. Bickel, "Computational Design of Mechanical Characters," *ACM Trans. Graph.*, vol. 32, no. 4.

[6] B. Thomaszewski, S. Coros, D. Gauge, V. Megaro, E. Grinspun, and M. Gross, "Computational Design of Linkage-based Characters," *ACM Trans. Graph.*, vol. 33, pp. 64:1–64:9, July 2014.

[7] A. H. Bermano, T. Funkhouser, and S. Rusinkiewicz, "State of the Art in Methods and Representations for Fabrication-Aware Design," *Comput. Graph. Forum*, vol. 36, pp. 509–535, May 2017.

[8] C. Hecker, B. Raabe, R. W. Enslow, J. DeWeese, J. Maynard, and K. van Prooijen, "Real-time Motion Retargeting to Highly Varied User-created Morphologies," *ACM Trans. Graph.*, vol. 27, pp. 27:1–27:11, Aug. 2008.

[9] M. Bächer, B. Bickel, D. L. James, and H. Pfister, "Fabricating Articulated Characters from Skinned Meshes," *ACM Trans. Graph.*, vol. 31, July 2012.

[10] M. Skouras, B. Thomaszewski, S. Coros, B. Bickel, and M. Gross, "Computational Design of Actuated Deformable Characters," *ACM Trans. Graph.*, vol. 32, pp. 82:1–82:10, July 2013.

[11] V. Megaro, J. Zehnder, M. Bächer, S. Coros, M. Gross, and B. Thomaszewski, "A Computational Design tool for Compliant Mechanisms," *ACM Trans. Graph.*, vol. 36, pp. 82:1–82:12, July 2017.

[12] J. Li, S. Andrews, K. G. Birkas, and P. G. Kry, "Task-based Design of Cable-driven Articulated Mechanisms," in *Proc. of the 1st Annual ACM Symposium on Computational Fabrication (SCF '17)*, pp. 6:1–6:12, ACM, 2017.

[13] H. Xu, E. Knoop, S. Coros, and M. Bächer, "Bend-It: Design and Fabrication of Kinetic Wire Characters," *ACM Trans. Graph.*, vol. 37, Dec. 2018.

[14] D. Wu, J. Terpenny, and D. Schaefer, "Digital design and manufacturing on the cloud: A review of software and services," *AI EDAM*, vol. 31, no. 1, pp. 104–118, 2017.

[15] Y. Yokota, "A Historical Overview of Japanese Clocks and Karakuri," in *International Symposium on History of Machines and Mechanisms*, pp. 175–188, Springer, 2009.

[16] Y. R. Chheta, R. M. Joshi, K. K. Gotewal, and M. ManoahStephen, "A review on passive gravity compensation," in *Proc. of ICECA 2017*, pp. 184–189, April 2017.

[17] C. Yu, Z. Li, and H. Liu, "Research on Gravity Compensation of Robot Arm Based on Model Learning*," in *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 635–641, July 2019.

[18] H. Zhao, C. Hong, J. Lin, X. Jin, and W. Xu, "Make it swing: Fabricating personalized roly-poly toys," *Computer Aided Geometric Design*, vol. 43, pp. 226 – 236, 2016. Geometric Modeling and Processing 2016.

[19] Y. Mori and T. Igarashi, "Plushie: An Interactive Design System for Plush Toys," *ACM Trans. Graph.*, vol. 26, p. 45–es, July 2007.

[20] M. Bächer, B. Bickel, D. L. James, and H. Pfister, "Fabricating Articulated Characters from Skinned Meshes," *ACM Trans. Graph.*, vol. 31, pp. 47:1–47:9, July 2012.

[21] S. Coros, B. Thomaszewski, G. Noris, S. Sueda, M. Forberg, R. W. Sumner, W. Matusik, and B. Bickel, "Computational Design of Mechanical Characters," *ACM Trans. Graph.*, vol. 32, July 2013.

[22] B. Thomaszewski, S. Coros, D. Gauge, V. Megaro, E. Grinspun, and M. Gross, "Computational Design of Linkage-Based Characters," *ACM Trans. Graph.*, vol. 33, July 2014.

[23] C. A. Coello Coello, A. D. Christiansen, and A. H. Aguirre, "Multiobjective design optimization of counterweight balancing of a robot arm using genetic algorithms," in *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, pp. 20–23, Nov 1995.

[24] J. Wu, T. Li, and L. Wang, "Counterweight optimization of an asymmetrical hybrid machine tool based on dynamic isotropy," *Journal of Mechanical Science and Technology*, vol. 27, 07 2013.

