Beyond Automation: Supporting Human-Computer Collaboration

in Designing with Active Materials

Humphrey Yang

CMU-HCII-24-108
September 2024

Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213

Thesis Committee:
Lining Yao (Co-chair), HCII, CMU & Mechanical Engineering, UC Berkeley
Nikolas Martelaro (Co-chair), HCII, CMU
Scott Hudson, HCII CMU
Takeo Igarashi**,** Department of Computer Science, University of Tokyo

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Human-Computer Interaction

The research reported here was supported, in whole or in part, by the National Science Foundation under Grant No. 2118924 and No. 2427455, Carnegie Mellon University Center for Machine Learning and Health, and Carnegie Mellon University's Manufacturing Future Initiative.

KEYWORDS

Computer-aided design, CAD, digital fabrication, design tools, optimization, shape-changing interface, tangible interaction, computational design, 3D printing, 4D printing, personal fabrication, active materials, morphing materials, actuation, sensing, device design, machine learning, interactive simulation, compliant mechanism, kinematics, mechanical design, virtual haptics, wearables, rehabilitation, robotics.

Abstract

Active materials design arises as a new concept in physical design and fabrication. These materials have dynamic properties that can be activated by external stimuli, enriching physical devices' functional versatility and interactivity upon integration. Their application spans various domains, from engineering to design and art. In human-computer interaction, active materials allow computational affordances to seamlessly blend into everyday life, augmenting physical interfaces' interactions and versatility. However, as we begin to leverage active materials in physical design practices, the need for a design infrastructure also surfaces. While plentiful digital fabrication tools help us materialize ideas, designing active material systems is inherently challenging due to their novelty and dynamism. Their exotic behaviors are unfamiliar to common designers; their dynamism also requires spatiotemporal reasoning. The challenge is further exacerbated when applying active materials in real-world design problems, where the designer must simultaneously navigate constraints, opportunities, and objectives that arise from the design context. This thesis acknowledges these challenges and asks: How can computational tools support designers to work with active materials when addressing real-world, contextualized design problems?

This work synthesizes a set of computational toolmaking motifs for active materials design. The motifs administer a computational design tool's interaction with the user, empowering their ability to manipulate and reason about active materials. This dissertation advocates a shift from developing "tools that solve design problems" to "tools that help designers find solutions." These tools collaborate closely with the designer instead of automating the design process. By allowing both the computational design agent and the human designer to co-steer the course of design, both parties may do what they do best in finding satisficing solutions for complex design tasks. In this thesis, each chapter contextualizes the toolmaking motifs in different active material systems to develop proof-of-concept computational tools. The work presented here discusses the toolmaking techniques and user interactions that respond to different classes of active material design problems, as well as their implications in supporting the user's design thinking and workflow. Each project then uses the tools to develop demonstrative artifacts to validate their usefulness in helping designers address contextualized design problems. The artifacts also highlight the novel design opportunities enabled by the emergent media.

Acknowledgment

My thesis would not have been possible without my family's unconditional support. My parents have always encouraged me to pursue my interests and passion, and I have them to thank for my passion for learning and applying new skills and knowledge. My brother and cousins were also great sources of comfort and encouragement when faced with the challenges of pursuing a Ph.D. degree. Of all people, I want to thank Ji Youn Jung for her support in completing this thesis. I am glad and grateful to have them by my side along the journey.

My lab mates in the Morphing Matter lab had always been a great source of ideas, and I truly enjoyed working with them. The interdisciplinarity was truly inspiring; it was my honor to have this opportunity to work with people across different fields. A good portion of my research skills were developed while working with them. The camaraderie will be a fond memory in the years to come. I want to thank **Jianzhe Gu, Lea Albaugh, Michael Vinciguerra, Zhong Ke, Stella Shen, Dinesh Patel, Adriane Minori, Umut Civic, Tucker Rae-Grant, Yibo Fu, Di Wu, Tate Johnson, Danli Luo, Guanyun Wang, Matthew McGeehee, Ye Tao, Yi-chin Lee, Jack Forman, Jasio Santillan, Jenny Hu, Taylor Tab, Zeyu Yan, Kuanran Qian, Haolin Liu, Tingyu Cheng**, and **Youngwook Do**. They are both my coworkers and dear friends for my time at CMU.

Among my lab mates, I want to highlight the two designers that I had the honor to work closely with – **Guanyun Wang** and **Tate Johnson**. Their talents in learning about and applying novel media inspired my line of research in CAD tools. As a tool maker, it is both a pleasure and motivation to develop tools to support their creativity.

I also had the privilege to work with many outstanding researchers along the way, many of whom I learned a lot from, and I was supported by them both in study and in career. I want to thank **Nurcan Gecer Ulu, Levent Burak Kara, Ozguc Bertug Capunaman, Catherine Mondoa, Jianxun Cui, Wen Wang, Yuxuan Yu, Teng Zhang, Gina Olson, Carmel Majidi, Mohammad Islam, Jiahe Liao, Maria Stang, Adam Feinberg, Neeha Dev Arun, Frederic Gmeiner, Kenneth Holstein, Saul Schaffer, Ashlee Liao, Victoria Webster-Wood, Jialin Deng**, **Florian 'Floyd' Mueller, Lyou Yeung, and Liang He**. Among all the people I have worked with, I want to especially thank Professor **Yongjie Jessica Zhang** for her support and encouragement during

my master's and Ph.D. studies. Her words and passion inspired me to pursue my Ph.D. degree as well as to branch out and develop my skills in mechanical engineering, which turned out to be a great asset in my research. Similarly, I want to thank **Liang He** for his support, encouragement, and advice during the latter part of my Ph.D.

A great portion of this thesis is supported by the National Science Foundation's International Research Experience for Students program. In this fellowship, I was honored to work with **Takeo Igarashi** sensei, **I-Chao Shen, Haoran Xie, Bo Zhu**, and **Koya Narumi**. **Igarashi** sensei was very supportive of my work, and I was deeply inspired by the breadth and depth of his research. His work largely inspired my vision for collaborative CAD tools. His hospitality during my stay was also appreciated. **Koya Narumi** was a true friend and a role model in research; he is the researcher I look up to even now. I also want to acknowledge **Ryan Zesch** and **Alina Chadwick** for completing the fellowship together. They were both friends and inspiring researchers to work with.

My friends and peers at CMU were great sources of inspiration and support, especially the Ph.D. students in the HCII department. The journey to complete a Ph.D. study is long and challenging. There have been many moments of doubt and uncertainty, and it was a pleasure to embark on this journey alongside them. I want to give a special applause to my peers in my Ph.D. cohort – **Alex Cabrera**, **Jesse Gonzalez, Napol Rachatasumrit, Erica Cruz,** and **Sara Kingsle**y. Pursuing a Ph.D. degree during the pandemic was hard, and mutual support was crucial in getting through the challenging time. **Jesse** was particularly supportive during the last year of my study. His passion for research was a great source of inspiration and encouragement during the writing of this thesis.

The administrators and assistants at the Human-Computer Interaction Institute, Carnegie Mellon University, were the unsung heroes behind my research and all research activities within the department. Our work would not have been possible without their support. I want to thank **Queenie Kravitz, Lea Buffington, Eric Davidson, Lindsay Olshenske, Ryan Ries, Becky Wang**, and **Geoff Kaufmann** for their support.

The faculty at the Human-Computer Interaction Institute at Carnegie Mellon University were great sources of learning and advice. It was a blessing to have a collaborative environment in the department that fosters idea-sharing and constructive feedback. I want to give heartfelt thanks to

Alexandra Ion, David Lindlbauer, **Scott Hudson**, **Kenneth Holstein, Aniket Kittur, John Zimmerman, Haiyi Zhu,** and **Brad Myers.** Their teachings brought new perspectives into my research and helped me establish a broad understanding of concepts in HCI.

I was honored to work with many mentors and advisors throughout my life. Professor **Cheng-Luen Hsueh** inspired me to take a research career and blend design and engineering into my work. His defiance against disciplinary boundaries inspired me to go beyond architecture and engage with broader topics. Professor **Daniel Cardoso Llach** was my mentor during my master's studies. His teachings and insight into computational tools as an affective design infrastructure seeded my inquiry into human-design tool collaboration.

The thesis committee members, Professor **Takeo Igarashi**, **Scott Hudson**, **Nikolas Martelaro**, and **Lining Yao,** supported and advised this thesis. Their insightful feedback helped me synthesize the thesis topic and polish the document to a satisfactory quality. I was honored to be advised by them.

Finally, my advisors are the people I want to thank the most. **Nik** has always given me good research advice, even before becoming my co-advisor. His patience and kind words guided me in writing this thesis. More importantly, his ideas on human-computational agent interaction and design urged me to review and articulate my research from a new perspective.

My gratitude toward Professor **Lining Yao** cannot be expressed in words. **Lining** has been a research advisor since my first day at CMU, and none of my research achievements would have been possible without her teaching. Her encouragement motivated me to push beyond my limits, and her interdisciplinary research inspired me to take a similar path. When faced with uncertainties in research, I knew that I could always rely on her training to respond to challenges.

Table of Contents

Glossary

Active Materials. Materials that can sense, actuate, or reconfigure their properties upon exposure to a specific triggering factor (e.g., heat, moisture, current, etc.). Their functions result from their innate molecular composition or microstructure.

Artificial Intelligence (AI). Computational systems that can perform tasks that conventionally require human intelligence.

Compliant Mechanism (CM). Mechanical devices that achieve motion or force transmission through the deformation of flexible components (flexures) rather than traditional joints.

Computational Design. The practice of using production rules over symbolized elements to generate a design. A computational design process does not necessitate the use of a computer; it only requires stringent application of the rules.

Computer-Aided Design (CAD). The practice of using computer software or programs to create, modify, analyze, or optimize designs.

Graph Theory. A branch of mathematics that studies the relationships between nodes (vertices) and edges (connections) in networks or graphs.

Human-Computer Interaction. An interdisciplinary field that studies how people interact with computer systems, machines, and technology. The study involves the design, implementation, and evaluation of interactive computing systems.

Kinematics. The study of motion, including the geometry, velocity, and acceleration of objects without considering the forces causing the motion.

Kinetics. The study of forces acting on bodies and their effects on motion, including forces such as gravity, friction, and applied forces.

Multilayer Perceptron (MLP). A type of artificial neural network architecture used in machine learning consisting of multiple layers of interconnected nodes (neurons).

Numerical Methods. Techniques for solving mathematical problems using approximate numerical solutions rather than exact analytic methods.

Parametric Modeling. A method of representing objects by adjustable numerical parameters. The parameters can be passed by certain algorithms to generate variational models of the same class.

Physical Creative Process. The practice of designing and making physical artifacts.

Screw Algebra/Theory. A branch of mathematics tailored to analyze the properties and behavior of rigid bodies' kinematics.

Simulation. The act of predicting a system's behavior or performance under given context parameters. In this thesis, we use this term to refer to computational simulation that is strictly digital.

Chapter 1. Introduction

Computer-aided design (CAD) tools have become popular and integrated into our social-technical infrastructure in the past few decades. From revolutionizing the manufacturing industry in the mid-twentieth century [34] to the recent promise of generative artificial intelligence [73], they have expanded and augmented human imagination and ability to materialize ideas. CAD tools outperform humans in repetitive tasks that are complex and at scale, such as optimizing structural trusses at a granular level to attain optimal weight efficiency [244] or solving design problems like circuitry design that involve an astronomical number of parameters and complex dynamics [189]. They also help us integrate the needs and constraints of real-world, contextualized designs (e.g., helping architects to meet building code requirements and managing utility routing in the Building Information Modeling software Revit [10]). CAD tools' usefulness has made them integral to our modern design and engineering workflows to tackle complex and contextual problems. The world as we know it is also a product of CAD; its uses can be found across various sectors, from engineering (e.g., reducing aircraft weight [359]), medicine (e.g., design and validation of medical implant [107]), fashion (e.g., garment fitting [171]), to entertainment (e.g., computer animation [48]), art, and design [13].

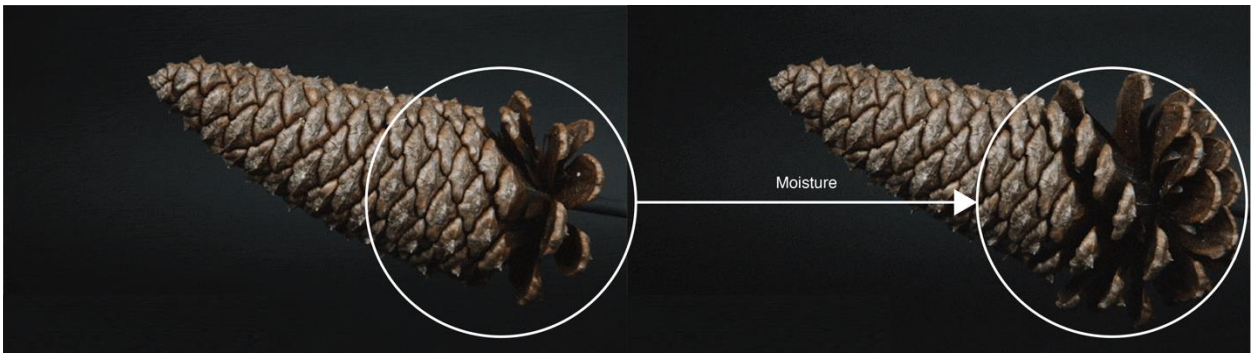

Figure 1-1. Pinecone's hygromorphic behavior (images: courtesy of the Morphing Matter Lab).

Active materials arise as a new concept in physical creativity. These materials are dynamic: they often have spatiotemporal behaviors activated by external stimuli. Through eons of evolution, living organisms have adapted to leverage such materials as part of their survival strategy responding to a dynamic world, such as pinecone's semination in response to moisture (Figure 1-1) [55], chameleons using nanocrystalline skins for adaptive camouflage [293], or treefrogs using fluorescent molecules to improve their visual acuity in low light environments [287]. These

examples signify an *"embodied material intelligence"* that differs from that resulting from neurons, consciousness, and cognition, as they are self-contained and responsive without mediation.

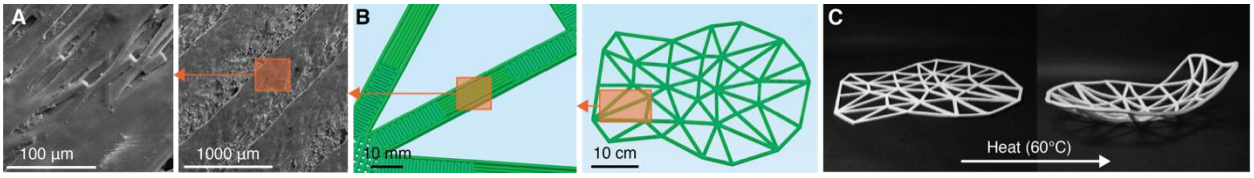

Figure 1-2. 4D printing as an exemplary active material system. (A) The microstructure of thermoplastics dictates the direction in which a printed structure may shrink. (B) When the toolpaths are designed and printed using computational tools, (C) the fabricated structure may transform from a flat piece to take a target 3D shape when exposed to heat. Images adapted from [315, 342]**.**

Active material behaviors and performances are often associated with their design and fabrication parameters. Creating a physical artifact that satisfies targeted function involves both finding the corresponding parameters and using precise craftspersonship to embody these behaviors. As such, their design practice often stands in the confluence of computational design and digital fabrication. The former helps users to model, explore, and prescribe material properties and behaviors, while the latter helps users to materialize their designs. When faced with complex and challenging design problems, active material practitioners also develop CAD tools to help make the creative process more tractable.

In the artificial world, designers and engineers have started using functional behaviors of active materials in their creation. These behaviors involve sensing, actuation (transformation), or property changes. For instance, a 3D printer could be used to align thermoplastic microstructures and result in heat-activated transformation (Figure 1-2) [172]; when combined with computational design tools, an object could be prescribed to self-transform into a targeted shape without manual handling [86]. Alternatively, with computational assistance, we could also embed property-changing materials in structures to reconfigure their physical affordance, increasing the versatility of physical interfaces. For instance, incorporating weight change could allow a tangible interface to simulate the haptic experience of holding materials of varying densities [211]. Computational tools could also help us incorporate sensing materials in inconspicuous devices to collect data about our environment [120]. In human-computer interactions, we are particularly interested in leveraging active materials to augment, program, and enrich interactions. These materials could be integrated into daily objects to enable ubiquitous computing [324] and interaction [309] or create autonomous agents that help us to engage the world around us (e.g., self-deploying sensors

[166]). When used in tandem with personal fabrication tools, they also enable us to create dynamism for self-expression [2, 68] and entertainment [232, 245].

However, as more and more active materials become available through collective research efforts, their design becomes the bottleneck for broader applications and impact [233]. The dynamism of active materials is both a boon and curse in physical creative processes. These materials' spatiotemporality is vital to their function, but it also requires designers to consider the design across different states, adding an additional dimension of time to their design rationalization, making it more challenging to design than conventional static media. On the other hand, there is also a gap for designers to appropriate and apply these materials. As an emerging media, designers must learn about the materials' capabilities and design rules while working with them. Active material's design rationalization is also convoluted by the involvement of form, material parameters, functional performances, and hedonistic values (e.g., aesthetics, fun, expressiveness) [5, 233], making it difficult to develop solutions for realistic, contextualized problems that require multifaceted design constraints and objectives. To this end, a need for active material design infrastructure surfaces.

1.1. Motivation

This thesis seeks to develop computational design tools for active materials to help designers appropriate and apply these emerging media. Taking inspiration from how CAD tools have helped us address complex design problems across various fields, I theorize that appropriate CAD tools for active material design could not only help users familiarize themselves with these materials but also use them to develop and rationalize design solutions that respond to realistic, contextualized design problems. However, unlike prior iterations of physical design paradigms that targeted static materials, active material design practice is still in its infancy with scarce computational design resources [233]. There are few tools available for active materials design. Existing tools are often handcrafted, and no tools are readily available in professional packages. A systematic computational toolmaking practice for active materials is lacking and yet to be established.

Situated among existing physical creative practices, active materials design establishes a new frontier for computational toolmaking. Such paradigm transitions from working with materials (physical media) that are prevalent in our daily lives to harnessing physical media that are exotic to our experiences. When working with conventional physical media, designers – knowing the media's affordance and design principles - can often formulate design tasks into self-contained, piecemeal problems with established frameworks. The problems could then be handed to CAD tools for problem-solving automation. By contrast, designers working with active materials are likely not knowledgeable about the media; hence, design problems are often partially or open-endedly formulated. The role of the design tool should then shift to helping designers explore the capabilities of the media while collaborating with them to tackle design problems.

The CAD toolmaking challenges I aim to address come from two themes: what makes active materials design challenging, and what challenges do designers face when working with an active material computational design tool? The first theme explores what properties set active material design reasoning apart from conventional media and what technical challenges hinder active material design from scaling up and being applied to contextualized design problems. The second theme aims to understand how designers may use computational design thinking and methods to rationalize design problems and what pitfalls could hamper designer-tool interactions in an open-ended design problem. These challenges are summarized in the following sections. Those sections also discuss how CAD tools and interactions could be designed to address these challenges.

This thesis sheds light on using computational tools to support active materials design and amplify their impact in and beyond HCI. More than active materials, some features and CAD toolmaking motifs discussed in later sections may also apply to other CAD domains. In particular, the explorations and prototyping of *"CAD tools that help designers find solutions"* may stimulate new forms of human-CAD interaction and facilitate human-computer partnership in designing for real-world open-ended problems.

1.2. Toolmaking Challenges

Rationalizing Active Materials

Dynamism sets active material apart from conventional static media and their design practices. The material's stimuli and time-dependent behaviors are core to their functional merit and manipulation difficulty. The dynamism leads to five challenges in active material design rationalization:

Spatiotemporality

While conventional material design could be conveniently represented using stationary drawings and 3D models, active material's time dependence and stimuli-responsiveness create an additional dimension when reasoning about them. Artifacts could possess forms and behaviors that shift over time, and designers must cater to their different states to produce a functional device. Knowing that each state responds to a different context, designers must concurrently tackle different sets of context-specific constraints and goals during the creative process, further complicating the efforts to manipulate them.

Unfamiliarity

Active material artifacts may be unintuitive to adopt. As an emerging media, common designers may lack an understanding of an active material's capabilities and design rules and, therefore, may not know what these materials could achieve nor what design tasks they may be applied to. Similarly, given a design objective, designers may also lack the tacit knowledge to manipulate active materials toward their targeted performance. A novice user may be clueless about how to alter an active material design's properties to arrive at different behaviors.

Unpredictability

The unintuitiveness of active material also came from its inclusion of materiality. In conventional static media design, the function and behavior of an object could often be deduced from its shape. However, in active materials design, there are more than meets the eye. In addition to their geometry, an artifact's behavior is also governed by the microstructure that constitutes its material, which may be invisible to the human eye. Consequently, two active material objects may appear

identical to the naked eye but demonstrate drastically different behaviors when exposed to the same stimuli due to their microstructural differences — the "program" that was encoded into them.

More than predicting artifact behaviors, the unintuitiveness of active materials design also extends to finding solutions. Different combinations of geometry and material programming may satisfy a design objective, and a design goal may be mapped to different solutions simultaneously. These variational solutions could be equally satisficing (i.e., reaching certain design performance thresholds [268]) toward their primary goals (e.g., morphed geometry) but vary in other nuanced, secondary qualities (e.g., fabricability). This "flexibility in solution" makes it difficult for even experienced designers to inversely deduce an ideal design solution from goals.

Integration

The challenge of unitive design reasoning is further complicated as we embed active materials as building blocks in devices or combine multiple building blocks to create an assembly. In this case, the building blocks are no longer self-contained and start interacting with (to influence and be affected by) the surrounding assembly. The device's overall behavior, in turn, is determined by the interactions (dynamics) between its constituting components. The building blocks may also contain heterogenous functions and properties, making it more difficult to predict how assemblies would perform as well as to identify effective design changes toward targeted functions. It could be arduous for even skilled active material designers to identify an integrated solution when tasked with a design objective. Nonetheless, designing active material integration and assembly to respond to realistic, contextualized design problems is not uncommon and is a pressing need.

Experientiality

On the other hand, active materials' expressiveness, which results from their dynamism, is also pivotal to their design. In addition to the different states an active material artifact may afford, the transitions between states also provide hedonistic values such as sensory stimulation, artistic display, or fun and enjoyment [5]. Designers may seek to leverage these qualities when using active materials. However, unlike geometry and quantifiable performances that computers could process and obtain, designers must fulfill these qualitative and subjective values themselves.

Computational Toolmaking for Active Materials

A structured computational tool and environment could address the cognitive challenges of active material design. The *Spatiotemporality* of active materials could be captured by adding time (or states) as an additional dimension to static representations to allow designers to manipulate designs across contexts. Incorporating material information and dynamics in the representations also makes estimating how artifacts may perform in the real world easier, thereby reducing their *Unpredictability*. The *Unfamiliarity* with emerging media could also be ameliorated by developing CAD tools to scaffold designers to explore and strategize design modifications. Specifically, to facilitate novice users in learning and understanding active material capabilities, CAD tools could use parametric design to expose factors that affect an artifact's behavior and facilitate systematic exploration of their limits and potential. Toolmakers could also incorporate tailored production rules in CAD tools to assist designers in effectively modifying an active material model toward their goals.

In addition to the modeling and digital design of active materials, CAD tools could also help designers rationalize designs for a given task. Depending on the formulation of a design task and the designer's familiarity with the media, solutions could be pre- or post-rationalized with different forms of computational aid [9]. These two types of design rationalization are also frequently referred to as *forward* and *inverse design* in HCI and beyond. In design pre-rationalization (forward design), designers may be familiar with the media and could project what a solution may look like; they could then actively sculpt the design toward their vision. In this case, CAD tools could provide functions (e.g., simulation, semantics analysis) to help designers rapidly assess the quality of their design without knowing their true intention with the media.

Conversely, in design post-rationalization (inverse design), the tool acknowledges the user's design goal(s) and helps them generate a solution that satisfice their vision. A CAD tool would then be endowed with design heuristics to become "the toolmaker's agent [34]" to help invert design objectives into parameters. Some tools may also identify designing variations instead of a specific solution to a problem, allowing designers to explore different alternatives in completing tasks.

Despite the benefit of CAD tools in active material design, designing for *Integration* and *Experientiality* remained challenging. The involvement of heterogeneous structures and dynamics

between elements makes it difficult to compose practical (i.e., accurate and fast) design tools for active material integration. The design space also increases exponentially, making it expensive to search for design solutions, not to mention navigating satisficing design alternatives. As a result, CAD tools may make strong assumptions about the workflow, problem, and heuristics to contain the problem within the time or computational budget, compromising their flexibility in solving unforeseen problems [49]. On the other hand, the *Experientiality* of active materials is challenging to parameterize and compute, and CAD tools cannot incorporate them in the search for design solutions. As a result, the qualitative factors may be compromised or neglected in workflows dominated by CAD tools. To this end, a new paradigm of active material toolmaking is needed to respond to these challenges and needs.

Applying Computational Design in the Real World

On the other hand, when addressing realistic, contextualized design problems, several challenges arise regarding how designers rationalize their designs and work with computational tools. Computational tools also have limitations when faced with the messiness of real-world problems—their numerical nature.

Intention

Throughout the creative process, designers may have different intentions with computational tools. The intentions could be exploring and discovering design options or optimizing designs to satisfice certain criteria [14]. In a computational design setting, these two intentions are often supported by forward and inverse design workflows, respectively. Still, these intentions and the resulting workflows are, in fact, intertwined [9]. Designers may freely switch between these intentions depending on what they need at the moment. As a result, computational design tools should support different intentions and allow users to switch between them freely [296].

Rigidness

Computational design tools consist of biases and assumptions about their use and heuristics. This limitation makes it impossible for computational tools to be comprehensive enough to cover all design scenarios and tasks [240], and literature has also argued that it is impractical to develop comprehensive tools [191]. However, this limitation also creates a tradeoff between the specificity

and generalizability of computational tools. On the one hand, computational design tools built for more specific uses could provide more assistance toward its targeted problem. However, their uses in other scenarios unforeseen by the toolmaker could be limited [241]. On the other hand, overly generic tools may also fall short of effectively informing design decisions for contextualized problems. Computational design tools should acknowledge this limitation and afford some flexibility in what problems they could be applied to and how they search for satisficing solutions [1].

Algorithmism

Computational tools that operate on geometries and numbers are also limited by their ability to process non-numerical factors like pleasure and aesthetics. These factors could also be inherently subjective; each designer may have different rubrics and priorities regarding these factors. Consequently, these qualities are beyond the reach of computational design tools. Still, non-numerical values may emerge from computational design, and designers should be able to acquire them alongside numerical objectives that computational agents readily address.

Reflectiveness

Design is a reflective process where designers constantly elicit, pivot, update, and experiment with their goals and constraints set for the given task [4]. Designers may not be aware of what they need until they see it [184], and their conception of an ideal design solution is subject to constant evaluation and modification throughout the design process [257]. As a result, computational design is a non-stationary search where the goals are constantly updated along with the design itself. Computational design tools should accommodate these shifts of goals and provide the stimulations needed to prompt designer reflection.

Communication

In a computational design problem, the outcome is only as good as how well the designer can wield the tools. To achieve this, the designer must develop a mental model of the computational tool's inner workings and behaviors (i.e., how they respond to specifications or instructions) [33, 352]. A design tool's agency is imprinted by its toolmaker, and its capabilities should be conveyed to its users (designers). Literature has also posited that rapid feedback could expedite learning and

accelerate users calibrating their mental model [93, 209, 225, 329]. Thus, the interactivity of tools should also be attended to while toolmaking.

On the other hand, there are also strategies to promote effective design communication between tools and users. For instance, when providing design updates and suggestions, the difference between the current and modified design should be adequately balanced to avoid premature design fixation (i.e., users locking into a design option and reducing reflection) that dissuade designers from actively participating in the design process [241].

1.3. Vision

With the abovementioned challenges, I seek to establish principles for developing active material CAD tools that help designers address realistic, contextualized design problems. Such a tool should provide the parameterized representation needed to model and navigate the material's design space, consequently helping designers to reason about their *Spatiotemporality* and *Unpredictability*. Moreover, I advocate for a paradigm shift from creating *"tools that solve problems"* to "*tools that help users find solutions."* This change steers us away from automating the search for design solutions to facilitating design tool-user collaboration in active material design problems, which in turn may allow human designers and computational agents to each take on what they do best [65], therefore producing more satisficing active material designs than either of the parties could achieve alone [1]. In particular, when a design problem necessitates complex (in the sense of *Integration*) active material design solutions, computational tools could provide the guidance needed to improve a design's quantifiable performance, but the designer still controls the design's evolution to acquire other non-algorithmic qualities (e.g., *Experientiality*).

To better support active material design reasoning, my vision manifests three toolmaking motifs:

Motif i. The tools should foster forward and inverse design thinking within each other.
Motif ii. The tools should not dominate the design workflow and allow users to co-steer.
Motif iii. The tool should help users navigate design variations toward or in the satisficing space.

The first motif responds to the different *Intentions* in computational design thinking and how they may be intertwined in solving a problem. The tool's workflow should be flexible and composable

to allow designers to switch between divergence and convergence at any creative process stage. The second motif, on the other hand, allows users to intervene and incorporate emergent or qualitative factors that fall beyond the tool's scope, bypassing its *Rigidness* and *Algorithmism*. Lastly, the third motif seeks to promote *Reflectiveness* by exposing design opportunities to the user and helping them recalibrate their design strategies and aspirations [44].

Taking inspiration from prior works [78, 337] and literature [209, 222], some other motifs that target the *Communication* between the CAD tool and the user also help to facilitate collaboration:

Motif iv. Tools should provide real-time feedback to facilitate learning and interactivity.
Motif v. Tools should exercise design in manners that assimilate human design patterns.
Motif vi. Tools should elaborate on their actions and suggestions.

The real-time responsiveness of an active material CAD tool will help users rapidly establish a mental model of the tool's workings and capabilities. Real-time behavior previews and just-in-time goal-conditioned design suggestions could also ameliorate the *Unfamiliarity* of novel active material.

1.4. Method

The active material toolmaking motifs are the key insights that this thesis seeks to communicate. We exercise the motifs to develop proof-of-concept CAD tools for various active material systems to exemplify how these ideas may be incorporated into active material design tools and computational workflows. These discourses also result in different toolmaking techniques and practices responding to some discussed motifs. For instance, in SimuLearn ([337], Chapter 3), we demonstrate that machine learning could be applied to capture the dynamism of active materials and provide effective simulation, allowing versatile workflows and interactive design problem-solving with the user. The tool also enables a hybrid design workflow to emerge between forward and inverse design, where the designers and the tool work in close collaboration to create an artifact. Alternatively, ReCompFig ([335], Chapter 5) shows that rule-based expert systems could provide just-in-time design examination and guidance for highly nonlinear active material design problems. These rules are later expanded in Compliant Metastructure Reconfigurable at Six Degrees of Freedom (Chapter 6) to help designers negotiate different strategies to achieve their design

objectives. Finally, in Interconnected Compliant Mechanisms with Active Material Integration (Chapter 7), we explore combining rule-based heuristics and numerical design solvers to provide different degrees of user control and intervention in an iterative active material design process.

The projects discussed in this thesis also exemplify different types of active material systems that toolmakers may encounter, therefore showing how the motifs could be applied to design problems of different natures. Specifically, SimuLearn represents a parametric design problem with a set number of parameters. The design tool was developed for morphing grid structures; each grid's morphing behavior is conditioned on several design parameters, and design problems often involve finding the behavior with given parameters or vice versa. On the other hand, the rest of the projects exemplify another class of active material design systems – combinatorial design. These projects use compliant mechanisms as a context for toolmaking and study, where the design's performance depends on the elements that make up the structure. In particular, a design could be assembled by a different number of elements, resulting in a design problem with nonlinear and scalable dimensions. I.e., adding, removing, and changing elements could lead to drastic changes in device performance and cause the number of design parameters to change. Finally, Chapter 7 expands the scope to explore a hierarchically combinatorial active material design problem. Here, compliant joints and active materials are integrated to produce devices with multifaced functions and enable new design opportunities. Still, the design space grows even more nonlinear and complex to navigate, but an effective computational tool helps to tackle such design problems.

On the other hand, to showcase that active material design practice can be augmented by computational tools, the developed tools are also used to create novel application demonstrations. In some examples, the computational design tools' ability to address realistic, contextualized design problems is also assessed by how they assist users during the design process and the quality of the design product. To validate this, we invite designers to use the developed tools to address design problems under a specified context (e.g., an environment model, functional requirements, manufacturing constraints, etc.). The design outcomes are then mechanically tested to validate their performance with given criteria or qualitatively examined.

1.5. Organization

Chapter 2. Related Work. In this chapter, we concisely review physical creative processes throughout history and the specifics and challenges of active materials design. Reviews on computational design tools and dead sign thinking also situate this thesis among literature and highlight several works that inspired this thesis.

Chapter 3. SimuLearn: Fast and Accurate Simulator to Support Active Materials Design and Workflows. In this work, we explore using machine learning to capture the dynamics of parametric active material design (i.e., the mapping between design parameters and their performance). The design tool also experiments with different types of active material design workflows. A hybrid workflow also emerged from this work, inspiring the following CAD tool developments to take a suggestive approach to better supporting active materials design.

Chapter 4. Background: Compliant Mechanisms. The rest of the thesis focuses on building computational tools for compliant mechanism design. In this section, we motivate compliant mechanism as an active material design context and briefly review the challenges and research gaps.

Chapter 5. ReCompFig: Designing Dynamically Reconfigurable Kinematic Devices Using Compliant Mechanisms and Tensioning Cables. In this work, we explore using rule-based expert systems to aid designers in combinatorial active material design. The CAD tool contextualizes in reconfigurable compliant mechanism design. This design problem differs from conventional parametric design tasks due to its nonlinear and dynamic landscape. To address this challenge, the ReCompFig tool employs suggestive interaction to help iterate the design: the tool provides real-time feedback to help users compose building components to establish the desired function.

Chapter 6. Compliant Metastructure Reconfigurable at Six Degrees of Freedom. This work continues the effort in ReCompFig and extends its expert system to help designers reason about different strategies for designing kinematically reconfigurable devices. Additional computational tools (e.g., analytical stiffness model, finite element analyses) are added to the design workflow to

help designers develop devices that respond to realistic needs like wearable haptics and rehabilitation.

Chapter 7. Interconnected Compliant Mechanisms with Active Materials Integration. This work targets hierarchically designed active material devices. Integrating heterogeneous materials and compliant mechanism joints assembly poses a challenging design problem that could only be tackled with the help of numerical solvers, which often find a specific solution. However, such problems could also have multiple equally satisficing solutions that should be navigated and intervened by a human designer. Therefore, we explore combining rule-based heuristics engines and numerical solvers to develop a CAD tool that handles such design problems. Such design tools also allow for different levels of designer-tool collaboration.

Chapter 8. Conclusion. The final chapter summarizes the outcome and limitations of this thesis, as well as highlighting several future work opportunities.

Appendix 1. Rationalizing Compliant Mechanisms. This section provides a review of the mathematical foundation of the compliant mechanism design method employed by this thesis.

Appendix 2. Supplementary Notes for Compliant Meta-structure Reconfigurable at Six Degrees of Freedoms. This section provides the implementation and experimental details behind Chapter 6.

1.6. Contribution

This thesis's main conceptual contribution is synthesizing a framework for computational design toolmaking of active materials. The framework pivots CAD toolmaking from creating "tools that solve problems" to "tools that help users find solutions" to better serve and augment computational design thinking and provide the assistance and flexibility needed to tackle contextualized design problems. Specifically, the framework manifests six toolmaking motifs. CAD tools developed with these motifs could allow designers and CAD tools to "do what they do best" to arrive at more satisfactory solutions. These motifs also help address certain identified challenges that make active material design difficult, making active materials design more appropriable for novice designers.

On the technical side, this thesis also contributes engineering methods and principles. Specifically, the projects propose and demonstrate different methods that could be used to create interactive active material CAD, each enabling different interactions between the designer and the tool. Each project also establishes engineering principles for different active martial systems, which in turn enabled new design spaces and opportunities. The research activities associated with this thesis also produced research artifacts, including several design tools, numerous demonstrations, and evaluation reports around these artifacts.

Chapter 2. Related Work

In this chapter, we draw inspiration from literature to inform the development of active material computer-aided design (CAD) tools. Here, we seek to answer two questions: what makes active materials design challenging, and what interactions are needed in a computationally supported creative process? To answer the first question, we first examine how physical Creative Processes have evolved over different stages of civilization. These changes are often marked by how knowledge components are relocated between different involved parties, and active materials design presents itself as a new design paradigm. Next, we review active materials and how they are designed into functional devices and artifacts. This review also highlights some challenges designers face when reasoning about active materials during the creative process.

The last two sections situate active materials design within the literature to compare and draw inspiration from related works to inform what makes a good CAD tool. In Computational Toolmaking, we select several design tools and use them to highlight different workflows and the supports they provide to a creative process. We also discuss different types of collaboration CAD tools may afford when working with the user on physical design problems. Finally, in the last section, we summarize the knowledge established about CAD Tools and Computational Design Thinking. We use this body of literature to reflect what designers may need to apply active materials in realistic, contextualized design problems.

2.1. Physical Creative Processes

The creative process involves manipulating a medium into a product, the artifact. The creative process paradigm has evolved with technological advancements, shifting industrial practices, and the tools available to creative workers. These changes are marked by the rearrangement of knowledge and skill components between (emerging) roles, often leading to paradigm shifts in how creative processes are structured (Figure 2-1). In particular, the interactions between roles are affected by how they share, complement, and synchronize knowledge.

Artisanship [117, 272] exemplifies the most primordial form of design and making as an integral activity, where the designer also doubles as the craftsperson and possesses all the necessary knowledge to materialize ideas (Figure 2-1A). Directly interacting with the media, artisans have a

good mental image of the material's capabilities and can incorporate this knowledge while formulating design solutions for certain goals. As craftspeople, they also have a good understanding of available craft skills and the viability of design ideas. Such an integral creative process is internal to the artisan and does not rely on communication and interaction between different roles. The possibility of design is bounded solely by the artisan's imagination and craftspersonship.

However, as design problems became increasingly complex and the craftspersonship required to execute plans grew, design and crafting became individual, specialized trades and finally departed, as discussed in Leo Battista Alberti's De Re Aedificatoria [3]. Alberti discusses the dichotomy between drawing and building in creative processes in the architectural treatise (Figure 2-1B). Drawing is the realm of architects and an intermediate product of design, where the arrangement of symbols and line strokes incorporate all ideas that arise from and respond to design goals, material capabilities, and the socio-technical substrate. On the other hand, building is the domain of a skilled craftsperson who strives to perfectly execute the drawings provided by architects. The dichotomy changed the media designers worked with and led to pros and cons. Designers are liberated from craftspersonship, and the making of artifacts is delegated to skilled craftspeople for higher quality and throughput. Ideas could be finely executed, and design opportunities are now limited by what designers perceive more skillful craftspeople can accomplish. Designers and craftspeople share common socio-technical-cultural knowledge upon which the symbolic representations are established. Early forms of computational design also started to appear at this time. Parametric modeling [29] emerged as a part of symbolic representations to describe shapes and forms for precise reproduction. E.g., stone shapes and cuts could be systematically explored and reproduced by using control points and guiding lines [29]. Nonetheless, instead of directly acting on materials, designers now reason about designs through representations, creating a level of abstraction between them and the artifact. That said, the close partnership between designers and craftspeople provides opportunities for designers and craftspeople to exchange information, synchronizing knowledge on material capabilities and the limit of crafts.

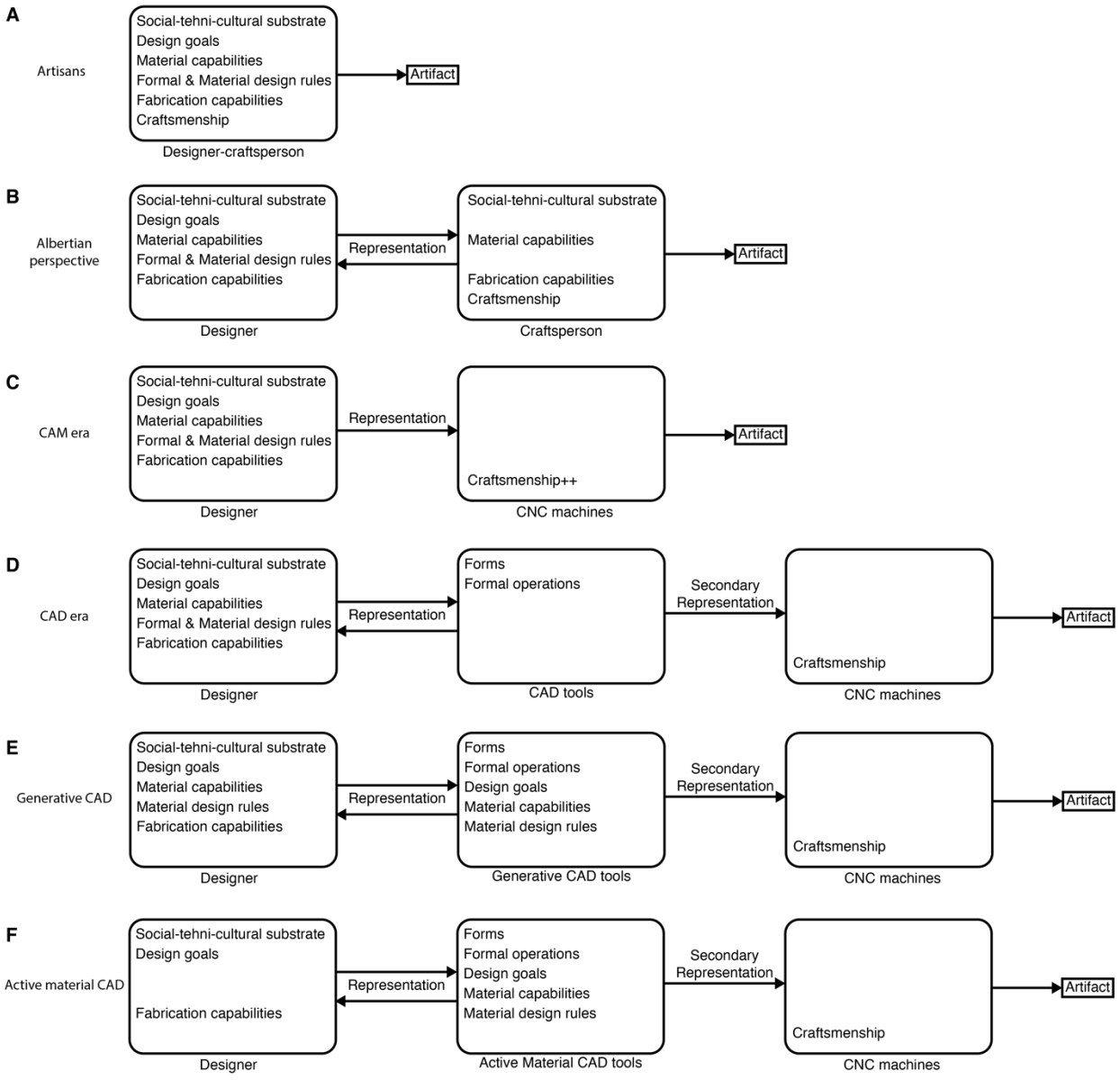

Figure 2-1. Evolution of creative processes across different eras and media and how knowledge/skill components rearrange between parties. (A)The artisan wields all tacit knowledge about the media. (B) The Albertian perspective marks the first separation of design and building in the creative process. (C) With the introduction of CAM tools, the builder (CNC machines) became even more precise but could not actively provide feedback to the designers. (D) CAD tools establish themselves as a new party in the creative process. Designers indirectly manipulate media using the rules encoded in the CAD tools to affect the outcome. (E) Generative CAD tools endowed with material knowledge to handle physical design tasks that are intractable by humans. (F) Active material design builds upon the creative process of generative CAD tools, but designers may have less knowledge and means to manipulate the media than that programmed into the CAD tool.

Fast forward to the post-war era (1949), designers are further empowered by computer-aided manufacturing (CAM) to explore an even larger design space. Advancements in electronic computers, servomechanisms, and manufacturing machines converge into computer numerically controlled (CNC) machines [177]. These machines can replicate shapes with precision

unparalleled by the human hand (Figure 2-1C), trivializing the making of complex curves and freeform surfaces (e.g., aircraft fuselage and automobile parts) [34]. However, unlike craftspeople, machines cannot understand drawings. They must be commanded by codes [76, 215] - an unintuitive representation that could be difficult for human designers to understand, let alone be used for design manipulation. This problem was later ameliorated by graphical CAD tools pioneered by Ivan Sutherland's Sketchpad [282]. CAD reestablished design through symbols and graphics for creative processes. Yet, powered by digital displays and computation, CAD tools allowed designers to iterate and alter design schematics much more effortlessly. The interface could also recognize element arrangements and patterns and automate certain drawing operations. Still, it is worth noting that at this stage, CAM and CAD are solely focused on geometry. While they allow designers to prescribe, manipulate, and manufacture shapes, they lack consideration for dynamic material properties and behaviors. For instance, while CNC machining toolpath generators consider both the shape's contour lines and the milling bit's geometry, it does not concern itself with what constitutes the milling part. The exchange between drawing and building is also unilateral. While machines can perfectly execute design drawings, they cannot actively inform designers' decisions by providing feedback on the fabrication plans or reporting their findings (e.g., errors or unexpected challenges) that occurred during the fabrication process. In contrast, a medieval craftsperson or a modern machinist would be able to co-work with the designer to help iterate the design. Consequently, CAD users become ever more detached from the materials and their affordances in the creative process.

Following CAM tools, graphical CAD tools inspired new ways of manipulating and generating designs. Continuing the adoption of parametric designs since the Renaissance, computational designers started to experiment with using production rules to generate shapes instead of explicitly modeling them (Figure 2-1D). Examples include shape grammar [135, 136] and cellular automata [193, 208], where repeated applications of production rules over graphical elements may lead to the emergence of shapes and patterns that were not explicitly modeled by the user. Later, designers and engineers started to integrate material-specific production rules into CAD tools to solve real-world design problems (Figure 2-1E). Computers could carry out calculations of astronomical magnitude and perform analysis on previously intractable problems, making them an indispensable part of designing complex physical artifacts. Examples include radar cross-section analysis for stealth aircraft and structural analysis [170]. The aid CAD tools provide also does not end there.

Generative CAD [137] was born by combining computational analysis and production rules. Given the analysis results, CAD tools and algorithms could deploy known heuristics to optimize the design for targeted performance, enabling inverse design with computer systems. These tools could help expert users easily navigate complex engineering problems, but they also help democratize design and engineering practices requiring extensive skill and knowledge. The tools are authored by expert-provided heuristics and are delegates of expert intelligence and agency. Users, even those less knowledgeable, could tap into this resource through the software and take advantage of the design methods that were previously exclusive to an elite group.

Nowadays, CAD tools had become an indispensable part of our social-technical infrastructure [34]. They are essential at solving complex design problems as well as facilitating various stages of design processes. Active materials design rises as a new episode of creative practice (Figure 2-1F). Recent developments and the wide availability of CAD and CAM tools confluence to create the foundation for harnessing the material's dynamism. Such material behaviors are often programmed into materials through precise fabrication parameter control [317], which was otherwise impossible to achieve with manual making and necessitates the finesse of a CNC machine. However, these materials are also difficult to design due to their innate, nonobvious behaviors - there's more than meets the eye. For instance, a thermoplastic sheet could be programmed to transform into a different shape upon heating [6, 172, 315], and pasta could morph when cooked to display a hidden message [290]. By contrast, prior iterations of the creative process mostly deal with materials in a static state; what you see is what you get.