[25] Y. Li, J. Wang, X.-J. Liu, and L.-P. Wang, "Dynamic performance comparison and counterweight optimization of two 3-DOF parallel manipulators for a new hybrid machine tool," *Mechanism and Machine Theory*, vol. 45, pp. 1668–1680, 11 2010.

[26] T. Takahashi, J. Zehnder, H. G. Okuno, S. Sugano, S. Coros, and B. Thomaszewski, "Computational Design of Statically Balanced Planar Spring Mechanisms," *IEEE Robotics and Automation Letters*, vol. 4, pp. 4438–4444, Oct 2019.

[27] R. Prévost, E. Whiting, S. Lefebvre, and O. Sorkine-Hornung, "Make It Stand: Balancing Shapes for 3D Fabrication," *ACM Trans. Graph.*, vol. 32, July 2013.

[28] M. Bächer, B. Bickel, E. Whiting, and O. Sorkine-Hornung, "Spin-It: Optimizing Moment of Inertia for Spinnable Objects," *Commun. ACM*, vol. 60, p. 92–99, July 2017.

[29] R. Prévost, M. Bächer, W. Jarosz, and O. Sorkine-Hornung, "Balancing 3D Models with Movable Masses," in *Proceedings of the Conference on Vision, Modeling and Visualization*, VMV '16, (Goslar, DEU), p. 9–16, Eurographics Association, 2016.

[30] N. Hansen, "The CMA evolution strategy: a comparing review," *Towards a new evolutionary computation*, pp. 75–102, 2006.

[31] N. Hansen, "The CMA evolution strategy: A tutorial," *arXiv preprint arXiv:1604.00772*, 2016.

[32] M. Tamis, "Comparison between Projected Gauss Seidel and Sequential Impulse Solvers for Real-Time Physics Simulations," 2015.

[33] D. H. Douglas and T. K. Peucker, "Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature," *The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.

[34] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer graphics and image processing*, vol. 1, no. 3, pp. 244–256, 1972.

[35] J. Hergel and S. Lefebvre, "3D fabrication of 2D mechanisms," *Computer Graphics Forum*, vol. 34, no. 2, pp. 229–238, 2015.

[36] J. Lasseter, "Principles of traditional animation applied to 3D computer animation," *ACM Siggraph Comp. Graph.*, vol. 21, no. 4, pp. 35–44, 1987.

[37] M. French and M. Widden, "The spring-and-lever balancing mechanism, George Carwardine and the Anglepoise lamp," *Journal of Mechanical Engineering Science*, vol. 214, no. 3, pp. 501–508, 2000.

[38] J. L. Herder, *Energy-free Systems. Theory, conception and design of statically*. PhD thesis, Delft University of Technology, 2001.

[39] J. A. Gallego and J. L. Herder, "Criteria for the static balancing of compliant mechanisms," in *ASME 2010 IDETC/CIE*, pp. 465–473, American Society of Mechanical Engineers, 2010.

[40] G. Radaelli, J. A. Gallego, and J. L. Herder, "An energy approach to static balancing of systems with torsion stiffness," *Journal of Mechanical Design*, vol. 133, no. 9, p. 091006, 2011.

[41] G. Radaelli and J. L. Herder, "Isogeometric shape optimization for compliant mechanisms with prescribed load paths," in *ASME 2014 IDETC/CIE*, 2014.

[42] T. Morita, F. Kuribara, Y. Shiozawa, and S. Sugano, "A novel mechanism design for gravity compensation in three dimensional space," in *Proc. of IEEE/ASME AIM*, pp. 163–168, 2003.

[43] J. P. Whitney and J. K. Hodgins, "A passively safe and gravity-counterbalanced anthropomorphic robot arm," in *Proc. of IEEE ICRA*, pp. 6168–6173, 2014.

[44] V. Arakelian, "Gravity compensation in robotics," *Advanced Robotics*, vol. 30, no. 2, pp. 79–96, 2016.

[45] Y. Chen, J. Fan, Y. Zhu, J. Zhao, and H. Cai, "A passively safe cable driven upper limb rehabilitation exoskeleton," *Technology and Health Care*, vol. 23, no. s2, pp. S197–S202, 2015.

[46] H.-C. Hsieh, L. Chien, and C.-C. Lan, "Mechanical design of a gravity-balancing wearable exoskeleton for the motion enhancement of human upper limb," in *Proc. of IEEE ICRA '15*, pp. 4992–4997, IEEE, 2015.