Active material CAD tools are still in their early stages of development, but they have started to address material dynamism. These CAD tools often use simulations to capture dynamic material behavior, parametric modeling to describe the relation between design-fabrication parameters and the resulting material performance and employ dedicated compilers to translate designs into fabrication files. Still, the novelty of these materials leads to a critical pitfall in the creative process. These materials are novel, and there could be an imbalance of knowledge between the CAD tools and the users (Figure 2-1F). Designers may have less understanding of material capabilities and design principles than those already implemented in CAD tools, and these knowledge components must be learned through interaction with the tools. This challenge sets active material design from already established generative CAD systems that address static materials that common people are

familiar with and are competent at manipulating. Moreover, this issue is even more pronounced when the designer has no experience working with active materials [233].

Knowing these challenges, this thesis explores different ways to design and implement CAD tools for active materials. We acknowledge that computational tools are agents made by experts, and they may embody more knowledge about the media, including their behaviors and design principles than that already learned by the users. Conversely, these tools are often scoped only to manipulate and prescribe material behaviors. They often lack contextualization to address realistic design problems, thus leaving it to the user to account for. A question then arises: how do we compose CAD tools as computational design experts that help users navigate contextualized design problems and solutions?

2.2. Active Materials

Affordances and Manufacturing

Active materials are often leveraged for their novelty and integrated functions. When applied to contextualized design problems, they could be used to enrich interactions through their dynamism [312, 339], promote sustainability by affording complex functions without electronics and computers (e.g., increase recyclability [313, 315], reduce need for electronics [164, 165]), and enable new design opportunities where conventional materials fall short at addressing (e.g., edible interactions [290]). These materials can be synthetic [46, 92], biological [322, 340], or biohybrid [322]. They possess behaviors that can be triggered by certain stimuli, creating dynamic affordances or responses that are not provided by conventional materials. From a functional point of view, this activeness could be leveraged for actuation, sensing, and property changes. For instance, artifacts that have differential mechanical properties could be used to control and rectify external forces (e.g., pneumatics [219, 252, 339], fluidics[194, 310]) to create specific, designed motions [339] or logic [164, 165]. Materials that afford intrinsic mechanical responses could be activated by pH [120], heat [6, 64, 69, 315], moisture [167, 340], and electric [150, 162, 247] and magnetic fields [310] to actuate mechanical systems [46, 64, 234] or change shape [6, 313, 315] to adapt to a different functional requirement. Alternatively, materials could also emit electric currents [138] or have altered electrical properties when strained [28, 174, 323, 327] or touched

[217]. Beyond shapes and transformations, active materials could also change color [120] or stiffness [190, 357] in response to specific stimuli or reconfigure circuit logic upon manipulation.

In HCI, active materials are often discussed alongside shape-changing interfaces and share the same benefits. They augment physical interfaces to detect user interactions [81, 312] and render visual [116, 120], formal [67, 288, 312], or haptic responses [190, 348] that are not afforded by computer screens alone. Devices could also reconfigure their functions to adapt to tasks [206, 279, 357], users [235, 311], and environments [120, 340]. However, compared to conventional engineering components like electromechanical motors [202, 285] or pistons [67], active materials could be more lightweight and require a smaller footprint, therefore making them ideal for design scenarios that are sensitive to device weight and bulkiness [243]. Their small and customizable form factor also makes it possible to be integrated into daily life objects and augment them for "computation that diminishes from plain eyesight" [5, 325]. Certain materials could also be inexpensive [172] and recyclable [199], therefore making them an ideal prototyping medium and method compared to conventional 3D printing. Materials that are resilient [167] or biodegradable [281] also enable us to develop and deploy interfaces in contexts that are otherwise hazardous for electronic systems, such as sensing in natural environments [167, 322] or enabling interactions through food [56, 290, 336]. Beyond HCI, active materials have also found applications in health [125], military [118], and sustainability [168, 226]. Nonetheless, active materials still fall short at providing a comparable actuation frequency and force throughput in comparison to electromechanical systems [233]Thus, they have complementary uses and strengths and should be used situationally.

On the other hand, despite the engineering advantages offered by active materials, they could prove to be difficult to design and integrate due to their inherent spatiotemporality and required crafting precision. In particular, the behaviors of many active materials are programmed through differential material microstructures, necessitating material patterning through the fabrication process, making it faintly repeatable and scalable by menial making. As such, we often leverage digital fabrication methods (e.g., direct ink writing [317], 3D printing [172], photolithography [321], etc.) to inscribe the desired material microstructures. Still, while their fabrication challenges are largely solved by recent advances in digital machines and their growing availability, designing with active materials remains a challenge.

Designing Active Materials

The temporality and intuitiveness of active materials are core to the challenging design practice [5]. While conventional designers might have experience in sketching, drawing, and modeling desired material behaviors, translating these visions into functional requirements and, in turn, material programming could be difficult without extensive knowledge and experience about the material's capabilities and design principles. Available computational design tools often provide forward and inverse design functions to help users reason about their creations. The former helps users preview active materials' behaviors without having to physically prototype the object, thereby facilitating users to learn about the causality between design parameters and performances. The latter helps users attain design solutions affording desired behaviors through a generative process. Therefore, the user is not required to wield and exercise material-specific design rules, allowing even inexperienced users to harness active materials. As such, computational tools also serve to democratize active material design paradigms in the HCI field [233].

Despite their indispensable roles in designing with active materials, forward and inverse tools still have their limitations when addressing topologically complex designs and contextualized problems. In forward design, we often represent active materials as parameterized building blocks whose behavior depends on their shape and differential material pattern. A functional design is then created through a bottom-up process of assembling building blocks together [227]. In ascending topological complexity, designs could be made from independent [172], serially connected [6, 313], or interconnected [315] building blocks. While a user might be able to iteratively modify a single-block design toward their desired behavior by trial and error, their chances of finding a viable solution for more complex topologies rapidly diminish with exponentially larger design space and the complex dynamics and kinematic constraints between building blocks in a networked system. In this case, a forward tool falls short of assisting the user to achieve their goals. For instance, while users may be able to design linear bending beams with forward design functions and workflows [313], designing for networks of beams could be challenging due to the beams' interactions with their neighbors [337].

Conversely, inverse design tools take a top-down approach and solve designs as a generative problem. Given a performance objective, the tool finds an optimal set of design parameters for the user by applying certain heuristics. However, the heuristics are often biased and limited by the tool

composer's conception of the design problem, and the tool may be rigid [50] to cater to objectives prioritization that may shift between specific design contexts and requirements. For instance, when designing morphing wearable devices [235, 311], there could be a tradeoff between comfort (conformation) and fabricability, and their priority may depend on the wearer's preference and body location. These design criteria and heuristics may differ in the design context, and it is impractical to have CAD tools that generalize to different situations. Consequently, these decisions must be made and implemented by the designer. This notion calls for CAD tools to allow designer intervention to co-work and steer the course of design for specificities that arose from contextualized problems.

It is also worth noting that while active materials design is generative by nature and often involves quantitative computation and prescription, their qualitative merits - expressiveness [233], hedonistic and symbolic purposes [5] - are also widely recognized and leveraged in designing interactions. These merits could be transmuted into aesthetic, sensorial, and "fun" values in addition to the utility they provide, increasing the enjoyment and pleasure experienced by the user. However, such qualitative metrics could be subjective and difficult to convey to CAD tools, and the designers must sculpt the designs for these qualities alongside their targeted, quantifiable performance, creating a unique computer-aided design landscape that simultaneously features both functional satisfaction and aesthetic form finding.

2.3. Computational Toolmaking

Computational toolmaking can be characterized by various dimensions. In this work, we review and compare design tools by the *interactivity* they provide and the *target task* they are designed to support (Table 2-1).

Table 2-1. CAD tools appeared in selected literature. Names with italic font are active material CAD tools.

	Sketching/Modeling	Generative
Forward	Medley [41] Toward Evaluating Material Design Interface Paradigms for Novice Users [122]	Printone [304] A Responsive Finite Element Method to Aid Interactive Geometric Modeling [305] ReparamCAD [127] Interactive Robogami [258] Interactive Design Space Exploration and Optimization for CAD Models [260] Printed Paper Actuator [312] Morphlour [290] A-line [313]
Suggestive	A Suggestive Interface for 3D Drawing [108] A Suggestive Interface for Image Guided 3D Sketching [298] Tsugite [144]	Guided Exploration of Physically Valid Shapes for Furniture Design [302]
Inverse	Modeling 3D Shapes by Reinforcement Learning [155]	Forte [43] Printone [304] OmniAD(KiteShop) [175] Pteromys [303] Interactive Exploration of Design Trade-Offs [259] Carpentry Compiler [331] Interactive Design Space Exploration and Optimization for CAD Models [260] Designing Composites with Target Effective Young's Modulus Using Reinforcement Learning [83] Thermorph [6] *4Dmesh* [315] *Geodesy* [86]

Interactivity

Computational design tools are commonly characterized by what they provide to the user. A forward design tool helps the user author and edit a design. The user is responsible for applying their knowledge about the media and exercising their own heuristics to modify the design toward their vision (e.g., [41]). Depending on the design task, some tools may also analyze the current model to inform further design decisions. For instance, [260, 304, 305] uses simulation to help users preview their design's performance for further changes. A design's manufacturability could also be examined by a CAD tool to help users identify potential problems [258]. Creative processes supported by forward design tools are often iterative and filled with trial and error, in which the

user repeatedly modifies the design until reaching a satisfactory version. When applied to active materials design, a forward design tool often exposes and helps users prescribe material behaviors. For example, the tools in [290, 312, 313] provide a preview of how the active material design may behave given a combination of parameters, facilitating design iteration without physical prototyping. Alternatively, inverse design tools acknowledge the user's numerical design objectives (e.g., structure [43, 83, 260], aerodynamics [175, 303], fabrication planning [331]) and help them attain the desired performance. In active materials design, these performances often pertain to certain material or device-level behaviors. E.g., [6, 86, 315] provided inverse design functions to help users attain morphing sheets and meshes that are initially flat but could transform to take a functional 3D shape.

The two modes of design tools also result in different interaction patterns and implications for a creative process. When interacting with a forward design tool, the user constantly takes the initiative in shaping the design, and the design is ever-expansive as each design modification leads to a new variation upon which more modifications could be applied. By contrast, in an inverse design process, the user only takes the initiative to set up the design problem, and the tool takes the initiative to modify the design toward the objectives. This process is convergent as each iteration would bring the design closer to the objective, and only a portion of design variations would achieve that.

Forward and inverse design tools stand as the two archetypes of a spectrum, and tools could have different degrees of affinity toward either end. Suggestive design tools are an exemplar of such middle ground. Similar to forward design tools, suggestive design tools provide a direct editing environment for users to author the media, but they also acknowledge the user's goals by, e.g., assumption [302] or inference [108, 298]. Yet, unlike inverse design tools, the tool does not take over initiatives to automate design processes but instead provides recommendations to the user to help refine it. For instance, in [298], the design tool could detect the user's 3D line drawings and help snapping line control points to a reference image. The tool could also detect the user's ongoing design and suggest edits (e.g., closing curves, extrusion) or existing models from the database to speed up completion.

Targeted task

Sketching and modeling are the most fundamental parts of using CAD tools. The goals are to tangibilize, manipulate, and communicate concepts and ideas into digital representations. I.e., an externalization of the design's image as conceived by the user. While they are somewhat goal-oriented, the user may have an ambiguous or even open-ended goal regarding the specific details of the design [184]. Design objectives may also shift during sketching and modeling. Each modification could lead to further reflection and alter the user's goals, or the user may adapt their design for opportunities and constraints emerging from edits [257]. To that end, sketching and modeling could be exploratory and eliciting. In these tasks, CAD tools are often focused on providing convenient functions to facilitate completion, such as interpreting user intentions to speed up edits [122, 155, 298] or providing editing macros [41, 108].

On the other hand, generative design through CAD is focused on efficiently solving design problems at scale. In this case, the user may have a clear understanding of the qualities a good design should possess but cannot explicate the design details [304]. The desired qualities are then translated into numerical design goals and constraints and passed to the tools, which then find an elaborated set of design parameters to satisfy the objectives (if a solution is possible). Compared to sketching and modeling, the design goals are more explicit and certain. A designer could approach a generative CAD tool with clearly defined performance metrics, and the tool would provide a satisficing design in return. Beyond solving quantitative problems to attain specific solutions, recent literature also invests in helping users navigate design variations that are equally satisficing against user objectives. In this case, numerous design alternatives could be clustered by semantic metrics [127] or performance descriptors [259] to facilitate user examination as opposed to manually reviewing design alternatives one by one. Generative CAD tools could also help users explore the tradeoff between different design objectives [179, 259]. The Pareto Frontier (i.e., a collection of best trade-off design alternatives identified in the performance space [334]) could model the tradeoffs to help the user reason between different utility considerations. These extensions of inverse design allow designers to compare different design alternatives to find a specific solution that better suits their needs beyond numerical measures.

Collaboration

Active materials design is generative by nature. The design objectives often involve finding the geometric and material configurations that produce the targeted behaviors. However, as an emerging media, users must also explore its capabilities and design principles through exploratory sketching and modeling. Some active material design problems also leverage their artistic values in interaction or product design[5, 233], where immediate and meaningful feedback is essential [122]. Literature has also explored different CAD tool design features that could support discursive and generative design problems (e.g., design alternatives navigation [127, 259, 260, 302]). We categorize these techniques into two mechanisms to help users produce viable designs, each leading to different forms of collaboration between the user and the tool.

The first is automation: the user models the design problem and domain (i.e., the physical space a design occupies and where design changes take place), and the tool solves the problem for the user. If such automation could be completed in a relatively short time through, e.g., smart pipeline scheduling [305], reusing and interpolating from pre-computed results [260], or first-order approximation [303], then the tools could help users save exploration and experimentation time by ruling out design subspaces that are unsatisfactory with the objectives. However, as the user could only alter the design by changing the input requirements, they lose direct control over its nuanced form and properties. Attempting to modify the solver outputs may also knock the designs off their optimality, reducing their performance [304]. Alternatively, CAD tools could refine user-authored designs to produce viable outcomes. To do so, the user could produce a partially complete design and specify the design goals of the tool. The tools then take over the designs and alter their parameters to produce a viable solution [175, 303]. This mode of collaboration enables more direct user control, but the fixes may also distort the designer's model in the end and compromise some qualities the designer created in their intermediate product.

In this thesis, we envision and prototype a different class of design tools for active materials that work closely with the user to produce viable outcomes in line with the suggestive design tools' vision [108, 144, 298, 302]. We acknowledge that computational assistance is essential for solving complex active material design problems, but the tools are also developed with strong assumptions about the workflow and design heuristics and lack the flexibility to automate realistic, contextualized design tasks. Thus, we shift our focus from designing "tools that find activate

material solutions" to "tools that help designers solve active material design problems." In these interactions, the tool monitors the user's actions and provides just-in-time (i.e., upon each design change) feedback to help the user produce viable design solutions. This way, the tool and the user could be enabled to "do what they do best" [65]. To achieve this, the tools must be capable of rapid evaluation and heuristics to inform real-time design decisions. In the following sections, we demonstrate several techniques and methods for achieving interactive evaluation, guiding users toward objectives, and helping them navigate alternate solutions.

2.4. CAD Tools and Computational Design Thinking

Design is the rationalization of ideas [157] and by nature, a series of *divergence* and *convergence* [14, 51]. The paths to finding a good design solution often begin with exploring different ideas and concepts that show promise. The early stage is often ambiguous, incomplete, and expressive [121], therefore endorsing diversification and unconstrained exploration [123, 242]. In this case, a forward CAD tool could help users to *diverge* - quickly generate, discover, and assess design concepts. However, when situated in active materials design, the *divergence* must be carefully scaffolded: not all design variations are feasible. When modeling the parametric generation of active material design, some combination of parameters may lead to invalid geometries or behaviors exceeding that afforded by the material's limits. In this situation, the design tool should help users to confine – or pre-rationalize [9] - the exploration of the feasible design space. Examples are provided in later chapters (i.e., SimuLearn [337], ReCompFig [335]); the computational tools inform and prompt users to correct designs that are illegal to the material system. This notion makes active material design tools distinct from conventional CAD modeling tools that aim to support (almost) unbounded design discourse.

Following *divergence*, *convergence* helps to eliminate subpar design variations. Designers may also iterate to improve designs (i.e., post-rationalization [9]) based on identified goals. Here, the designer pays more attention to constraints and parameters [123, 242]. Inverse CAD tools could help designers fine-tune sub-satisficing designs to attain a satisficing variation. By cross-comparing design variations, the designer could elicit and disambiguate the ideal, satisfactory design. Design tools could also provide structured comparisons to help designers reason about the differences between variations and trade-offs between competing objectives. For instance, [259]

analyzed physical designs in their performance space and applied additional semantics (e.g., stability, mass, force, and stress) to help users better understand the generated design's qualities – like a ranking system. When faced with large numbers of design variations, tools could also cluster them into small groups of similar candidates to facilitate hierarchical exploration [179].

Still, creative processes are rarely linear, and the *diverge*-and-*converge* process could be repeatedly applied until reaching a satisfactory output. Neither forward (pre-rationalization) nor inverse (post-rationalization) creative processes alone could make complex design tasks feasible [9], and the two must be used in tandem to aid designers in addressing realistic, contextualized design problems that value both creativity and feasibility. Design tools should also be flexible to support both kinds of design processes while avoiding overly structuring the design process to hamper creativity [296].

On the other hand, creative processes had also been described as finding satisficing designs – or "good-enough" solutions – as opposed to finding the global optimal [268]. During this search, design goals could also be flexible and dynamically changing, especially when the design is embedded in a grand system and subjected to negotiable constraints [4]. Designers may not know what they want or what is possible until they see it [184, 257]. Experienced designers often reflect [257] on their conception of satisficing design (i.e., functional design [295]) and modify constraints [27] in a creative process. Navigating the satisficing space may also help users branch out [44] and provoke generative design questions like "What could other alternatives look like [62]?" In a computational design process, novel solutions generated by an inverse CAD tool could highlight new information and prompt the user to reflect and iterate on their conceived design constraints [20]. However, if a large number of details and design complexity build up too quickly when presenting design variations, designers may become overwhelmed and prematurely fixated on a specific solution, resisting changes and compromising their willingness to explore, reflect, and modify [241]. This creates a pitfall and challenge when designing collaborative generative CAD tools: how do we balance the impact of changes and provide effective communication to maintain designers' active reflection and intervention?

CAD tools are often designed with specific workflows, targeted tasks, and design contexts in mind, making them brittle [240] and difficult to generalize to more complex and diverse design problems that users may encounter in a realistic design setting. In particular, CAD tools developed for

research work – such as active material CAD tools in HCI [233] –often exclusively consider only the factors (parameters and metrics) that are essential to the design of a device and lack general consideration of contextualization. This presents a tension for computational toolmaking: how do we design tools to adapt to different design tasks and appropriation (i.e., generalizability to tasks) while making them helpful in design tasks? Tools may infer strong conception and agency about design processes to provide end-to-end design synthesis. While such a tool provides convenience, it may also limit what can be created [241]. An overly rigid tool may also hinder users from modifying and applying it for new use cases that toolmakers did not anticipate [60, 61, 134, 198, 240].

Literature has presented different guidelines in response to this dilemma. In the ideal situation, a designer should be free to express their intent, design context, and goal, but such generalization was considered impractical [191]. Still, Dix [60] suggested that CAD tools could allow users to view, interpret, and steer the course of design as opposed to assuming total control, establishing co-rationalization. This way, users and CAD tools could combine their intelligence according to their needs and incorporate values not provided to the tool in the creative process and arrive at a more satisficing outcome [1].

In terms of the interaction between designers and computational systems, generative CAD tools are often referred to as "co-creators" [12] that share initiative and have their own agency during the creative process. However, the agency of generative CAD tools may also enshroud them as black boxes with obscured behaviors and utility, bottlenecking their usefulness in creative use. As such, designers must learn to collaborate with these tools by cognitively modeling their behaviors and capabilities [33, 352]. Designers often use inductive learning (i.e., examples and hands-on tests) over declarative knowledge (i.e., manuals) to hone their mental image of the design tool [35, 62], and eliciting the CAD tool's knowledge and actions could further strengthen such learning [75].

In active materials design, the CAD tools share the same challenge of communicating design tool behaviors and capabilities. Yet, the materials may also possess novel behaviors that users must concurrently learn to master the creative process. In both cases, real-time interactivity has been shown to be useful in facilitating user learning [93, 209, 225, 329] and expediting design

exploration [122]. Textual explanations may also help to explicate CAD tool actions and avoid misuse and misinterpretation [223].

2.5. Summary

In this series of reviews, we discussed how the active material creative process differs from prior iterations of physical design paradigms. These materials have behaviors that are not obvious to the naked eye, setting them apart from the materials that we work with in conventional creative processes. Active materials' novelty and dynamism make them unintuitive to work with, and designers may have less knowledge about the media than the tools (as agents composed by experts) they work with, necessitating a shift from making CAD tools that automate to making CAD that facilitate reflection and learning. Their design also involves simultaneous rationalization of both aesthetics and quantifiable performances, creating a design practice that targets both subjective and objective values. On the other hand, existing design tools are also limited by rigid workflows and strong assumptions about design tasks, thereby hindering designers from applying active materials to address realistic, contextualized design problems where goals are ambiguous, and objectives are non-stationary (i.e., changes as the design evolves).

Examining CAD tools in other domains and establishing an understanding of designers, we posit that active material CAD tools could be made to better support designers by shifting our intention from designing "tools that find solutions" to "tools that help designers solve design problems." Such tools could take on the numerical (objective) aspect of active materials design and guide user actions accordingly. Taking inspiration from suggestive CAD tools, we also envision that users could actively take initiative during design processes instead of relying on CAD tools for automation. This way, design tools and designers could each take on "what they do best" in an active material design task. Additionally, CAD tools that co-steer the course of design may also allow designers to flexibly nudge the design toward procedurally identified design objectives. CAD tools that provide multiple strategies to complete a design task may also help users navigate design variants that are equally satisficing and prompt them to recalibrate design constraints and goals. Finally, we also highlight several CAD tool features that promote better collaboration, including prompt feedback and explained actions for mental modeling. In the following chapters, we consider these notes to prototype CAD tools contextualized in different active material systems.

Chapter 3. SimuLearn: Fast and Accurate Simulator to Support Active Materials Design and Workflows

3.1. CAD Toolmaking Motivation

SimuLearn [337] targets at a conventional parametric active material design problem – designing targeted morphing behaviors in an artifact. A design is defined by several parameters whose interactions are complex due to the innate topological arrangement of morphing elements. As a result, their dynamics are hard to predict. While readily available CAD tools [313, 315] could provide forward and inverse functions suitable for low-fi modeling and preview purposes, such active materials design practice cannot economically scale to realistic manufacturing needs (e.g., modularization) without an effective design engine that precisely captures the material dynamics. Consequently, users often resort to physical prototyping to iterate designs. Additionally, a CAD tool that supports different workflows is also lacking in the literature and is needed to allow designers to appropriate them across different stages of a creative process with active materials.

Here, we envision creating a toolmaking method that empowers users to adopt active material design practices and use them to address realistic, contextualized problems. Specifically, we posit that a fast and accurate simulator will be instrumental to active materials design. In a forward design workflow, such a simulator will allow designers – even novices – to quickly learn about the material's capabilities through rapid trial and error. In an inverse process, the tool will help users to accomplish design objectives that are otherwise difficult to achieve with existing tools or manual effort. Design alternatives could also be developed and assessed in real-time to promote design branching and reflection. We also speculate that users will be able to develop design strategies by observing the design tool's actions throughout optimization episodes.

3.2. Technical Motivation

Active materials allow us to create new modalities of interaction and fabrication by leveraging the materials' dynamic behaviors. Yet, despite the ongoing rapid growth of computational tools within this realm, current developments are bottlenecked by the lack of an effective simulation method.

As a result, existing design tools must trade-off between speed and accuracy to support a real-time interactive design scenario. In response, we introduce SimuLearn, a data-driven method that combines finite element analysis and machine learning to create real-time (0.61 seconds) and truthful (97% accuracy) simulators. We use mesh-like 4D printed structures to contextualize this method and prototype design tools to exemplify the design workflows and spaces enabled by a fast and accurate simulation method.

3.3. Introduction

In recent years, the HCI community has become interested in using active materials to enable new modes of interaction. These materials allow us to create shape-changing interfaces that are electricity-free and can respond to surrounding stimuli [291], the wearer's physiological conditions [341], or to realize novel fabrication methods [316]. However, due to their spatiotemporal behaviors and nonlinear material properties, it is difficult to predict the performance of active materials design. As a result, conventional computer-aided design (CAD) tools often must make tradeoffs between speed and accuracy. In HCI, this complication further poses a challenge in making design tools because both real-time interactivity and visual fidelity are desired to inform design decisions.

Existing simulation methods can be divided into three primary categories: geometrical methods, mass-spring models, and finite element analysis (FEA). Geometrical methods predict material performance by modeling the relationship between design parameters and experimental data (e.g., associating the length [6, 315] or layer thickness [313] of a printed thermoplastic actuator with its resulting bending angle). While they are fast to compute, these methods often take few, if any, physical parameters into account and thus are not physically accurate. Alternatively, mass-spring models [86] seek to incorporate some physical factors present in the actuation environment, but they cannot account for the complex, nonlinear physics inherent to active materials and are prone to diverge. Advanced methods such as elastic rods [19, 228] are also restricted to certain material properties and shapes, thus having a limited design space. In contrast, while FEA is physically based, its sheer computational cost renders it unviable in interactive design tools [342]. Moreover, these materials are often soft during transformation and have virtually infinite degrees of freedom, requiring high-resolution discrete models to avoid divergence, which further slows down the

computation. While model reduction methods [15, 333] can be used to achieve interactive FEA, they require pre- and re-processing whenever the model geometry is changed. Therefore, they are less ideal for supporting iterative design workflows.

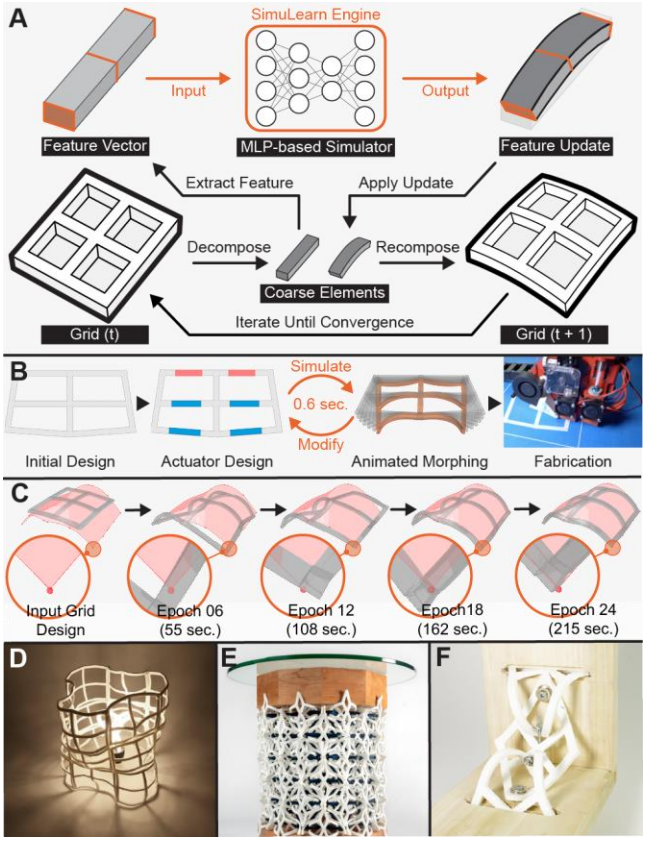

Figure 3-1. SimuLearn overview - (A) the computational theme of SimuLearn enables fast and high-fidelity (B) forward design iterations and (C) inverse design optimizations. These workflows enable design spaces that demand both simulation speed and accuracy, such as (D) modularization (lampshade), (E) material-driven parametric design (table stand), and (F) interlocking mechanisms (decorative joinery).

To address the need for an effective simulation method that allows an interactive design process of active materials, we propose SimuLearn, a data-driven simulation technique that combines FEA with machine learning (ML) to make physically accurate predictions in real time (Figure 3-1C). This method takes FEA-generated data to ensure simulation accuracy and uses ML to generalize and achieve fast computation. We apply this concept to 4DMesh-like 2x2 grid structures [315] to demonstrate this simulation technique and SimuLearn's workflow applicability. Results show that SimuLearn can produce high-quality simulations (97% accuracy) in real-time (0.61 seconds, over 1000 times faster than state-of-the-art FEA models). While the accuracy requirements may differ between use scenarios, we show that CAD tools based on this simulator (Figure 3-1A) can readily afford various modalities of design workflows (forward, inverse, and hybrid) and support complex

design tasks that require different levels of accuracy (in descending order: modularization, parametric design, and exploration), which are exemplified by three design examples derived with our CAD tool prototype. The contributions of this work include:

1. **A simulation method** that combines FEA and ML to simulate active materials fast and accurately.
2. An ML architecture based on graph convolutional network (GCN) that is adapted to topological active material systems.
3. an exemplary simulator development pipeline for 2x2 grid structures that comprises data generation, model training, and CAD toolmaking.
4. **a CAD tool prototype and design examples** that demonstrate the enabled design space.

3.4. Related Work

Active Material Simulation

Geometrical abstraction-based simulators are often used in active materials design and trade physical accuracy for fast computation. While the prediction results can visualize the transformation trend, they are not sufficiently accurate to support design tasks that require high precision like modularization. In relatively small scales, Thermorph [6], Printed Paper Actuator [312], A-line [313], and bioLogic [340] combined parametric geometries with forward kinematics to simulate tree-topological patterns, but this approach is incompatible with more complex or larger patterns like 4DMesh [315] due to their omission of physical forces. To tackle more complex patterns, [237] and Geodesy [86] used linear mass-spring models to approximate the materials' transformation. Still, this approach requires taking small time steps to avoid divergence, leading to long simulation rollout (i.e., a trial of simulation) time and cannot afford real-time CAD interactions and iterations. Similarly, although elastic rods [228] have been used to assist the design of deformable objects, their limitations (i.e., the tradeoff between noncircular cross-section shapes or viscoelastic materials [19]) make them inapplicable to certain material design problems (e.g., the viscoelastic transformation of [313, 315, 342]). Compared to these methods, SimuLearn can provide more accurate predictions and support larger design spaces while requiring similar or less computation time.

Numerical methods like FEA have also been applied to predict material transformations [26] and are often used in standard commercial systems. These methods use physically-based material models and boundary conditions to produce more accurate results, and their accuracy allows for utility beyond visualization. For instance, FEA has been applied to design adaptive actuators [24], multi-stage transformations [24, 25], and self-folding structures of complex topologies [351]. A recent work [342] also demonstrated using FEA as a backend engine to design robust artifacts made of composite materials. Yet, FEA involves establishing and solving large linear systems, making them time-consuming to perform even on supercomputing servers. Alternatively, Transformative Appetite [318] produced fast simulations by geometrically interpolating between precomputed FEA results, but this approach can only support a limited number of design parameters. Model reduction methods have also been used to achieve interactive FEA in animation [15] or material design [333]. Although they afford two-orders faster speeds, they also require precomputing the model's input motion and material modes, which leads to a delay when launching the editor. Changing the model's shape also requires reprocessing, thus making them less practical to use in the early stages of design, where geometrical modifications are frequent. By contrast, SimuLearn uses abstract graphs to flexibly represent shapes that follow a specific topology and can take on more design variables while requiring little precomputation, leading to three-orders faster acceleration, larger design spaces, and better interactivity.

Data-Driven Simulation

Data-driven simulation methods have recently been used to accelerate simulation in various ways, such as numerical coarsening [37], subspace dynamics modeling [94], and reaction-diffusion[149]. These approaches use accurate simulators to trade precomputation effort for better runtime performance. When combined with ML, data-driven methods can also make simulations more accessible in various domains like fluid dynamics [142], biomechanics [154, 176], and solid mechanics [128]. While ML-based techniques require additional data collection, and their generalizability is limited by the dataset, they also offer unique advantages such as parallelizability, end-to-end differentiability [349], and often three-orders faster speed. SimuLearn takes an identical approach and uses FEA as the source of data to ensure simulation accuracy. Moreover, in order to support the object-oriented modeling (i.e., constructing design by compositing elements) of active materials design, we take inspiration from the GCN in [17, 196, 248] and use graphical

representations in this work. Unlike convolutional neural networks (CNN) [210] that require high-resolution voxelization/pixelization, GCN also takes advantage of the model's intrinsic topology to represent them with fewer yet more effective features and make ML models easier to train.

Functional Simulation in HCI

Simulations have been widely used in HCI to make inverse design tools for various material types. For elastic materials, literature[36, 169, 346] used variations of FEA to enable users to predict and design shape-changing interfaces with complex deformation behaviors. For rigid materials, Forte [42], AutoConnect [132], and [260, 338] used physical simulations to augment design tools and produce structurally optimized objects. In architectural scales, TrussFab [131], and TrussFormer [129] also used interactive simulation to guide users to design pavilions that met structural demands.

Other than design optimization, simulations also played a central role in computational fabrication. For instance, using simulation as a backend engine, Ion et al. [110] enabled users to create complex Metamaterial Mechanisms [109], [254, 330] can optimally embed electronic components into 3D printed objects, and AutoConnect [132] empowered users to create 3D printable and robust connectors. Sequential Support [213] also used simulations to harness time-dependent material dissolution as a fabrication strategy. Situated among this literature, we believe that SimuLearn's speed and accuracy will allow available CAD tools to become more augmentative and effective in forward and inverse design tasks. Taking inspiration from Dream Lens [178], we also believe that SimuLearn can support generative tasks and allow users and computers to co-design active materials.

3.5. Material System

Our 4D printing material system is based on polylactic acid (PLA) and is identical to the bending-based printing strategy of 4DMesh [316] (Figure 3-2). However, we constrain the grids to have a 2-cell by 2-cell configuration to simplify the ML problem space, and we opt not to use even smaller grids (i.e., 1x1 grids, rectangles) due to their confined design space. While the length of the beams may vary, their width and thickness are set at 7.2 mm and 4 mm, respectively. The actuators are assigned to the beams in 25% increments (Figure 3-2C), and the maximum curvature was

measured to be 1.95 degrees/mm. We also make several improvements to 4DMesh's printing toolpath to facilitate FEA modeling (Figure 3-2A), which includes substituting the porous passive with solid (i.e., 100% infilled) constraint blocks and printing the joints with alternating infill directions to minimize their transformation.

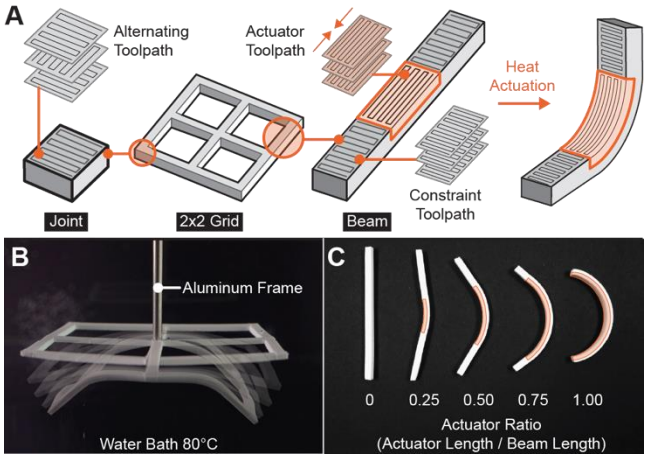

Figure 3-2. Our 4D printing material system - (A) Grid structure and toolpath design, (B) actuation setup, and (C) quarterly assigned actuators (printed and actuated). Actuators are highlighted with an orange outline in (A) and (C).

Printed structures are fixed on an aluminum frame to remain still and submerged throughout actuation in an 80 °C water bath (Figure 3-2B). Note that the grids are glued to the aluminum stand at the central joint, corresponding to FEA's fixed-joint assignments. Actuated grids are retrieved from water when the temperature drops below 60 °C, PLA's re-solidification temperature. In our batch-to-batch printing and actuation consistency tests, we observe that a 150x150 mm^2 grid takes 45 minutes to print, and the diagonal span of grids may vary by 4.09% (with respect to grid dimension) after actuation. This number is regarded as the baseline accuracy requirement of SimuLearn.

3.6. Algorithm Design

SimuLearn's implementation comprises two steps - dataset curation and ML model training. Dataset curation uses a physically-based FEA model to generate raw FEA results, which are later extracted to create a dataset for ML model training. Next, a GCN-based ML model learns from the dataset to become a generalized and accelerated simulator, which can then be used to compose design tools. In particular, SimuLearn relies on multilayer perceptron (MLP)-based GCN models to carry out fast computations. This ML model allows us to represent the design using coarse

elements described with succinct yet critical features, which drastically cuts down the number of computational units (Figure 3-3). Leveraging MLPs as nonlinear regressors, SimuLearn can also simulate with large time-steps without compromising accuracy. Moreover, MLPs and GCNs are based on rapid, vectorized computations, making SimuLearn faster to compute than FEA and even comparable with geometrical methods.

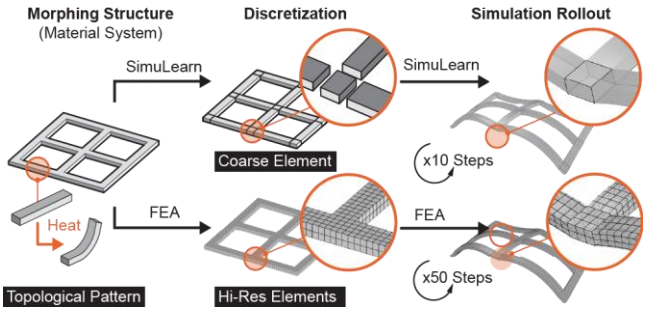

Figure 3-3. Differences between SimuLearn and FEA.

Figure 3-1A summarizes the computational theme of SimuLearn. Given an input design, we decompose the model into coarse elements represented by numeric features, compute pairwise interactions and elementwise updates with MLPs, integrate the update into the numeric features to derive the elements' status at the next time step, and repeat these steps until the simulation converges (i.e., no further transformation). In this computational flow, each iteration of the steps is identical to making one simulation increment in FEA. We can also arrange multiple SimuLearn engines in sequence to tackle complex physical systems that involve multiple stages - such as the sequential transformation of 4D printed PLA due to stress release and creeping.

3.7. Dataset Curation

FEA Modeling

We use the analysis software Abaqus and follow [343] to establish a physically-based FEA model for our material system. This FEA model adopts a two-stage strategy to simulate 4D printed PLA: the first stage corresponds to the residual stress-induced transformation, and the second stage depicts PLA's creeping under gravity. The accuracy of this FEA model is reported to be above 95%. We refer readers to [343] for more technical details. The FEA solvers are configured to output a smooth animation of transformation processes - the first stage solver outputs ten equally

spaced frames by procedurally releasing 10% of the total residual stress. In contrast, the second stage solver outputs only one frame due to relatively small deformations. Lastly, we use Abaqus2Matlab [221] to convert FEA results into .csv files.

Data Generation

We use a parametric script to generate different grid designs and FEA input files. The script initializes a design as a regular 2x2 grid and varies its morphing behaviors by randomly moving vertex positions in-plane and assigning actuators (Figure 3-4). As a result, the generated grids would have different shapes and transformation behaviors while being topologically consistent, allowing for using regular expressions during feature extraction. It is worth noting that the variance of the design parameters bounds the ML model's generalizability, and if the ML model is presented with out-of-range grid design parameters, it is likely to produce less accurate results. Thus, to produce a simulator for a targeted design space, these factors should be taken into consideration and conveyed in the design tool.

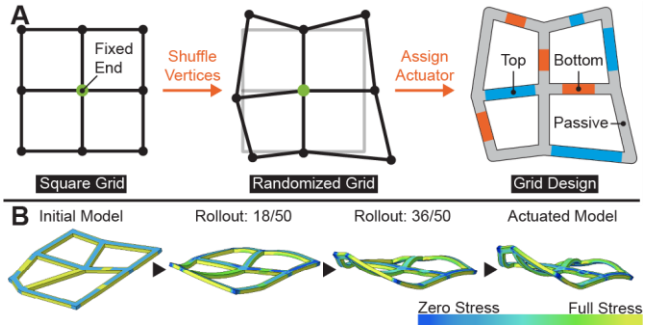

Figure 3-4. (A) Random grid design generation procedure. (B) FEA result of a randomly generated grid.

Feature Extraction

Computed FEA trials are used for feature extraction to obtain the training dataset. We rotate and mirror the simulation trials in-plane to eliminate orientational biases and procure more data points. At each time step, a grid is represented as an abstract graph $G = ([J], [N], [E])$, in which the unsigned adjacency matrix $[J] = \{J_{ij}\}_{i=1\cdots N_e, j=1\cdots N_n}$ describes the connectivity between joints and beams, and the node and edge feature matrices $[N] = \{N_i\}_{i=1\cdots N_n}$ and $[E] = \{E_i\}_{i=1\cdots N_e}$ encode the joints' and beams' feature vectors, respectively.

Each edge feature vector E_i encodes the information of three cross-sections located at the start, center, and end of a beam. The coordinates of the four corner vertices describe the shape of a cross-section, and the residual stress is represented by the stress field located at the Gaussian quadratures [238] around each of the corners (Figure 3-5A). On the other hand, N_i uses eight corner vertices to encode a joint's cuboid shape and omits the stress field information due to the lack of active transformation (Figure 3-5B). In addition to the physical information, N_i and E_i also have additional feature values to describe their design (i.e., a float value to indicate beam actuator assignments and a binary value to indicate fixed-end conditions of joints) and relative position to the fixed-end (i.e., the element's center point). Lastly, for each adjacent joint-beam pair, J_{ij} uses a non-zero number to encode their face-to-face adjacency mode (Figure 3-5C).

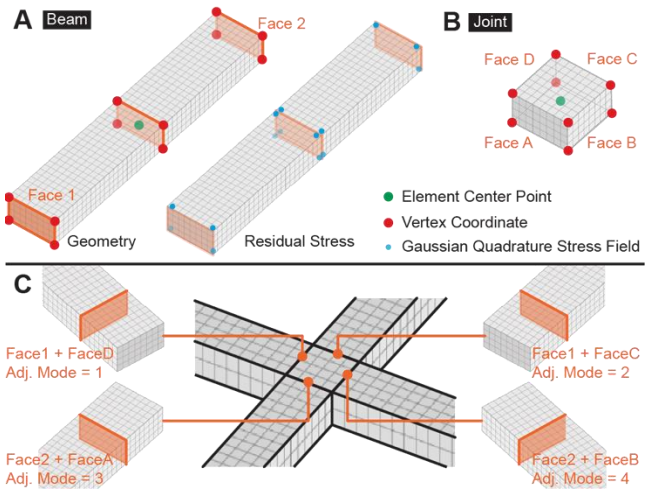

Figure 3-5. The feature sampling points of (A)beams and (B) joints. (C) An illustration of different adjacency modes.

3.8. Machine Learning Model

Model Architecture

Figure 6 provides a hierarchical overview of our ML model architecture. At the top level, the complete simulator consists of two SimuLearn engines that correspond to each stage of the FEA solver (Figure 3-6A). The first engine recursively updates the input grid ten times to capture the first FEA stage's incremental simulation, whereas the second engine only updates once. At the next level, taking inspiration from [249], each SimuLearn engine uses two sequentially arranged interaction networks (INs) [16] to compute a grid's update (Figure 3-6B). Given G_t, a grid's

graphical representation at time t, the first IN allows for element-wise interactions to propagate throughout the grid by obtaining a latent graph G_t' that abstractly describes the summed effect subjected by all other elements over a beam or joint's transformation. The second IN then takes the concatenation of G_t and G_t' to compute the input grid's update ΔG_t. Finally, the graph at the next timestep G_{t+1} is obtained by adding ΔG_t to G_t.

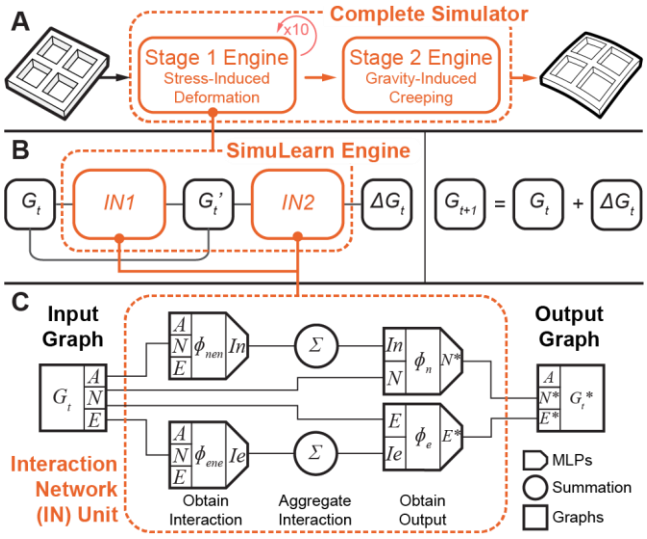

Figure 3-6. The hierarchy of our ML model architecture. (A) Using two SimuLearn engines to approximate the two-stage FEA model. (B) The double IN architecture of a SimuLearn engine. (C) The MLP layout within an IN unit.

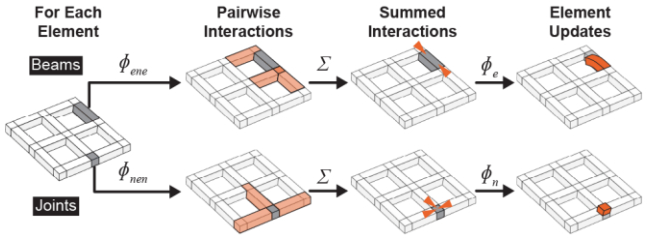

Figure 3-7. Visualization of GN units' forward computation.

INs are the fundamental building blocks of SimuLearn. Figure 3-6C characterizes an IN's forward computation to obtain its output. First, the model uses two interaction MLPs (ϕ_{nen} for node-edge-node and ϕ_{nen} for edge-node-edge interactions) to compute the pairwise interaction vectors (In or Ie whose length is a hyperparameter), which describes a neighbor's influence over a receiver element. Next, for each element in the grid, the IN sums the interaction vectors that the element is the receiver of to obtain a convoluted interaction vector (In_{conv} or Ie_{conv}) that represents the entire grid's influence over its transformation. Lastly, the element's corresponding output MLP (ϕ_n *or* ϕ_e) then takes the element's feature and convoluted interaction vector to obtain their output ($[N^*]$

or $[E^*]$). Figure 3-7 visualizes this computation flow, and Algorithm 3-1 is a snippet of the forward computation of an IN. In an interaction pair, the first three items (N_i, E_j , N_k or E_i, N_j , E_k) describe the sender, conduit, and receiver of interaction, and the last two items (J_{ij}, J_{kj} or J_{ji}, J_{jk}) indicate the adjacency mode between the elements.

Algorithm 3-1. Interaction Network, IN

Input: Graph, $G = ([J], [N], [E])$,

for each $J_{ij} \neq 0 \in [J]$ **do** // edge-node-edge interaction

  for each $J_{kj,k\neq i} \neq 0 \in [J]$ **do**

    Gather interaction pair E_i, N_j, E_k, J_{ij}, J_{kj}

    Compute interaction $Ie_{ik} = \phi_{ene}(E_i, N_j, E_k, J_{ij}, J_{kj})$

for each $J_{ji} \neq 0 \in [J]$ **do** // node-edge-node interaction

  for each $J_{jk,k\neq i} \neq 0 \in [J]$ **do**

    Gather interaction pair N_i, E_j, N_k, J_{ji}, J_{jk}

    Compute interaction $In_{ik} = \phi_{nen}(N_i, E_j, N_k, J_{ji}, J_{jk})$

for each node $N_i \in [N]$ **do** // node update

  Aggregate $In_i = \Sigma_j In_{ij}$ per receiver

  Compute output, $N_i^* = \phi_n(N_i, In_i)$

for each edge $E_i \in [E]$ **do** // edge update

  Aggregate $Ie_i = \Sigma_j Ie_{ij}$ per receiver

  Compute output, $E_i^* = \phi_e(E_i, Ie_i)$

Output: Graph, $G^* = ([J], [N^*], [E^*])$

ϕ_{nen}: node-edge-node interaction network

ϕ_{ene}: edge-node-edge interaction network

ϕ_n: node output network

ϕ_e: edge output network

In: latent node interaction vector

Ie: latent edge interaction vector

Unsupervised Data Normalization

In order to reduce redundant data variance and improve feature quality, we statistically analyze the training dataset to produce normalizers for each MLP in our ML model. A normalizer applies a series of transformations to a data point to form the MLP's input, including moving the interaction

pairs or elements to the spatial origin to remove locational variance, using principal component analysis (PCA) to reduce feature dimensions, and using affine transformation to produce zero-mean, unit-variance inputs for MLPs. In our implementation, setting the PCA information cut-off to 98% leads to halving the feature lengths, enabling faster model convergence and reducing overfitting. Note that all latent vectors are omitted during normalization.