[47] P. Robertson, C.-H. Kuo, and J. Herder, "A Compatibility Study of Static Balancing in Reconfigurable Mechanisms," in *ASME IDETC/CIE '16*, 2016.

[48] A. Garg, A. Jacobson, and E. Grinspun, "Computational Design of Reconfigurables," *ACM Trans. Graph.*, vol. 35, no. 4, 2016.

[49] M. Bächer, S. Coros, and B. Thomaszewski, "LinkEdit: Interactive Linkage Editing Using Symbolic Kinematics," *ACM Trans. Graph.*, vol. 34, pp. 99:1–99:8, July 2015.

[50] V. Megaro, E. Knoop, A. Spielberg, D. I. W. Levin, W. Matusik, M. Gross, B. Thomaszewski, and M. Bächer, "Designing Cable-driven Actuation Networks for Kinematic Chains and Trees," in *Proc. of the ACM SIGGRAPH / Eurographics SCA '17*, 2017.

[51] S. Ha, S. Coros, A. Alspach, J. Kim, and K. Yamane, "Computational Co-Optimization of Design Parameters and Motion Trajectories for Robotic Systems," *The International Journal of Robotics Research*, no. 0278364918771172, 2018.

[52] M. Lustig, A. Dunning, and J. Herder, "Parameter analysis for the design of statically balanced serial linkages using a stiffness matrix approach with cartesian coordinates," in *14th IFToMM world congress*, pp. 122–129, 2015.

[53] S. R. Deepak and G. Ananthasuresh, "Perfect static balance of linkages by addition of springs but not auxiliary bodies," *Journal of mechanisms and robotics*, vol. 4, no. 2, p. 021014, 2012.

[54] P. Lin, W. Shieh, and D. Chen, "Design of Statically Balanced Planar Articulated Manipulators With Spring Suspension," *IEEE Transactions on Robotics*, vol. 28, pp. 12–21, Feb 2012.

[55] Y.-Y. Lee and D.-Z. Chen, "Determination of spring installation configuration on statically balanced planar articulated manipulators," *Mechanism and Machine Theory*, vol. 74, pp. 319 – 336, 2014.

[56] E. G. Merriam, M. Colton, S. Magleby, and L. L. Howell, "The design of a fully compliant statically balanced mechanism," in *ASME IDETC/CIE '13*, 2013.

[57] G. Radaelli and J. L. Herder, "Shape optimization and sensitivity of compliant beams for prescribed load-displacement response," *Mechanical Sciences*, vol. 7, no. 2, pp. 219–232, 2016.

[58] C. Gosselin, "Gravity compensation, static balancing and dynamic balancing of parallel mechanisms," in *Smart Devices and Machines for Advanced Manufacturing*, pp. 27–48, Springer, 2008.

[59] D. Ceylan, W. Li, N. J. Mitra, M. Agrawala, and M. Pauly, "Designing and Fabricating Mechanical Automata from Mocap Sequences," *ACM Trans. Graph.*, vol. 32, pp. 186:1–186:11, Nov. 2013.

[60] G. Bharaj, S. Coros, B. Thomaszewski, J. Tompkin, B. Bickel, and H. Pfister, "Computational Design of Walking Automata," in *Proc. of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA '15)*, pp. 93–100, ACM, 2015.

[61] J. M. Bern, K.-H. Chang, and S. Coros, "Interactive Design of Animated Plushies," *ACM Trans. Graph.*, vol. 36, pp. 80:1–80:11, July 2017.

[62] J. M. Bern, G. Kumagai, and S. Coros, "Fabrication, modeling, and control of plush robots," in *Proc. of 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2017)*, pp. 3739–3746, IEEE & RSJ, Sept 2017.

[63] P. Song, X. Wang, X. Tang, C.-W. Fu, H. Xu, L. Liu, and N. J. Mitra, "Computational Design of Wind-up Toys," *ACM Trans. Graph.*, vol. 36, pp. 238:1–238:13, Nov. 2017.

[64] R. Zhang, T. Auzinger, D. Ceylan, W. Li, and B. Bickel, "Functionality-aware Retargeting of Mechanisms to 3D Shapes," *ACM Trans. Graph.*, vol. 36, pp. 81:1–81:13, July 2017.

[65] R. Roussel, M.-P. Cani, J.-C. Léon, and N. J. Mitra, "Exploratory design of mechanical devices with motion constraints," *Computers & Graphics*, vol. 74, pp. 244 – 256, 2018.

[66] Y. Liu and J. M. McCarthy, "Design of mechanisms to draw trigonometric plane curves," *Journal of Mechanisms and Robotics*, vol. 9, no. 2, p. 024503, 2017.