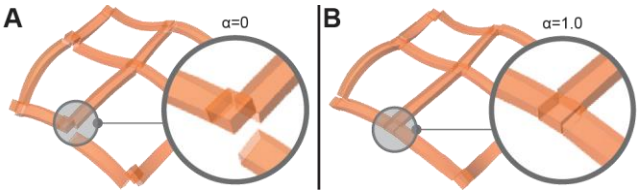

Figure 3-8. Rollout results at t=10 predicted by (A) the baseline model and (B) a model trained with the dislocation penalty.

Loss Function

When training the model with mean squared loss (MSE) alone, the vertices located at the junction of joints and beams are likely to become dislocated (i.e., becoming separated) after simulation, which violates the grid's topology and yields visually confusing results (Figure 3-8A). In, we add a penalty term to our objective function to constraint the model from producing vertex dislocation (Figure 3-8B):

$$Loss = L_{reg} + \alpha \cdot L_{disloc}$$

eq. 3-1

$$L_{reg} = \Sigma_i \left\| \check{N}_i - N^* \right\|_2 + \Sigma_i \left\| \check{E}_i - E^* \right\|_2$$

eq. 3-2

$$L_{disloc} = \Sigma_{(i,j)\in P} \left\| V_i - V_j \right\|_2$$

eq. 3-3

The first term L_{reg} is the MSE between the model output $G^* = (A, N^*, E^*)$ and the FEA ground truth $\check{G} = (\check{N}, \check{E})$, and the second term L_{disloc} the penalty term that measures the summed vertex dislocation. $\boldsymbol{V} = \{V_i\}_{i=1\cdots N}$ is the set of vertices encoded in $[N^*]$ and $[E^*]$, and $\boldsymbol{P} = \{(i,j)\}_{i,j=1\cdots N, i\neq j}$ indexes supposedly contiguous point pairs in $\boldsymbol{V}$. Lastly, α is the penalty strength and is regarded as a hyperparameter.

Model Training

Each MLP in our model comprises five hidden layers of logarithmically decreasing widths (e.g., 2048, 1024, 512, 256, 128). Since our model contains multiple MLPs and latent features, we train a SimuLearn engine as a deep network using batch gradient descent and an Adam optimizer. To improve the model's resilience against accumulated errors during simulation rollout, noises are added to the input graphs during model training:

$$Noise = (G_t - G_0) \cdot \mathcal{N}(0, \gamma^2)$$

eq. 3-4

G_t and G_0 are a grid graphical representation of time t and 0, and $\mathcal{N}$ is a normal distribution with 0 mean and variance γ. In other words, the noise is proportional to G_t's cumulative update, and γ defines the magnitude of the noise. Lastly, the hyperparameters of our method, α, γ, and dataset size are determined with a hyperparameter grid search (Figure 3-9). The optimal setting is identified as $(\alpha, \gamma) = (1.0, 0.1)$ for their small dislocation error.

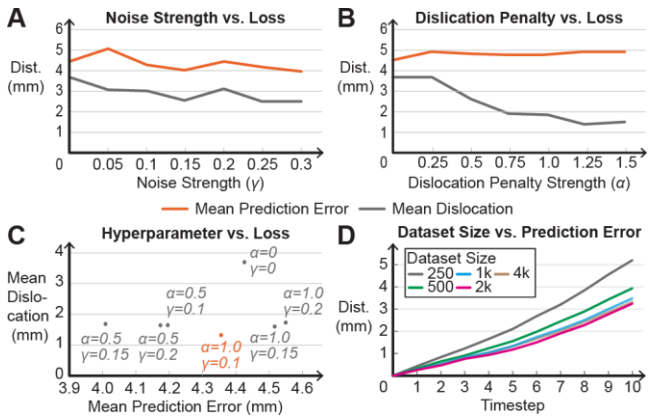

Figure 3-9. Hyperparameter search results for (A) noise strength and (B) dislocation penalty, (C) selected hyperparameter combinations, and (D) dataset size.

3.9. Evaluation

We use the data generator to obtain 4,377 2x2 grid FEA trials. Depending on the grid size, each trial takes 8 to 14 minutes to compute (mean: 10.35 min.) on a consumer-grade desktop PC (8-core Intel i9-9900k processor at 5 GHz). The mean grid dimension (the largest span from the fixed end to an outlying joint) is 94.44 mm (3rd and 96th percentile: 65.64 and 124.39 mm), and the average beam length is 51.29 mm (3rd percentile: 23.11 mm, 97th percentile: 80.19 mm).

We benchmark a simulator with 2,000 randomly drawn FEA trials (1,600 for training, 400 as held-out test data). On average, a single rollout takes 0.61 seconds to complete (including input formatting, simulation, and writing result files), which is 1018x faster than using FEA on the same machine. SimuLearn also supports parallel, near real-time simulation using a GPU (1.94 seconds for 100 grids on an Nvidia RTX 2080Ti), which is difficult to achieve with FEA due to the sheer computational cost. When measuring vertex coordinate errors between SimuLearn's predictions and FEA ground truths, the mean error is identified as 2.89 mm across all test data (Figure 3-10), which is 3.03% with respect to the dimension of grids (97_{th} percentile: 6.93mm, 4.13%).

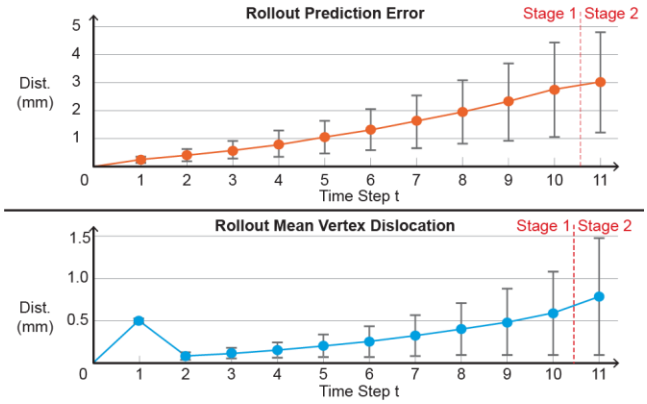

Figure 3-10. Simulation rollout accuracy of 400 held-out data.

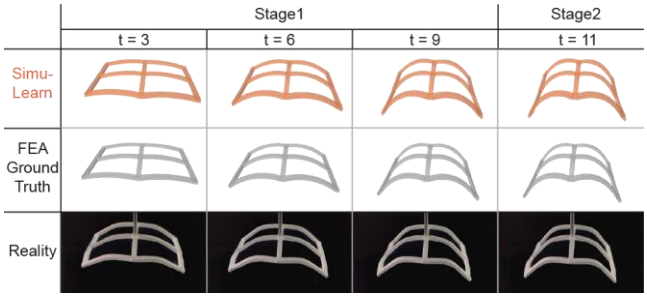

Figure 3-11. Side-by-side comparison of SimuLearn, FEA, and physical ground truth. Grid size: 132.36 mm * 77.13 mm.

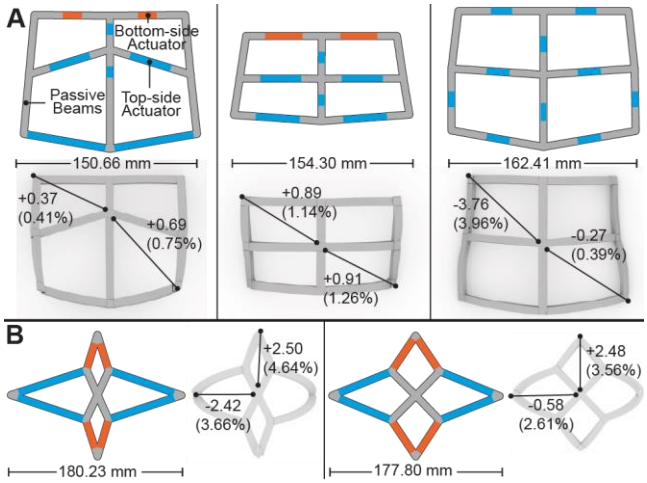

Figure 3-12. SimuLearn accuracy versus real grids shown in the (A) lampshade and (B) aggregated table design. (units: mm)

Compared to physical prototyping, SimuLearn allows users to preview a grid's transformation x9000 faster (90 minutes for printing and triggering the grid shown in Figure 3-11). As for accuracy, since vertex coordinates are unavailable, we measure the distances between several feature point pairs and report the mean error to be 2.22% for the grids shown in Design Examples (Figure 3-12), anecdotally implying a 97.78% accuracy with respect to the physical truth. While this number is inconclusive due to the limited number of samples, the error is lower than the fabrication error. Thus, it is sufficiently accurate to support design tools and tasks. In terms of smoothness, Figure 3-1C and Figure 3-17B showed small changes in design parameters would not lead to drastic changes in simulation results.

3.10. Design Tool and Workflows

We incorporate the trained SimuLearn model in a design tool to demonstrate the forward, hybrid, and inverse design workflows supported by a fast and accurate simulator. A forward design workflow allows users to iteratively modify and simulate the model until satisfaction (Figure 3-1B), enabling them to explore design options with low latency and without a clear goal in mind. On the other hand, an inverse design workflow (Figure 3-1C) helps users to achieve transformation goals when target shapes are identified. A hybrid workflow lies in between these two design modes - it allows the design tool to automate the objective aspects (e.g., optimizing design parameters towards a target shape) of the design process while enabling the users to enforce their subjective values (e.g., aesthetic concerns).

Forward Workflow

The design tool is implemented as a Rhinoceros 3D and grasshopper script, such that users can model and simulate the grids in a single environment and generate fabrication files. When forward designing a 2x2 grid, users can initialize its shape by choosing from a predefined library or by drawing the grid's skeleton as polylines (Figure 3-13A) and assigning actuators to beams (Figure 3-13B). Simultaneously, a validation subroutine will check the design against topological constraints imposed by the material system and the dataset to ensure its compatibility with the simulator (Figure 3-15). Once validated, users can then use SimuLearn to predict the grids' transformation and navigate between each timestep with a slide bar (Figure 3-13C). Users can make design decisions and manual iterations based on the simulation results, but the design tool also has a set of functions to assist users in achieving desired transformations (Figure 3-14). Once completed, the tool then processes the model design into G-code files for fabrication (Figure 3-13D).

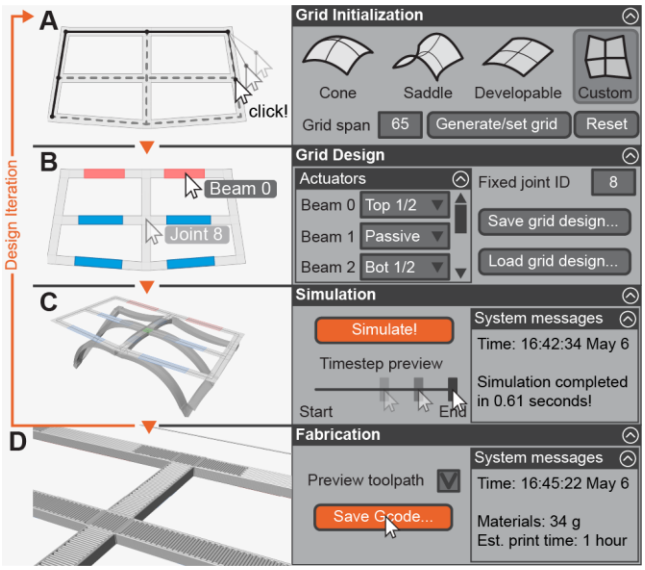

Figure 3-13. A forward design workflow - (A) initializing a grid by sketching its skeleton, (B) assigning actuators and fixed joints, (C) simulating transformation, and (D) export print files.

Inverse Workflow

In an inverse or hybrid design workflow, following the initialization of grid design, the user can specify vertex transformation goals as target points to the design tool (Figure 3-14A). The design tool then modifies each of the parameters (i.e., changing beam actuator ratio by ± 25% or moving joint positions along octagonal directions with a specified distance) to generate design variations,

batch-simulates their transformations, and compares the results against the target points to rank the effectiveness of design modifications. The rankings are determined by the averaged distance between target point pairs, and the top-ranking modification is automatically applied (Figure 3-1C). Qualitatively speaking, this gradient-free, brute-force method provides a simple yet effective way to perform design optimization.

Hybrid Workflow

The ranking system for an inverse design workflow enables a new modality of design workflow. Instead of automatically applying the highest-ranking modification, the hybrid design workflow presents the top five modifications to the user to choose from (Figure 3-14B). The user could then select the modification that best satisfies their need based on other design considerations that are not provided to the tool, e.g., aesthetics, symmetry, and fabrication concerns. These steps can be repeated as many times as the user specifies, and each epoch can be completed in near real-time (2 seconds for simulation and ranking, 8 seconds for rendering the interface).

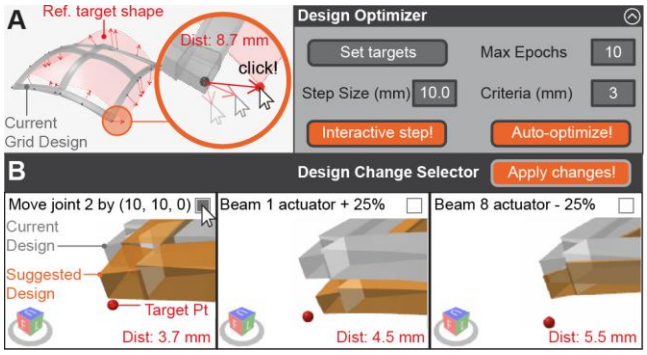

Figure 3-14. Inverse and hybrid design workflows. (A) User specifying transformation goals in the design tool. (B) The design tool suggesting ranked design modifications for the user.

Design Validation

During the modeling step, the validation subroutine provides visual cues to guide users to design grids that comply with the material system's intrinsic topology. The design tool presents two types of messages to users: *errors* (Figure 3-15A, B) that make the grid topologically incompatible with the simulator and *warnings* (Figure 3-15C, D) that may affect simulation accuracy. From a user's perspective, error messages will block the simulator from running until addressed, whereas warning messages will only prompt users to modify but do not hinder simulation.

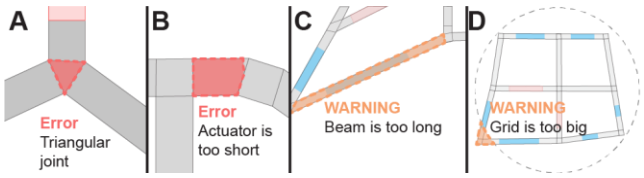

Figure 3-15. Validation messages showing (A) joint configuration error, (B) actuator length error, (C) beam length warning, and (D) grid size warning.

3.11. Design Examples

With the design tool, we invite two designers to design morphing structures in a contextualized setting to explore the effectiveness of the SimuLearn-powered tool. One designer (i.e., the designer for the decorative wood joinery and modular lampshade) had less than six months of experience working with 4D printing, while the other (i.e., the designer for the aggregated table) had no prior exposure to active materials.

Forward Design Workflow: Decorative Wood Joinery

In this example, the designer is tasked to create a 4D-printed diagonal support for a miter wood joint. The designer adopts a forward design workflow by manually modeling the grids. The simulations are used to avoid collisions, identify insertion hole placements, and predict interlocking behaviors (Figure 3-16A) between units. SimuLearn's speed allows the designer to make quick iterations and explore various design options (Figure 3-16B). In total, the designer produces 4 design variations (4-8 iterations each) in 25 minutes (1.5 minutes for simulation and 22.5 minutes used for modeling).

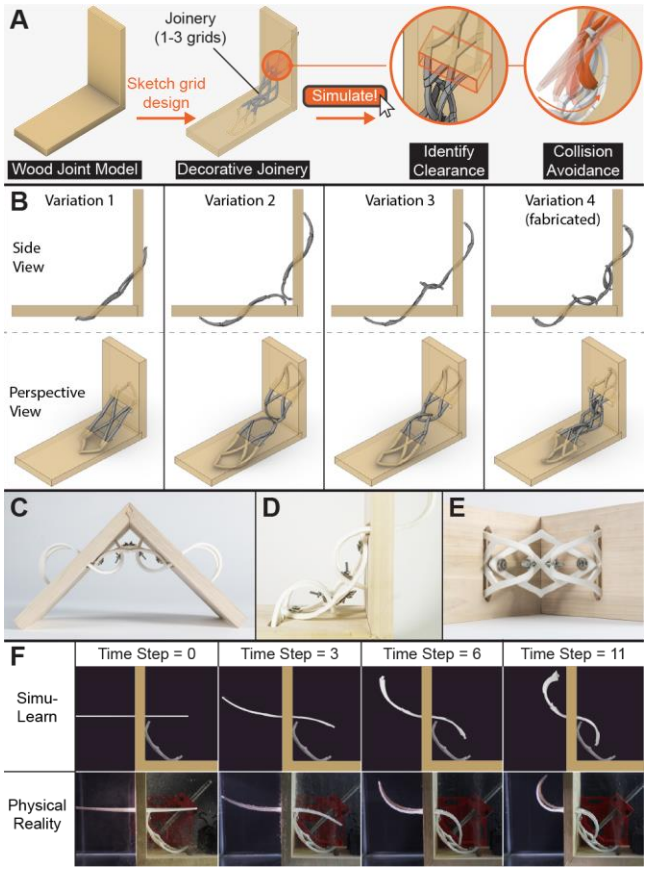

Figure 3-16. Wood joinery design - (A) Forward design scheme, (B) selected design variation, (C, D) side views and (E) details of the assembled design, and (F) transformation process.

The final design (Figure 3-16C-E) comprises three 2x2 grids that interlock and fasten together by sequential actuation. The centerpiece is first actuated and assembled into the wood joint, and then the two side pieces are actuated while being inserted into the wood panels' slits to fasten the structure together. Compared to physical reality, the max simulation errors were 2.80 mm (8.44%) and 3.09 mm (7.90%) for the center and side pieces, respectively. Noticeably, due to the model's large size, the corner joint blocks appear distorted at the end of the simulation (Figure 3-16F), but the prediction result still captures the trend of transformation and an approximative shape of the actuated grid, thus satisfying the accuracy demand of this design task.

Inverse Design Workflow: Modularized Lampshade

SimuLearn's accuracy affords design tasks that require high precision, such as patching surfaces to form larger structures [292]. In this example, the user first creates a surface model of the lampshade, but its large dimension makes it difficult to 4D print as a whole. Thus, the designer patches the surface with three repeating modules to make it more fabricable. Each module is fitted

with a 2x2 grid (Figure 3-17A) by specifying target points for vertices, and the design is carried out using the design tool's inverse design function (Figure 3-17B). Once optimized, the user then manually reorients the modules back to the surface model to generate an assembly preview. The design tool makes 22 iterations and explores 1,958 design variations in 12 minutes to bring the mean fitting error to 4.44 mm (i.e., the distance between target point pairs). Note that most of the computation time is used to render results into the design interface, and the actual simulation time is less than 50 seconds.

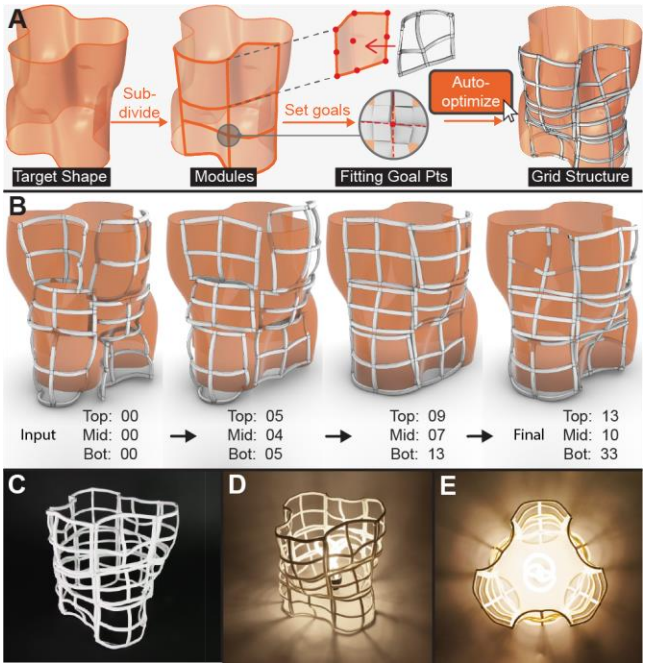

Figure 3-17. Lampshade design - (A) design scheme, (B) selected optimization epochs (epochs labeled at the bottom), and (C) assembled and (D, E) illuminated lampshade.

Figure 3-17C-E shows the printed, actuated, and assembled lampshade design. When comparing SimuLearn results with the physical reality, the max errors are 3.95 mm (4.21%), 2.422 mm (3.30%), and 3.572 mm (3.57%) for the top, middle, and bottom pieces, respectively. The errors are sufficiently small, and the modules are assembled without any noticeable issues. We report that the target shape is unachievable using previous methods because its geometry violates 4DMesh's [316] algorithmic constraints. The folding- or wire-based strategy [314] also cannot produce artifacts with sufficient structural strength. More, modularization also compartmentalizes printing time and material usage, thus mitigating fabrication risks. This design also shows that a fast and accurate simulator like SimuLearn can help us produce larger-scaled 4D printing structures, further advancing the fabrication flexibility of 4D printing.

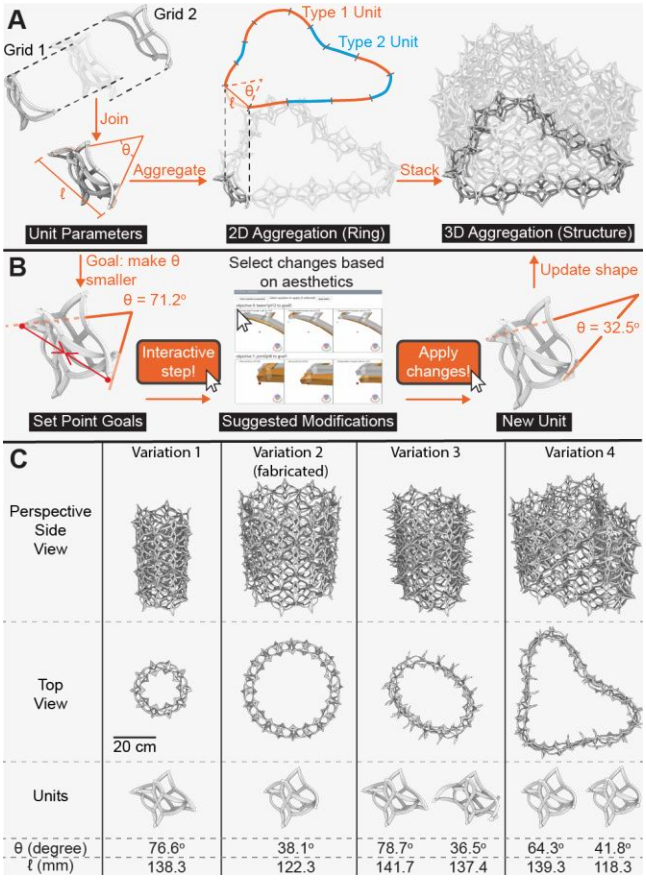

Figure 3-18. Aggregated table design - (A) parametric design scheme, (B) the hybrid design workflow to change the aggregation's shape, and (C) selected design variations.