[67] W.-T. Chang and L.-I. Wu, "Mechanical error analysis of disk cam mechanisms with a flat-faced follower," *Journal of Mechanical Science and Technology*, vol. 20, pp. 345–357, Mar 2006.

[68] J. Zhao and N. I. Badler, "Inverse Kinematics Positioning Using Nonlinear Programming for Highly Articulated Figures," *ACM Trans. Graph.*, vol. 13, pp. 313–336, Oct. 1994.

[69] C. G. BROYDEN, “The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations,” *IMA Journal of Applied Mathematics*, vol. 6, no. 1, pp. 76–90, 1970.

[70] R. Fletcher, “A new approach to variable metric algorithms,” *The Computer Journal*, vol. 13, no. 3, pp. 317–322, 1970.

[71] D. F. Rogers and J. A. Adams, *Mathematical elements for computer graphics*. McGraw-Hill Higher Education, 1989.

Research achievements

種 類 別 (By Type)	題名、　発表・発行掲載誌名、　発表・発行年月、　連名者（申請者含む）(theme, journal name, date & year of publication, name of authors inc. yourself)
ジャーナル（査読あり）	
○	[1] T. Takahashi, J. Zehnder, H. G. Okuno, S. Sugano, S. Coros, B. Thomaszewski: Computational Design of Statically Balanced Planar Spring Mechanisms. 2019 IEEE Robotics And Automation Letters (RA-L), pp. 4438 – 4444. Vol.4, Issue 4, 2019.
	[2] S. Suzuki, T. Takahashi, H. G. Okuno: Development of a Robotic Pet Using Sound Source Localization with the HARK Robot Audition System, Journal of Robotics and Mechatronics Vol.29 No.1, 2017
国際学会（査読あり）	
○	[3] T. Takahashi, H. G. Okuno, S. Sugano, S. Coros, B. Thomaszewski: Computational Design of Balanced Open Link Planar Mechanisms with Counterweights from User Sketches. 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (Oral Presentation with Review)
○	[4] T. Takahashi, J. Zehnder, H. G. Okuno, S. Sugano, S. Coros, B. Thomaszewski: Computational Design of Statically Balanced Planar Spring Mechanisms. 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 450-455. 2017 (Oral Presentation with Review)
○	[5] T. Takahashi, H. G. Okuno: Design and Implementation of Programmable Drawing Automata based on Cam Mechanisms for Representing Spatial Trajectory. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 450-455. 2017 (Oral Presentation with Review)
○	[6] T. Takahashi, T. Soma, Y. Miwa, N. Hiroko: Design of hand contact improvisation interface supporting co-creative embodied expression, Human-Computer Interaction International 2017 (HCII2017), HIMI 2017, Part II, LNCS 10273, pp. 631-639, 2017.7 (Oral Presentation with Review)
○	[7] T. Takahashi, R. Hayashi, Y. Miwa, N. Hiroko: Co-creative Expression Interface : Aiming to Support Embodied Communication for Developmentally Disabled Children, Human-Computer Interaction International 2016 (HCII2016), HIMI 2016, Part II, LNCS 9735, pp.346-356, 2016.7 (Oral Presentation with Review)
	[8] Y. Kajita, T. Takahashi, Y. Miwa, S. Itai: Designing the Embodied Shadow Media Using Virtual Three-Dimensional Space, Human-Computer Interaction International 2015 (HCII2015), HIMI 2015, Part II, LNCS 9173, pp.610-621, 2015.8 (Oral Presentation with Review)

種 類 別 By Type	題名、　発表・発行掲載誌名、　発表・発行年月、　連名者（申請者含む）(theme, journal name, date & year of publication, name of authors inc. yourself)
国内学会（査読なし）	[9] Takuto TAKAHASHI, Masanori TSURUTA, Yoshiyuki MIWA, Hiroko NISHI: Measurement co-creative bodily expressions by hand contact interfaces, LIFE2016, 2016.9 (Oral Presentation without Review) [10] TAKAHASHI Takuto, KAJITA Yusuke, ITAI Shiroh, MIWA Yoshiyuki, NISHI Hiroko: Design of Bodily Expression Media by Integration of Skeleton Information and Shadow, SI2015, 2015.9 (Oral Presentation without Review) [11] Takuto TAKAHASHI, Masanori TSURUTA, Yoshiyuki MIWA, Hiroko NISHI: Measurement co-creative bodily expressions by hand contact interfaces, HI2015, 2015.9 (Oral Presentation without Review)