Hybrid Design Workflow: Aggregated Table

This structure is created by connecting multiple cell units to create a ring and stacking several rings to form the entire aggregation. A cell unit is made of two intersecting 2x2 grids, and their tangential lines determine the contact angle theta of the unit, which consequently decides the curvature of the aggregation (Figure 3-18A). In this parametric scheme, the designer cannot directly control the aggregation's shape but must indirectly change the cell-unit contact angle instead (Figure 3-18C). To do so, the designer uses the hybrid workflow to specify the design tool to bring two vertices closer or away from each other, then selects from the ranked modifications to update the cell unit. During this co-design process, the design tool suggests viable options based on the simulation results, and the designer makes aesthetic judgments to ensure the aggregation is aesthetically consistent throughout the structure (Figure 3-18B). Parametric design often requires high interactivity, and this modality of active materials design is only achievable with SimuLearn as a

back-end engine. Other simulation methods will either make the design workflow impractically slow or lack the accuracy needed by parametric design schemes.

The final design is achieved after seven iterations over 15 minutes, in which only less than 1 minute is used for simulations. The max error between SimuLearn predictions and physical prototypes is 5.22 mm (8.02%) and 3.96 mm (5.95%) for the grids in the unit. The table is assembled using a Native American off-loom bead weaving technique (Figure 3-19). The level of detail and structural overhangs make this design difficult to fabricate with conventional 3D printing methods, and it would also be uneconomical to print using dissolvable support materials. In total, sixty cell units are used to produce a 52.6 cm tall, 46.8 cm wide structure.

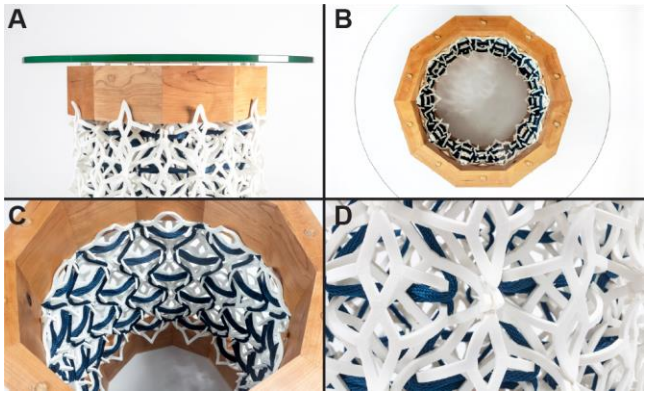

Figure 3-19. Aggregated table (A) side, (B) top, and (C) inside view pictures, and (D) the weaving technique detail.

3.12. Discussion

Emergence of the hybrid design mode

The hybrid design workflow highlights a new mode of active materials design. It combines the benefits of both methods: the tool points the user in the correct direction, but the users still have control over the course of the design. The stepwise design suggestions help the user pick modifications that would bring the design closer to the targeted morphing profile while giving the user the degree of control to account for additional traits that the design tool is unaware of, such as aesthetics or symmetry. This way, the design outcome could possess more positive qualities than the other workflows alone. The user could also seamlessly switch between the two design modes by selecting the highest-ranked option to imitate an inverse design process or select whatever they feel like as in a forward workflow.

Reflecting on the three modes of design workflow supported by SimuLearn's tool, it highlighted that inverse and forward design are interwoven when applied to active materials design. There is an element of forwardness in using inverse design tools and vice versa. Designers may not have a complete understanding of the design goals and constraints but a sensation of what a good design may look like. Some qualities of a truly satisficing design may also be lost in translating the designer's visions into machine-readable objectives. For instance, the modular lampshade's curvature is limited by that afforded by the bending beams, and the input surface model was iterated a few times until the designer was able to produce a viable design. As a result, the solution attained from inaccurate and incomplete objectives could be sub-satisficing to the true vision. The designer then needs to iterate on the design goals to improve the output design's quality. To this end, inverse design tools are also generative, parametric design systems that capture the exploratiVeness and reflectiveness of a forward design workflow, in which the input and output – parameters and objectives, respectively – swap places. The user could explore different design possibilities by slightly varying the input objectives. Thus, using inverse design tools in a creative process is akin to forward design workflows characterized by trial and error.

Similarly, forward design processes are also filled with fragments of inverse design. When the designer arrives at a point where they have a clear vision of where the design should go and become, the following steps are transformed into an inverse process where the designer seeks and implements a series of actions to modify the design toward their vision. E.g., in the wood joinery example, there were several episodes of incremental design adjustments to make sure the grids were long enough for interlocking. These "micro-optimizations" could be very small and piecemeal in the grand scheme of finding a good design that the designer does not even become aware of. The goals for these micro-optimizations could also be clearly defined or trivial so that no negotiations or uncertainty ensue. Yet, they are still critical components in exploring the design landscape – or in the pursuit of satisficing designs – by focusing the exploration on sub-design spaces that demonstrate the desired qualities.

Nonetheless, neither forward nor inverse design workflows and tools are superior to the other, and they should be used situationally in designing satisficing objects. Active materials design, in particular, considers not just numerical performances but also qualitative values like aesthetics, interaction, and pleasure that are difficult, if not impossible, to specify as optimization objectives

in CAD. There could be infinitely many solutions that are all satisficing. It could be difficult to effectively iterate a design toward numerical satisfaction in a completely forward design process, especially when the design landscape is complex and the user is inexperienced. Conversely, in the inverse design workflow, the tools are biased by their design and inner workings, and the tool's agency may not reflect the designer's true vision and additional qualitative goals. Partially conveyed design goals may also lead the design tool to fixate on unsatisficing suboptimal spaces and make it impossible to locate a satisficing design. Thus, we need some forwardness in CAD tools to afford free roaming and exploration, but also the inverseness to facilitate the search for satisfaction.

Mixed-Initiative Active Materials Design

Design goals could also be flexible and dynamically changing, especially when the design is embedded in a grander design problem [4]. Regarding design as a top-down process that divides a complex task into isolated sub-problems, active materials, and their CAD tools could be deployed to address one or a few of them. The design goals are then dictated by the conditions upon which sub-problems are interfaced with other parts in the grand scheme. In the modular lampshade example, the goal of the morphing grid modules design was to make sure the edges fit tightly together to allow connections and form a continuous surface. The inverse design function was then used to achieve this criterion. However, while these requirements may appear rigid when looking at the sub-task alone (i.e., scoped to a single module), they could become flexible in the grander scheme. The design objectives could be changed if it is also reflected on the other side(s) of the interface. For instance, the edges of a surface module could be manipulated if all modules were adjusted at the same time, and the resulting design would still be continuous and modular. Therefore, the design objective could be flexible and pivoted if necessary. This is a decision that the CAD tools cannot proxy, and designers must actively reflect, assess, and update the constraints and goals by situating the design task in context.

Human intervention is often something that is avoided by CAD tools, especially in inverse design workflows where automation is paramount. However, I want to advocate that designer intervention could play a crucial role in the search for a satisficing design. As mentioned before, the user may not have a complete understanding of a good design. The understanding must be reflected and updated by purposeful exploration of design variations. The understanding could also be difficult

to convey or translate for CAD in a numeric form, such as the case for aesthetics. New values may also arise from the design process. Literature [184] argued that in the search for satisfaction, designers may not know what they want until they see it. Yet, when the values do arise, they need to be reflected in the design process. This is challenging to achieve in a pre-programmed inverse design tool where the tool modifies the design with the given heuristics. CAD tools are shallow in the sense that they only account for the factors that the toolmaker deems relevant, and no design tool is comprehensive enough to account for all metrics associated with unforeseen design problems. Therefore, when the two shortcomings clash, the user could become frustrated when an inverse design tool fails to converge toward a satisficing design or when a forward design tool falls short of providing any support at all. This challenge is even more punishing for users who are not fluent in programming as they have no way to incorporate new, emergent factors by hacking into the tools. This notion was also observed in SimuLearn, where some study participants complained that it was difficult to translate their design goals for the tool's inverse design function, and the objectives already provided in the tool were insufficient [79].

In active materials design, design spaces and viability are also circumscribed by material- and fabrication-specific constraints that the designer may not be readily aware of or pay attention to (e.g., bending curvature limits in the modular lampshade design). Thus, involving designers in the inverse design process could be critical in prompting them to reflect and update their perception of material capability and what makes a design satisficing. Designers could also be involved beyond observation: they could be allowed to intervene in the design process to enforce and attain the objectives that arise from a reflective design process [257] and/or those that are difficult for numerical solvers to convey. Such interaction would allow designers to co-steer the course of design, akin to the suggestive design tools that had been proposed by literature [108, 144, 298, 302]. Demonstrating design alternatives or supporting users to explore design alternatives may also help them reflect and revise their conception of a satisficing design.

Learning With and About the Tool

On the other hand, does SimuLearn actually support learning? Literature has suggested that prompt feedback would help users to learn more quickly and effectively [93, 209, 225, 329]. SimuLearn's fast response could allow users to quickly learn about the physical design paradigm and its affordance through real-time interactions and trial and error with the forward design function.

Design strategies could be learned by observing the tool's "demonstrations" in the inverse or hybrid design setting.

In a later study [79], in collaboration with F. Gmeiner et al., we invited novice designers (three males and four females with ages ranging from 21 to 64 years) to use the tool to create bottle holders for bikes and observed how they interact with SimuLearn. All participants have more than two years of experience using CAD tools in architectural, industrial, or mechanical design and are either students, researchers, or contractors. At the beginning of the study, the participants were given a brief (i.e., 30-minute) introduction to the transformative grids and the design tool's basic functions.

In the post-study interview, participants reported that the active materials were indeed "unintuitive" and" challenging" to design. Yet, most were able to leverage SimuLearn's rapid Simulation to learn and become comfortable with the physical design paradigm. A participant described their learning experience:

> *"Even though I tried sketching some stuff, I think it just didn't work. So, I thought it's better if I just go into the tool and see if I will be able to do this. I tried stuff like folding one corner upwards and one corner downwards or stuff like that. I took lots of screenshots and those really helped me to understand like 'if I do this, then it's gonna behave like that' so I think initially it was a lot of trying to form a mental model and like what's the capability of this tool."*
>
> *P-S10 Interview*

On the other hand, learning design strategies through observing the tool's demonstrations was rejected. Users even complained about how unintuitive and problematic it was to work with the inverse design function. When using inverse design tools in a physical design problem, the user must learn about the physical design practice while building a mental model of the tool's inner workings at the same time, which could be overwhelming and leave the user feeling dominated by the tool:

"I feel like the collaborative process [...] it seemed a little difficult to control. I felt that SimuLearn had more control over it than I did."

P-S01 Interview

As a result, users would improvise hacks around the inverse function or abandon it altogether and proceed with the forward design mode. The same problem was also observed in the hybrid design mode. The tool simply suggests the actions to take without elaborating on the reasons, and users could become frustrated by it. Consequently, most users had difficulties interpreting the suggestions made by the tool. We note that future tools should be more explanatory in presenting solutions and suggestions. For active materials, in particular, visual suggestions alone may not be sufficient as they add a temporal dimension to the design problem, and CAD tools that primarily rely on 2D or 3D could not adequately explain the design's behaviors.

Users also reported difficulty understanding the inverse design function's output even when they were using it correctly. We speculate that this challenge may come from the disparity between how humans and the tool modify a design. The tool attempts to adjust several parameters at a time, which is contradictory to human capabilities, where we can only change one or few parameters simultaneously. While users were able to understand, at a high level, the effect of the changes (i.e., optimizing the design), they would have trouble inducing the cause for such effect and generalizing their observations into design strategies. Therefore, users were unable to capture the causality of design modifications with respect to the design goals, and the expected learning did not take place. In the future, design tools could be developed to exercise design in a human-readable format by making design changes piecemeal or unitary, as opposed to concurrently adjusting all parameters (e.g., gradient descent). Design strategy explorations and development could also be further scaffolded by a curriculum. I.e., The design tools could suggest the user design changes to test out and observe how they affect the design's behaviors.

3.13. Limitation

FEA Limitation

SimuLearn's accuracy is limited by the data source and is susceptible to the limitations of the FEA model. In this work, although our FEA model is physically accurate, it does not account for collisions during the transformation process. Although these phenomena are unlikely to occur in our material system due to their transformation capacity, future work may take inspiration from [17] to take account of collision and open new design spaces.

ML Simulator Accuracy

While our physical prototyping results showed that SimuLearn's speed and accuracy could readily support and facilitate their design workflows and tasks, there is still room for performance improvements. For instance, our current method does not use the temporality of simulation trials to train the model. Inspired by [248], we speculate that adopting recurrent ML models may further improve SimuLearn's accuracy. Incorporating other ML techniques such as encoder/decoder, system identification, or hierarchical convolution [196] may also lead to improved performances. Future works may also leverage our pipeline to generate larger datasets to mitigate the dimension issue observed in the decorative joinery design example.

Development Cost

SimuLearn trades development time for workflow conveniences by using FEA to curate large datasets for training ML-based simulators. In this work, we prioritize our data generation for the design parameters that we deem most important. To incorporate new design parameters, developers would have to curate new datasets to update the simulator. Indeed, when targeting a more general design space, methods that do not require training on any possible topology may appear to be more economical. Yet, the development cost of SimuLearn can also be easily justified by its three-order faster workflow acceleration and parallelizability, especially when the design tool is mass-deployed or repeatedly used. We also believe that SimuLearn allows developers to compose augmentative design tools for well-established active material systems like 4D printing, thus contributing to the democratization of advanced fabrication technologies.

3.14. Future Work

Generalizability and Scalability

While this work is adapted to a specific material system, SimuLearn's algorithm is also adaptable to other material systems by exchanging the FEA model and/or the feature representation. E.g., SimuLearn can adapt to Geodesy [86] by describing the continuous shells as aggregations of rectangular patches, which are then represented by their corner points, or it can further adapt to Transformative Appetite [318] by swapping the FEA model from stress-release PLA to swelling gel. Existing works have also validated the viability of ML-based physics in various engineering and design contexts [349].

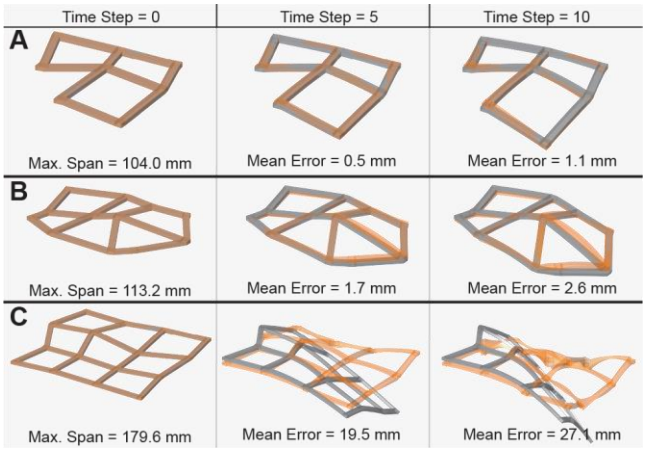

Figure 3-20. Simulation results of topologically mutated grids - (A) a 2x2 grid with partial removal, (B) a 2x3 grid, and (C) a 3x3 grid. (orange: SimuLearn result, grey: FEA ground truth).

As for scalability, GCNs intrinsically generalize to designs that have different numbers of units and are adaptable to different length scales [17], and the only limitation is dataset coverage. Anecdotally, we observe that the simulator can generalize to unseen grid topologies (i.e., having missing or extra beams) if their geometrical dimensions are within the dataset's coverage (Figure 3-20). Nevertheless, SimuLearn can also be trained to tackle topologically larger grids (e.g., a 4x4 grid) by expanding the dataset to cover targeted topologies and increasing the degree of convolution in the ML architecture. Note that the computation speed would remain identical because the elementwise transformations can be computed in parallel. We speculate that while adapting SimuLearn to larger grids would quadratically scale up the parameter space (i.e., elements may be located further from the fixed joint and be subjected to higher magnitudes of stresses), the

amount of element data available for training MLPs would also increase quadratically. Thus, it may be possible to achieve an identical accuracy using the same number of FEA trials, though further research is necessary to validate this conjecture. Nonetheless, we argue that while the simulator is limited to 2x2 grids, its speed and accuracy allow users to design larger structures using a modularization approach with even higher efficiency than previous work [315].

SimuLearn-Based Design Agents

Currently, the inverse design function optimizes the model with an unguided brute-force approach. Future works may consider using different optimization approaches to achieve better results. In particular, SimuLearn's parallelizability and speed lend itself well to genetic algorithms and evolutionary computing that require frequent performance evaluations. More than being faster, SimuLearn also enables converting indifferentiable simulations like FEA into differentiable computations, which can be leveraged to create gradient-based optimizers. Similar methods have also been shown in robotics for efficient control policy-finding [11] and co-design [106]. Situating this concept in HCI, SimuLearn as a backend engine will allow CAD tools to simulate, evaluate, and suggest designs in real time to inform high-quality decisions. With SimuLearn's debut, we also envision conversational design agents to emerge in the shape-changing interfaces and active materials context.

3.15. Conclusion

SimuLearn combines FEA and ML to enable physically accurate and real-time simulations for active materials. Results show that SimuLearn is nearly as accurate as state-of-the-art methods while being orders of magnitude faster. It also enables design tools to become multimodal platforms that support a broad spectrum of design workflows. Beyond the grid- and PLA-based material system presented in this paper, we also believe that SimuLearn can generalize to other topological patterns and materials by swapping the representation and FEA model. SimuLearn, as an enabling technology, is particularly well-suited for the HCI community, not only because of its effectiveness in improving design efficiency, but also because its interactivity allows users and computers to codesign, paving the way for human-CAD tool collaborations to unfold in the design field. We also believe that SimuLearn can augment active material CAD tools to become

conversational, educative, and accessible to the public. As the HCI community accumulates growing interests in harnessing active material behaviors, SimuLearn will likely enrich the available design and technology toolbox and empower us to unfold the potential of active, smart, and morphable materials.

3.16. Computational Toolmaking Remark

On the design tool side, SimuLearn shows that parametric active material design tools could be made to be interactive and multimodal to support both forward and inverse design workflows. The follow-up user study also showed that designers could use rapid simulations and trial-and-error to quickly familiarize themselves with the active material's unintuitive behaviors and develop tacit knowledge to work with the media. While implementing the design tool, a hybrid workflow emerged between forward and inverse workflows. This workflow allows users to steer the course of design with the design tool informing effective actions to iterate toward their vision, resembling the suggestive interfaces in literature [108, 144, 298, 302]. This mode of designer-tool collaboration may allow users to explore active material design's expressiveness in addition to numerical functions and behaviors, arriving at more satisficing outcomes.

On the other hand, the follow-up user study also highlighted the challenge of communicating design updates and decisions with the user. SimuLearrn's inverse and hybrid design functions aggressively modified the design. In particular, the inverse design function simultaneously modifies multiple parameters at a time, making it difficult for users to follow its steps. As a result, users felt overwhelmed by the tool and were repulsed by its dominance. Alternatively, the hybrid design workflow lacks explanations for its suggestions, making it difficult for users to understand.

These findings informed later design toolmaking to take a more piecemeal, constructive approach. Instead of fully automating the design process, an active material design tool may take a more stewardship approach in helping the designer at each step of the design process. The design tool should also allow users to co-steer and manipulate the design to encourage reflection and learning (about the media) through interaction.

Chapter 4. Background: Compliant Mechanisms

4.1. Compliant Mechanisms as a Study Context

Compliant mechanisms (CMs) have been an established field for centuries. However, with recent advances in their understanding and manufacturing techniques, they have reemerged as a popular mechanical design paradigm. Still, their design space is complex and turbulent. Slight changes in the design parameter could lead to drastically different device performance and functions. While a wide variety of tools could help us model and evaluate individual CM designs, the field lacks tools to support engineers in reasoning about their designs.

On the other hand, satisficing CM designs also require the user to consider factors that are difficult to include in a CAD tool, such as fabrication strategy, integration of other components, etc. Therefore, designers must explore, navigate, and alter their goals throughout the design process. A CAD tool could assist the user in achieving some design requirements, such as its kinematic affordances or motional trajectories. However, user intervention is needed throughout the design process to incorporate values the tool has not considered.

Considering these challenges, I reckon CM design is an ideal context to further my inquiries into active materials design toolmaking. The unintuitive design problem rendered it a challenging domain for computational toolmaking. Available tools are often focused on analysis or automation (i.e., finding a specific solution) as opposed to stewarding designers to complete a design task and explore different strategies to complete a design. Thus, my latter works are focused on developing design tools to help users find compliant mechanism design solutions.

4.2. Compliant Mechanisms as a Hardware Design Paradigm

Compliant mechanisms [103] are structures that enable motion by their inherent flexibility. Conventional mechanical structures rely on hinges, bearings, and articulations to afford motion, which requires multiple parts and assembly. The interface between parts also creates maintenance needs such as lubrication, dusting, and wear as parts rub against each other. Exposed joints also made conventional mechanical joints vulnerable to environmental hazards such as wetness and

dirt. By contrast, CMs could be made as a monolithic device, reducing the need to produce and assemble multiple parts. Their simplicity made CMs favorable in low-cost or mass-production use scenarios such as bottle caps, springs, and crafting tools (e.g., forceps). Without parts rubbing against each other, CMs are also less susceptible to wear and have reduced need for lubrication, making them more reliable in extreme environments (e.g., space exploration, military use, underwater) where the operational condition is punishing, and conventional mechanisms would require frequent maintenance.

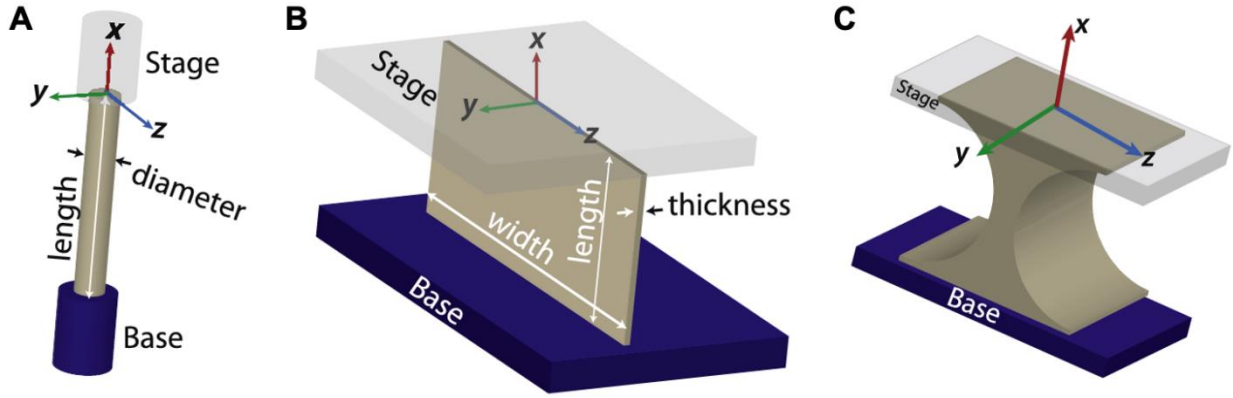

Figure 4-1. Typical flexure examples. From left to right: wire, blade, and notch flexure. Adapted from O. Turkkan et al. [300]

A CM typically comprises slender flexural elements connecting between blocky rigid bodies. The mobilities of any rigid body within the system are primarily determined by the flexures' layout (i.e., position and orientation). Typical flexure types include linear elements that have a circular or square cross-section (i.e., wire or rod flexures, Figure 4-1A), thin sheets of material (i.e., blade flexures, Figure 4-1B), or two solid blocks of material connected by a thin pivotal link (i.e., notch flexures, Figure 4-1C). A flexure's largest dimension is typically 10x or more times larger than its smallest dimension; more drastic aspect ratios (e.g., 50x and up) would make the flexure more susceptible to buckling and failure. Structurally speaking, flexures exhibit minimal deformations when subjected to loads along their longitudinal axis but are more compliant along bending or twisting loads. On the other hand, the rigid bodies' geometries have small, often negligible effects on the mobility of the device given that they are sufficiently rigid (i.e., no slender necks and a relatively cubic aspect ratio).

Despite its advantages, CMs have unique limitations compared to conventional mechanical articulations. They are made as a continuum and afford motions by bending and twisting; therefore, they do not allow continuous motions. During deformations, stress-concentration at corners or flexural interfaces could also lead to structural failure (e.g., crack, plastic deformation) that compromises the device's performance. This issue may be avoided by smoothing (i.e., filleting) the concentration sites. Lastly, structures subjected to cyclic loads, oscillation, or repeated displacement could be susceptible to structural fatigue, where the device could not return to its original resting position and become permanently deformed. Therefore, these factors should be considered when designing CM devices and applications, or they could be a common fallacy for designers and engineers transitioning from conventional mechanical design to CM designs.

To summarize, CMs have their unique pros and cons as a mechanical design paradigm and are not a uniformly superior option over conventional mechanism design leveraging hinges and assembled parts/joints. Both paradigms have their favorable use scenarios and should be used situationally. Their disadvantages could also be addressed by careful engineering within the design criteria. That said, CMs could still be competitive if not preferred in certain situations, and their wide adoption is partly bottlenecked by effective design methods, which we will elaborate on in the following sections.

4.3. History of Compliant Mechanisms and Their Manufacturing

CMs are abundant in both biological and artificial worlds. In nature, trap-jaw ants leverage the compliance of their exoskeleton to store energy and produce a forceful bite [225], Venus flytrap's specialized leaves are essentially CMs powered by hydration and buckle-snapping [70, 308]. In the history of crafts, hunting bows are one of the earliest examples of menial CMs: by pulling the string, the structure deforms and stores energy for a sudden release to thrust an arrow. The bow's cross-section is designed to be thicker on one end and thinner than the other, thereby creating a "blade" flexure that tends to deform on the thinner dimension and controlling the direction and profile of how it bends.

However, for a long time over history, conventional linkages and articulations have been the dominant approach for designing and making articulated structures and devices. These structures

are easier to model and analyze, whereas CMs could have complex and more nonlinear behaviors that are unintuitive to understand and synthesize. Still, since the last two centuries, with the advancement in mechanical understanding, manufacturing, and material science – especially with the discovery of polymers, CMs have emerged as an alternative design approach. Their popularity was further boosted by the invention of micro-electro-mechanical systems (MEMS) [90]: structures were created on micro and smaller scales that render manual assembly impractical, if not impossible, and CMs became perhaps the only viable option in such a context. Designing and creating articulated structures as single pieces using CMs makes it compatible with MEMs fabrication methods, e.g., photo-lithography [192], etching [138], and chemical vapor deposition (CVD) [356] that work at such small scales. As an iconic example, a MEMS electrostatic comb drive [326] uses electrostatic forces between two sets of fins of different polarities to displace a structure.

Still, most CMs are limited to planar, 2D structural designs in MEMS due to the limitations imposed by the fabrication processes. Most manufacturing methods start from a planar substrate to progressively create the desired CM structure. Subtractive methods like etching and lithography have depth limits; they cannot add much thickness on top of the substrate. Similarly, additive methods like CVD can only deposit thin layers (often a fraction of the substrate's thickness) of materials on the surface. Furthermore, free-standing CMs are likely to slack or vibrate due to their inherent compliance. As a result, three-dimensional CMs are more feasible in a mesoscale (i.e., millimeters and larger) where assembly is possible. In this scale, CMs may be made by molding or putting together planarly-made parts. Yet again, the benefits of structural simplicity (reduced labor and cost) are lost while doing so. It is also worth noting that 2D CMs are also less complex to design as bodies in 2D have at most three degrees of freedom (DOF, i.e., in-plane translations about two axes and rotation about the out-of-plane axis), whereas bodies in 3D have up to six DOF (i.e., translation and rotation about each of three axes).

3D CM designs became more advantageous and popular with the blossoming of additive manufacturing methods like fused-deposition modeling (FDM), selective laser sintering (SLS), and stereolithography (SLA). The methods "print" materials at their desired locations and often add support structures to prevent the printed parts from falling or slacking during the process (FDM and SLA). In some cases (e.g., SLS), unsolidified materials could also act as supports to be

removed after the part is completed. Such fabrication processes allow users to create 3D CM structures in one step and unlock their full potential. Still, the effective design of such structures remains a challenge.

Compliant Mechanisms in Engineering

Here, we provide more (non-exhaustive) examples of CM that leveraged their advantages over conventional articulations. Besides MEMS, CMs have found applications across different engineering and science domains, all the way up to meter scales in architectural design [231]. The lightweight and reliability make monolithic CMs an ideal solution for aerospace engineering needs, such as making thruster gimbals [185]. Merriam et al. used SDS-printed titanium to produce a 2-DOF thruster gimbal articulated with compliant joints. The design was also engineered with longevity and a fatigue life cycle magnitudes above the expected usages. In robotics, CMs are also favorable in biomimetic designs as they are more miniaturizable and could operate in wet or harsh environments (e.g., water strider-mimicking robots [124]). Literature has also explored creating planar CMs that could pop up to become a manipulator (e.g., a positioning stage [85], vibration isolator [214]) or an input interface (e.g., a joystick [188]).

CMs have unique advantages in biomedical engineering that conventional mechanical articulations cannot provide. As a prosthetic joint (e.g., knee-replacement [88]), CM's lightweight adds smaller weights to the extremity, leading to more wearer comfort and efficiency. The lack of lubricated joints also means that CMs could be used in the body as implants. In the case of orthopedic corrections [107], a conventionally articulated implant could be damaged or clogged by tissue growth and rapidly degrade due to challenges in maintenance, leading to loosening parts and osteolysis. By contrast, CMs are less susceptible to wear and damage, and the reduced part count and mechanical/structural simplicity make it easier to achieve complex and patient-specific geometries. In the rising field of tissue engineering [23], articulated scaffolds could also benefit muscles grown ex vivo [139]. In this case, the scaffold should be submerged in an aqueous media, and lubrication should be avoided due to biocompatibility and safety concerns, which render conventional mechanical articulations unfeasible.

Compliant Mechanisms in Human-Computer Interaction

HCI researchers have explored compliant mechanisms to create interactive devices with prescribed kinematic behaviors. Their applications include customized action figures [270], mechanical computers [111], or gadgets and toys [182]. Authoring tools like KinetiX [218] and Metamaterial Mechanisms [109, 110] allowed users to construct 2D, morphing lattice grids by connecting material voxels with prescribed transformation behaviors. In the presence of an external load, the cells would move in synchrony and carry out certain motions. On the other hand, there are also design tools focused on the inverse design of compliant mechanisms. That is, given a targeted motion path [110, 182], deformability [270], or DOF [264], the design tool will automatically generate a corresponding compliant mechanism for the user. Furthermore, compliant mechanisms may be combined with transformable robots [129, 202, 285] to create more compact devices that afford versatile kinematic functions or integrated into mechanical metamaterial designs [109, 111]. When used with active materials, CMs may also enable users to design shape-changing interfaces [204, 312, 339, 340] with context-dependent and complex kinematic behaviors.

4.4. Compliant Mechanisms Integrated with Smart Materials

Compliant mechanisms integrated with active materials have become an emerging and rapidly growing area in the scientific and engineering landscape. Compliant mechanisms, on their own, are passive structures or skeletons with prescribed deformation profiles. While CMs are characterized by their continuum and monolithic design that made them easy to manufacture, powered by recent advances in additive manufacturing, fabrication complexity has become more of a solved problem and less of a challenge. Literature began to explore compounding CMs with materials that can sense, actuate, and reconfigure to augment their functions and affordance. For instance, compliant mechanisms made from conductive composites could sense their configuration (i.e., mechanosensory [141]) and function as an input device [82]. Mechanisms integrated with actuatable materials could become active and interact with the surrounding environment (e.g., grippers [294]) or self-propelled robots [124]. Flexures made of shape-memory materials could reconfigure a robot's limb shape and alter its gait on the fly [279] or deploy-package aircraft wings when combined with the bi-stability of CMs [119]. Yet, despite its expanded functions, designing active material integration remains a challenge, especially when multiple joints are needed.

4.5. Compliant Mechanism Design Methods and Tools

Compared to conventional mechanical articulations, compliant mechanisms are difficult to design due to their inherent non-linear behaviors and design spaces. Conventional mechanical systems, such as linkages [267], are relatively easy to predict, model, and synthesize as the motional freedoms and axes are constrained at the pivots. By contrast, flexures in a CM bend and deform to allow motions; therefore, they do not have an apparent pivot point. Slight changes in a flexure's orientation and position could also drastically change the CM's functions and performances, making it unintuitive to iterate a design. Designing CMs could also involve optimization and achieving multiple objectives at the same time (e.g., range of motion, fatigue, stiffness). The interplay between parameters and objectives could easily become difficult to navigate for the human mind, and a systematic design method is needed.

With these challenges in mind, we discuss three mainstream design methods for compliant mechanism design. We note that the methods each have their tradeoffs, and their applicability could differ depending on the use scenario.

Rigid body replacement method

Given a desired function, the rigid body replacement method [181] starts by finding a rigid linkage mechanism that produces the desired function and replaces the linkages or joints with flexures. This approach effectively breaks the design problem into independent sub-problems that contain much fewer parameters. In this design approach, the pseudo rigid body model (PRBM [104]) is often used to model the CM's mechanical performance, and the process is often carried out by hand. For more complex design objectives, such as path-matching [253], numerical simulation and optimization could be deployed to procedurally and iteratively optimize flexure geometries and placements to have a body in the system displace along a desired trajectory [182]. However, as the name suggests, the method requires a rigid body mechanism to inspire the design and does not support ground-up CM synthesis. Some rigid body mechanisms may also require continuous motion at some joints, making CM replacement impossible. However, this design method could effectively convert conventional mechanisms into CMs, reduce part counts, and lower manufacturing costs.

Topology Optimization

Topology optimization (TO) is a common structural design method for creating a device with an optimal performance-to-weight ratio. TO problems are defined by a domain of discretized elements (i.e., often in the form of voxels [71, 145] or structural springs [71, 256]), boundary conditions such as fixed ends, and desired inputs (force) and outputs (displacements). By iteratively simulating the design and removing elements with some heuristics (e.g., removing least loaded elements [244]), the structure would have progressively fewer elements and become lightweight. When applied to CM designs, removing elements is instead used to shape the mechanism's flexures. TO could be a convenient method for CM design as it does not require much but an initial guess of the design domain, and the rest could be computed. Compared to the rigid body replacement method, TO does not require a rigid body reference as input and thus may navigate a broader design space. It could augment designer intelligence: the method or a TO program could take in a partially finished design and return an optimized result.

Still, the quality of a TO solution highly depends on the input setup. A poorly discretized domain could lead to instability and solutions with impractical flexures (e.g., infinitely small point flexures). This issue could be avoided by using higher-order voxel models [344]. The optimization objectives as hyperparameters (e.g., solid volume fraction, stiffness) could drastically impact the viability and appearance of the optimized result, making it difficult for novice users to define a problem appropriately. Moreover, TO algorithms and programs are often computationally expensive as they involve iterative finite element analysis (FEA) that requires solving colossal linear systems. As a result, most TO designs are limited to 2D problems with a magnitude fewer element counts than in 3D. The sheer computational time also makes it difficult to do trial-and-error and iterate toward design objectives.

Freedom and Constraint Topology Method

The freedom and constraint topology (FACT) method [97, 98] leverages screw algebra [280] (see Appendix 1: Rationalizing Compliant Mechanisms) to represent, model, and synthesize the kinematics of CMs. Under this scheme, the two rigid bodies in a compliant joint can displace in certain DOF with respect to each other and are restrained from displacing in their degrees of constraint (DOC). The DOF and DOC are complementary; the DOC is determined by the

topological arrangement of flexures connecting the rigid bodies. Thus, a mathematical mapping between flexural arrangement and the joint's mobility can be described analytically. Screw algebra provides a convenient representation to describe individual DOF or DOC as six-dimensional vectors of "rays" extending from a point in space along an axis. The mobility freedoms and constraints of a joint could be regarded as a linear span (i.e., a parametric combination) of DOF (twist) or DOC (wrench) vectors, respectively. This provides a convenient tool for analyzing and synthesizing CMs. If the flexure layout is known (i.e., as in a design analysis problem), we may calculate their DOC and henceforth DOF to describe its mobilities. Alternatively, if the DOF of a system is known (i.e., as in a design synthesis problem), we could algebraically find its complementary DOC and project the vectors in 3D space to find the flexure arrangements affording the input DOF. In particular, the projected DOC describes the allowed space for placing flexures, which could be created using different "concepts" (i.e., distinct flexure layouts that are kinematically equivalent, Figure 4-2). As a design method, the FACT approach models and navigates the solution space while allowing more design freedom. The FACT method also provides a comprehensive library of flexure topologies between zero to five DOF. For more information, we refer readers to Appendix 1 and the literature. Beyond single joints, the extended FACT method could also analyze topological CMs such as serial [99], parallel [102], or networks of joints [278].

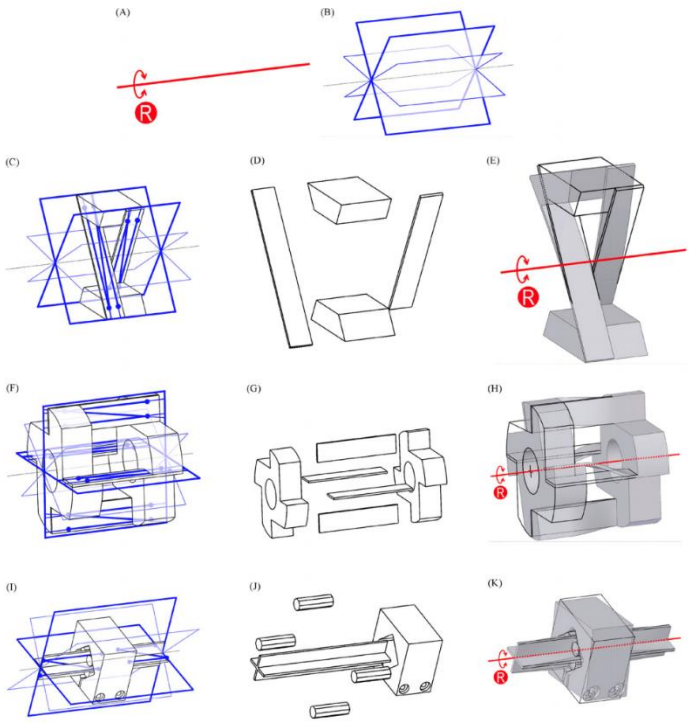

Figure 4-2. An example design process using the FACT method. (A) Given a targeted 1-DOF rotation, we could use the FACT method to find its (B) corresponding constraint space that identifies (C) needed constraining wrenches for (D) placing flexures and (E0. create physical designs achieving the targeted DOF. (F-K) There could be various ways to render the needed constraints using this method, allowing for the generative design of compliant mechanisms. Adapted from J. Hopkins et al. [98]**.**

It is worth noting that the FACT method is a purely kinematic technique. While we could leverage it to analyze and design the kinematics (i.e., DOF and DOC) of a mechanism, it does not concern itself with the elastomechanics of a flexure layout and design metrics like stiffness, range of motion, and path-tracking are not readily compatible with the approach. However, we note that recent advances in this field have explored extending the screw algebraic representation to account for these factors. For instance, Turkkan et al. [300] combined screw algebra, Euler beam theory, and numerical methods to rapidly analyze and predict the stiffness and trajectory of CMs. Literature [91] had also explored combining machine learning and FEA to predict CMs' motion range and natural frequencies.

4.6. Choosing a Design Method

In this thesis, we take inspiration from the FACT method and adapt it to develop a series of design tools and algorithms. It offers the benefits of both methods, allowing users to interactively and freely explore design opportunities and making it an ideal computational engine. Compared with

rigid body replacement methods, both share the same divide-and-conquer strategy of dissecting CMs into rigid and deformable parts, effectively reducing the problem to independent, actionable chunks. However, the FACT method allows ground-up synthesis without a rigid linkage counterpart as input. Compared to TO, the FACT method provides faster speed and navigates a larger design space. While TO is slow to compute, especially in 3D (as a reference, Autodesk's Fusion 360's TO function takes hours to generate a solution), the FACT method could generate and visualize the constraint space in under a second, providing and affording real-time iterations.

Moreover, TO and the rigid body replacement method were invented to find a particular solution for a CM design problem, and they are not inherently built to help users navigate alternative design solutions. By contrast, the FACT method models the solution space that fosters multiple satisficing design solutions. A design tool adopting the FACT method could then help users build toward and navigate this satisficing space.

The representations used by the FACT methods (i.e., rigid bodies and flexural elements) are piecemeal, object-oriented, and more intuitive for the human mind than the voxel-based representation of the TO. A few parameters parameterize each flexure building block. Hence, a design tool could apply or suggest modifications in a human-readable format (i.e., changing a parameter about a building block at a time), making it easier to develop tools for effective communication.

To summarize these differences and motivations, the FACT method supports CM design by modeling the properties a satisficing solution should possess. This information could then guide users in iterating a model to obtain these qualities procedurally. Modeling the qualities of a solution also implies that we could use the FACT method to navigate satisficing design spaces instead of being locked to a singular solution. This interaction modality resonates with the vision of building "tools that help users find solutions" and allows users to reflect and adapt their strategies during hands-on manipulation.

A brief review of the FACT method's mathematical foundation is provided in Appendix 1: Rationalizing Compliant Mechanisms. The review covers both the fundamentals of screw algebra and its adaptation to compliant mechanism freedom and constraint analysis. The following

chapters present a series of design tool developments featuring the FACT method that targeted different challenges in CM design and enabled new design spaces in engineering and HCI domains.

Chapter 5. ReCompFig: Designing Dynamically Reconfigurable Kinematic Devices Using Compliant Mechanisms and Tensioning Cables

5.1. CAD Toolmaking Motivation

ReCompFig [335] targets reconfigurable compliant mechanism design. Compared to SimuLearn, which addressed a static parametric design problem that contains a predefined number of parameters, the design space targeted by this research is more dynamic, where the number of design parameters may change with different objectives, contexts, and upon each design modification. The design landscape is also nonlinear and discontinuous. Drastic changes could sometimes lead to minor to no difference in a device's kinematic behavior. For instance, adding new flexures to a compliant mechanism could have no impact on its kinematic freedom or almost negligible differences in the location of its rotation axis. Conversely, small changes could occasionally lead to entirely different device kinematics. E.g., rotating a flexure by a small amount might acutely increase its resistance against a specific DOF, rendering it inoperable. This challenge makes it inherently and immensely different from other active material design problems (e.g., [6, 290, 313, 315]).

Given the nonlinearity, it is difficult for designers to predict the effects of design modifications and strategize their actions to optimize a design toward targeted kinematic functions. Thus, purely forward design workflows are prone to be ineffective. Inverse design workflows, including precomputation and clustering methods [127, 259], also struggle to capture the vast and diverse design alternatives afforded by compliant mechanisms. Post-rationalization techniques [175, 303] or pre-computation [260] also do not respond well to such noncontinuous, nonlinear problems. Therefore, the goal of toolmaking shifts from "designing to solve problems" to "designing to find solutions."

Given the complexity of CM design problems, now compounded by kinematic reconfigurability, designers may tend to fixate on specific solutions [241] and become less reflective of the design task. In response to this, we draw inspiration from suggestive design interfaces to develop a tool

that guides the user in producing a valid design that meets the targeted kinematic functions. Specifically, we employ a heuristics-based design solver to generate user-readable modeling instructions and error checks while still allowing users the freedom to model the design within the numerical constraints identified by the tool. This way, the tool and the designer could both take on what they do best [65] in a contextualized CM design task.

5.2. Technical Motivation

From creating input devices to rendering tangible information, the field of HCI is interested in using kinematic mechanisms to create human-computer interfaces. Yet, due to fabrication and design challenges, it is often difficult to create kinematic devices that are compact and have multiple reconfigurable motional degrees of freedom depending on the interaction scenarios. In this work, we combine compliant mechanisms with tensioning cables to create dynamically reconfigurable kinematic mechanisms. The devices' kinematics DOF are enabled and determined by the layout of bendable rods. The additional cables function as on-demand motion constraints that can dynamically lock or unlock the mechanism's DOF as they are tightened or loosened. We provide algorithms and a design tool prototype to help users design such kinematic devices. We also demonstrated various HCI use cases, including a kinematic haptic display, a haptic proxy, and a multimodal input device.

5.3. Introduction

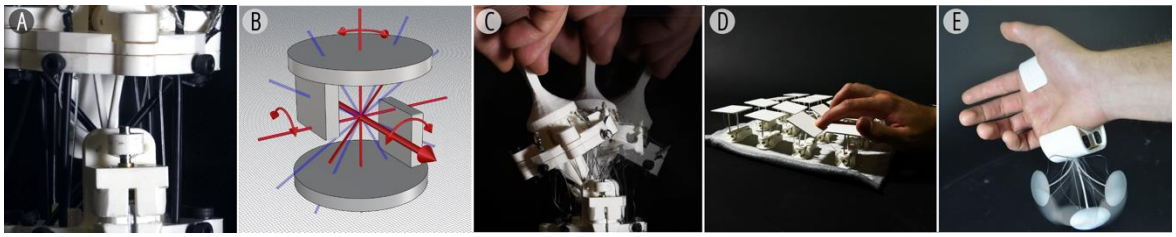

Figure 5-1. ReCompFig overview - (A) compliant mechanisms that have multiple cable-based reconfigurable kinematic degrees of freedom. (B) Design tool prototype and algorithms based on the screw theory are provided to assist users. Application examples including (C) a multimodal input device, (D) a kinematic material display, and (E) a haptic proxy are provided to demonstrate the enabled interaction design space.

The human hand is a versatile instrument that allows us to interact with the environment tangibly and dynamically, and the resulting kinematic experiences make up an essential part of how we perceive the world around us. In HCI, we are also gaining interest in creating devices that afford

these kinematic interactions to support a more natural, intuitive, and enriched interaction experience. Along this line of effort, prior works had explored creating haptic proxies for virtual reality [245], displaying material properties and enabling tangible interactions [67], and designing gadgets and toys [109, 110]. However, as we push against the boundary of kinematic device design, designing multimodal (i.e., having more than one kinematic mode) and reconfigurable kinematic devices still remains a challenge. Specifically, a mechanical joint (e.g., hinge, slider) is needed for each targeted degree of freedom, and their integration may require expert knowledge and skills. The footprints of the mechanical components also impose a size limit, making it difficult to create miniaturized and compact interactive devices. Moreover, adding reconfigurability to the devices also requires latching or locking mechanisms, further exacerbating the design and fabrication complexity. As a result, existing interactive kinematic devices are often highly specialized and unimodal, possessing single kinematic modes, and cannot adapt nor reconfigure to different use scenarios.

We address these challenges by combining compliant mechanisms and tensioning cables to create multimodal and reconfigurable kinematic mechanisms (ReCompFig), as seen in Figure 5-1. Compliant mechanisms can be made of a single material with prescribed DOF and kinematic behaviors by making local parts slender and flexible (i.e., flexural elements) [103]. Compared to conventional joints made of hinges and sliders, CMs are mechanically simple to fabricate as they involve fewer parts and less assembly. A CM can also be designed to offer multiple DOF without increasing its mechanical complexity, making it a viable method for designing compact and multimodal kinematic devices. However, conventional CMs cannot be reconfigured once fabricated. To enable dynamically reconfigurable kinematic behaviors for HCI uses such as different haptic feedback and physical input modes, we further introduce tensioning cables into CMs to reconfigure (i.e., temporarily lock/unlock individual) DOF on the fly, as well as adding stretchable strain sensors to augment their functionalities as an input device.

In this work, we leverage and adopt recent advances in CM design methods [97, 98] to propose a framework to assist HCI researchers and practitioners in designing multimodal and reconfigurable kinematic devices. Our method uses the screw theory to help users achieve two design goals - *prescribing desired DOF modes* and *enabling dynamic DOF tuning*. The designed devices would then afford the DOF modes as specified by the user, and the individual modes can be

enabled/disabled by selectively tightening/loosening the cables. We also developed the method into a computational tool to assist users in designing these kinematic mechanisms (Figure 5-1B). Finally, we present application examples, including a kinematic material display (Figure 5-1D), a miniaturized haptic proxy (Figure 5-1E), and a multimodal input device (Figure 5-1C) to demonstrate the proposed kinematic mechanisms' relevance to the HCI community and the enabled design space.

While both CMs and tensioning cables are not novel concepts in HCI, the combination is. This work provides tools and algorithms to enable their integration. It is worth noting that our method is focused on designing the *kinematics* of the devices (i.e., how can it move), and their *kinetics* (i.e., how much can it move) is omitted and will be a future research opportunity. Therefore, our primary contributions include:

1. Method and principles to design multimodal and reconfigurable cable-driven kinematic compliant mechanisms.
2. Technical artifacts, including design tools and computational algorithms related to multimodal kinematics design.

In addition to the methods and tools, this paper also presents studies and data as a secondary contribution:

3. Evaluation of the designed kinematic mechanisms' viability.
4. Application examples of multimodal and reconfigurable kinematic devices.

5.4. Related Work

Tangible and Kinematic Feedback in HCI

In HCI, we are concerned with recreating different types of haptic feedback to achieve more immersive interaction experiences. Examples of this haptic feedback include weight or volume change [211, 286], forces [200, 246], vibrations [246], and texture [163]. In particular, inForm [67] is an instrumentalized table that uses linearly actuated pins to render shapes [112, 307] or kinematic feedback [67, 148, 203] on demand in a range of interaction contexts (e.g., teleconferencing, video games, and composing). ReCompFig builds on top of these works and navigates a larger kinematic

interaction design space (i.e., enabling linear and rotational motions in all directions). The DOF reconfigurability also enables us to recreate the haptic sensation of different materials (e.g., liquids, elastic rods, rigid objects) using a single device, making it possible to make compact and portable devices.

Haptic Proxies in Virtual Reality

On the other hand, rendering haptic feedback is also a central topic in virtual or augmented reality. In addition to visual and aural cues, prior works had also explored simulating shapes [63], weight [161], or using a combination of both to imitate the hand feel of grasping different objects [45, 130, 133, 265, 348]. Noticeably, recent developments in this area [245, 297] also started to explore using deformable materials and structures to create force feedback. ElaStick [245], for one, is a reconfigurable device that uses rubber bands and cables to simulate the haptic feedback of swinging a rigid or elastic stick. Yet, as the authors pointed out, the device relied on a pre-designed mechanical structure and had limited degrees of freedom. The mechanical structure based on linkage systems and motors also led to an increased weight and size. By contrast, the motions of ReCompFig devices are solely enabled by the structure's flexibility and may enable us to create more compact and lightweight devices. Our design tool and framework also support users to customize devices with the desired DOF as well as adding sensing capabilities, further expanding the haptic proxy design space.

5.5. Design Principles

Expanding Flexure Design Space Using Cables

ReCompFig is composed of three basic elements: rod flexure, tensioning cable for mechanical reconfigurability, and elastic cables for sensing.

Rod flexure. In CMs, a rod flexure typically has an aspect ratio between 1:10 to 1:20 and has buckling strengths well above the expected use load and negligible tensile extensions in normal situations, meaning they cannot buckle nor extend under normal use scenarios, therefore limiting linear motions along its longitudinal axis (Figure 5-2A).

Tensioning cables. Tensioning cables also hold similar structural properties: they have negligible extensions when subjected to tensile loads. Yet, unlike rod flexures, cables would buckle when subjected to compression forces due to their extreme aspect ratio (typically larger than 1:40) and low buckling strength, therefore making them "half rods" in a structural sense, i.e., cables behave like rods under tensile loads but are virtually ineffective under compression (Figure 5-2B). Nonetheless, we can still simulate a complete rod flexure by placing a pair of cables on both sides of the motional axis, such that when one buckles under compression, the other cable in the pair still withstands the forces in tension. Based on this property, we leverage cables as an on-demand, reconfigurable flexure element to design CMs. When controlled by a motor, the cable pair can be further shifted in or out of effect when tightened or loosened, respectively, thus enabling reconfiguration for different interaction scenarios without modifying its structure (Figure 5-2D).

Elastic sensing cables. On the other hand, cables made of elastic materials hold no structural functions in a CM because their elasticity allows them to stretch under tension and thus cannot be used as flexures in any way. However, we can take advantage of this fact and add stretchable sensors (e.g., conductive rubber cord stretch sensors) to detect its deformations without modifying its DOF (Figure 5-2C). This addition allows us to identify how the kinematic mechanism is being interacted with and use it as an input device.

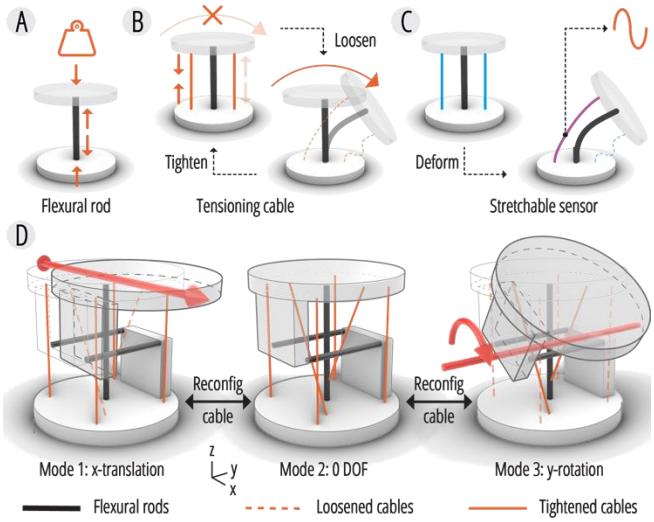

Figure 5-2. Design principles of reconfigurable, sensing, and multimodal kinematic devices. (A) A flexural rod constrains DOF by resisting compression and extension, whereas (B) a cable only constrains extension when tightened, and (C) an elastic cable (sensor) imposes no constraint. (D) The tensioning cables can be tightened/loosened to reconfigure the device's degree of freedom.

Designing Multimodal Kinematic Mechanisms

The cable-driven reconfiguration can be used to create devices that have dynamic, different kinematic modes and affordances. These devices can be analytically designed using the screw theory and a Venn diagram of constraint/freedom spaces in several steps (Figure 5-16). First, given the desired DOF for each kinematic mode, we can identify their corresponding constraint spaces. These constraint spaces can then be used to determine the placement of the non-reconfigurable rods - the flexural rods shared by all kinematic modes. At this point, the mechanism should have the combined DOF of all kinematic modes. Next, tensioning cables are added for each kinematic mode to constrain undesired DOF, and a control scheme (i.e., a table that matches cable group status with kinematic modes) is generated. Stretchable sensors are also optionally added to the device following the tensioning cables. Finally, the model is completed by assigning a thickness to the rods and designing guiding tubes for cables and housings for motors.

In the following section, we will demonstrate the design process of a multimodal input device by following these steps. The process is assisted by a design tool prototype and uses the screw theory and the FACT method as a back-end engine. The tool takes kinematic mode specifications as input and provides prompts and visualizations that guide users through the design steps. Note that instead of generating a design solution for the user, the tool suggests where the flexures and cables can be placed, and the user should place the elements based on the prompts. This modality of assistance allows the user to make informed design decisions while leaving room for creativity and freedom to achieve different design goals (e.g., aesthetics and functional considerations). The algorithms behind the design tool and physical implementation are also explained in later sections.

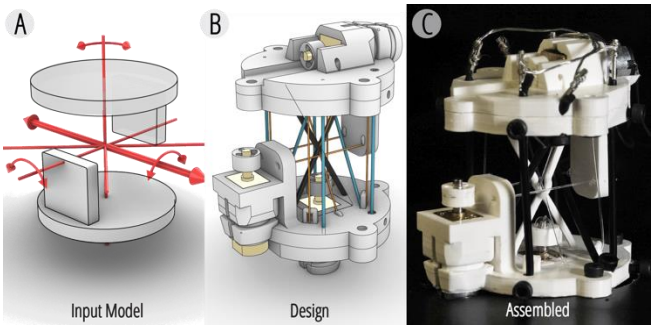

Figure 5-3. A multimodal input device designed using the proposed framework - (A) the design goals and (B) the resulting model, and (C) the assembled device.

5.6. ReCompFig Walkthrough: Multimodal Input Device

In this section, we describe the user workflow through the design of a multimodal input device (Figure 5-3). Figure 5-4 through Figure 5-10 show the design tool and process as seen by the user with slight adjustments to colors and fonts to promote readability. The device has three kinematic modes - slider, joystick, and dial knob - that provide different interaction affordances, and the goal is to produce all these functionalities within a single CM joint. The algorithms are implemented in Python and the design tool in Rhinoceros 3D with plugins (Grasshopper, Human UI, and CPython). The three enabled interaction modes are described in later Figure 5-11.

Step 1: Assign kinematic mode

The workflow starts with modeling the two rigid stages - a fixed and a free end - of the kinematic device (Figure 5-3A), which were left plain at the beginning since the cables, motor housings, and flexural rods will be added in later steps. The user starts by creating three DOF modes for this device (Figure 5-4). The first mode approximates a joystick, which allows for rotations about any axis on the x-y plane that passes through the center of the mechanism. To represent these freedoms, the user adds several possible rotation axes to the design tool, which in return visualizes the freedom space for the user (Figure 5-5A). Similarly, the user assigns a z-axis rotation to the second kinematic mode to simulate a dial knob and a translational motion along the x-axis to the third mode as a slider (Figure 5-5B).

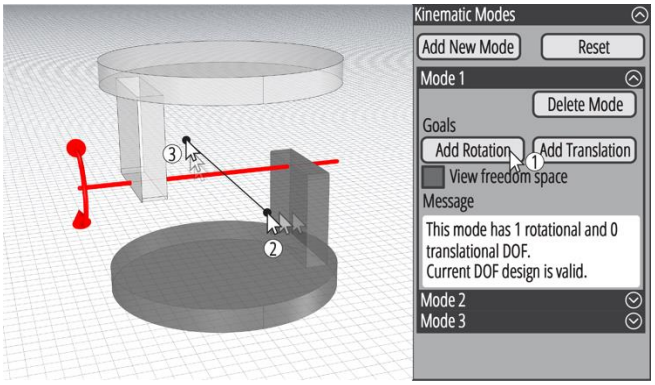

Figure 5-4. After importing the input model, the user begins the design task by adding kinematic modes and specifying the DOF of each mode.

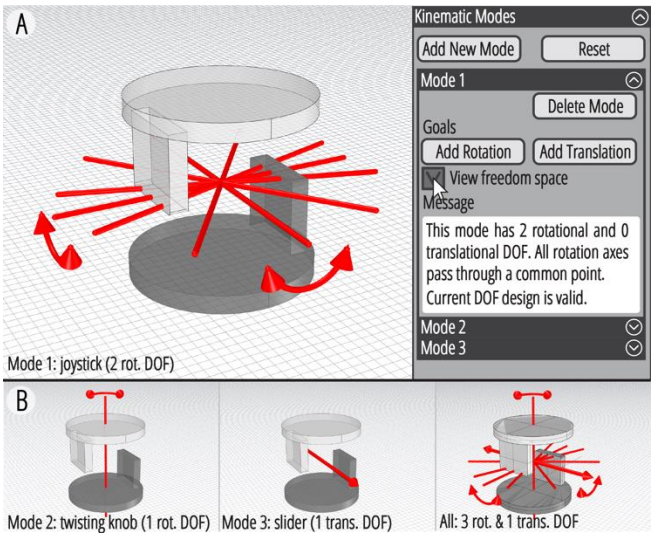

Figure 5-5. (A) Once the DOF are assigned, the design tool provides visual and textual prompts to inform the user of the motional freedom space. (B) The DOF goals added for each kinematic mode.

Step 2: Place Flexural Rods

Once the user specifies the DOF of all kinematic modes, the design tool then proceeds to guide the user through designing the flexural rods. These rods are shared by all kinematic modes and are not reconfigurable. The constraint space visualization suggests the location and orientation of permitted rod placements, and the user can take this information as well as the textual descriptions to place the flexures iteratively until the constraint spaces are satisfied and complete (Figure 5-6). In this case, the tool prompts the user to add at least two rods that pass through the center point and lie on the y-z plane. If the user places a rod in an invalid position or orientation, the CM will be over- and ill-constrained, and the design tool will highlight and prompt the user to correct it (Figure 5-7).

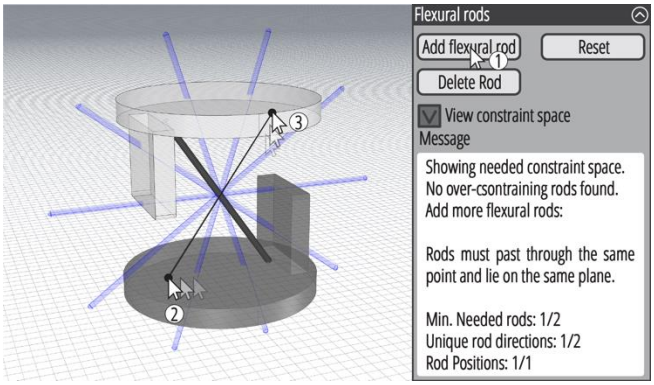

Figure 5-6. Once the desired DOF is specified, the design tool then prompts the user to add flexural rods that are shared by all kinematic modes.

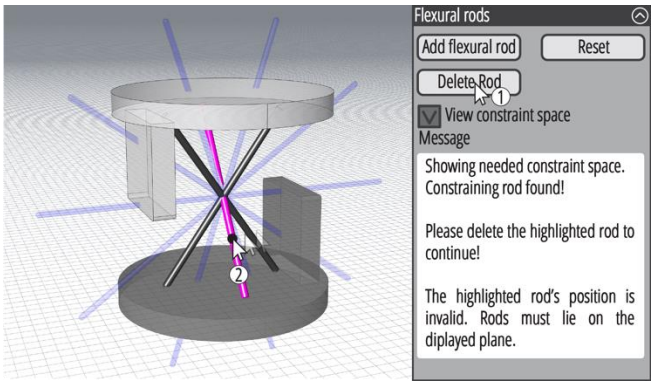

Figure 5-7. If the user adds a flexural rod in an invalid placement, the tool will highlight the rod and inform the user to remove it.

Step 3: Place Tensioning Wires

Following placing the flexural rods, the design tool guides the user to place tensioning cables that reconfigure the kinematic device (Figure 5-8). The tensioning cables are assigned into groups that are actuated together, producing a reconfiguration plan for the kinematic modes. When adding the cables, unlike flexural rod placements, the user has a lot more freedom to decide where to place the tensioning cables as long as their directions follow the design rules. Nonetheless, since the cables must be added in pairs to balance their loads, the design tool will generate the coerced twin cable whenever the user places one. The tool also prompts the user when the cable placement is invalid.

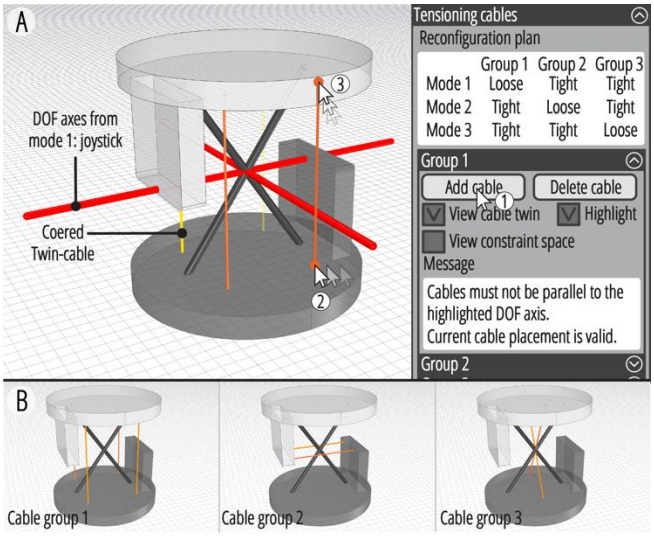

Figure 5-8. (A) The design tool provides visual and textual prompts to inform the user of valid tensioning cable placements. (B) The cable groups added for this design.

Step 4: Place Sensors (Optional)

The user has the option to add stretchable sensors to the CM to convert it into an input device. To do this, the design tool also provides guidance to help the user place the sensors in appropriate orientations. The rules and textual prompts are similar to that of the tensioning cables. However, the design tool provides a panel to preview which DOF's motion is detectable given the current setup (Figure 5-9). The design tool also tells the user which sensors may be stretched along a DOF, which informs the circuit logic and controller firmware design. Figure 5-9 shows the user adding three pairs of sensors to detect the deformations of each kinematic mode.

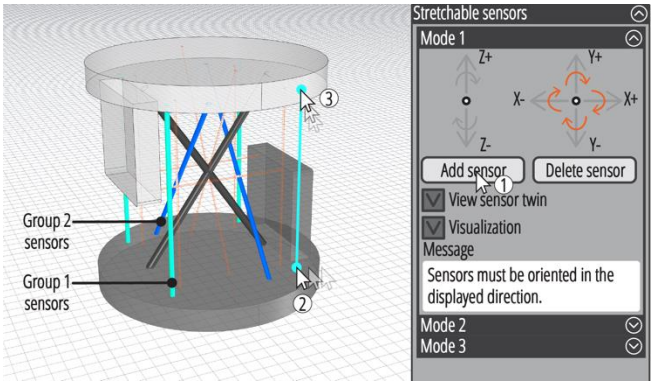

Figure 5-9. The user has the option to add stretchable sensors to the kinematic device, and the design tool checks if a DOF can be sensed given the current sensor layout.

Step 5: Model Flexure Elements and Rigid Stages

Once the kinematic mechanism is completed, the user should proceed to finalize the model by adding mechanical components and assigning thicknesses to the flexural rods (Figure 5-10). To ensure sufficient deformability, the rods have a diameter of 2 mm, which is approximately 5% of their length. The user has the freedom to modify the rigid stages into any shape given they are sufficiently stiff. In this design, the user added guiding tubes and anchors for the cables, housings for the geared motors, and insertion holes. The mechanism was also divided into four parts for 3D printing.

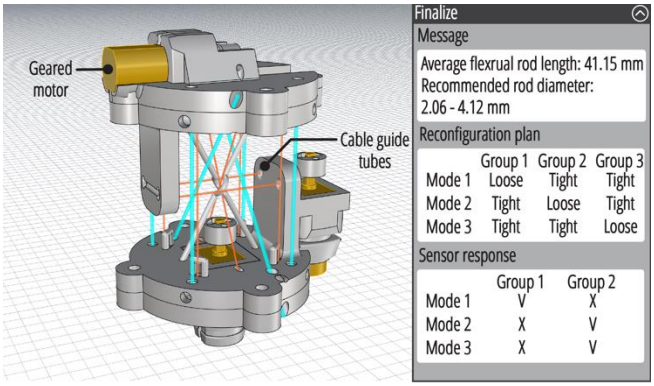

Figure 5-10. Once a design is completed, the user proceeds to finalize the design by adding mechanical details and modularizing the model for printing.

5.7. Application Examples

Multimodal Input Device

Graphical user interfaces on a screen can change from one mode to the other in split seconds. E.g., a slider can change into a knob with virtually no delay. However, unlike their digital counterparts, physical input devices like keyboards, mice, and joysticks only have a single input mode, and users are often required to switch between them for different interaction scenarios. Here, we demonstrate that as a multimodal interface, a ReCompFig input device can support various kinematic interactions at a time (Figure 5-11). The device can change between three kinematic modes on demand, each recreating the haptic experience of using a common interface (i.e., joystick, slider, twisting knob).

Three sets of cables were used in this design, and it has six stretchable sensors to detect the user's action. The reconfiguration takes less than a second, achieving almost real-time modality change. However, we report that qualitatively speaking, based on our user experience, while the device is robust and supports large motion ranges (e.g., >45° of rotation), it does require a relatively large amount of force to activate compared to conventional input devices. Future work may consider optimizing the flexure dimension or choosing a more deformable flexure rod material to achieve a more comfortable interaction. On the other hand, we also speculate that taking advantage of the device's mechanical simplicity, it might be possible to further miniaturize the design to embed it into commercial products (e.g., video game controllers) and enable more dynamic interaction experiences.

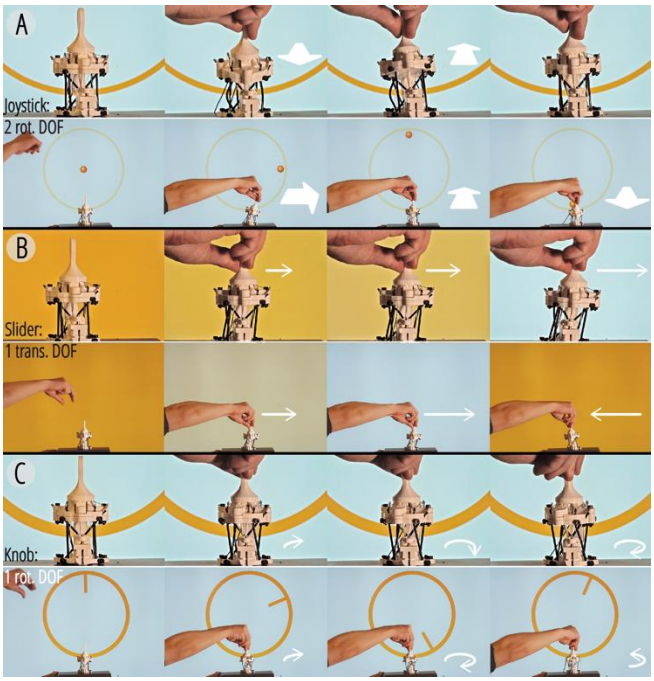

Figure 5-11. The multimodal input device functions as (A) a joystick to move a ball on the screen in the background, (B) a slider control the color on the screen, and (C) a twisting knob to rotate a digital dial.

Kinematic Material Display

In this example, we leverage the compactness and reconfigurability of ReCompFig mechanisms to create a kinematic material device. Different from conventional displays that render images or shape displays that physicalize geometries, this kinematic display is used to tangibilize the kinematic freedoms of a piece of material. I.e., an object's deformability when touched by hands. The display is made of a 4x4 grid of individually addressable kinematic bits (Figure 5-12A) and the modules have a pair of cables to enable/disable their translational and rotational degree of freedom (Figure 5-12B). Figure 5-13 shows the device in action: when the cables are loosened, the interface has the kinematic affordance of mud, whereas when tightened, the display simulates stiff dried soil.

In this application example, we report that qualitatively speaking, designing kinematic devices using ReCompFig brings about several advantages. Compared to the inForm table [67] that provides kinematic response by actuating linear pins, our device affords more kinematic motions (bending, twisting) while having a smaller footprint. The display can even wrap around nonplanar

surfaces when sewn onto a fabric substrate (Figure 5-12D). Future work may also consider incorporating actuatable tendons to further instrumentalize this design.

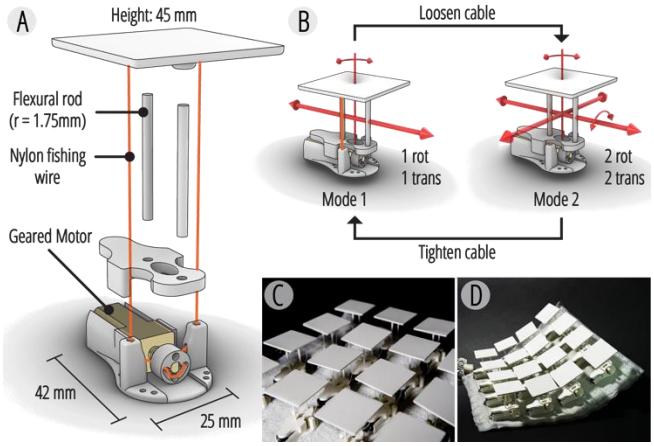

Figure 5-12. Kinematic material display - (A) unit module design and (B) DOF before and after cable tightening. (C) Arrays of kinematic modules as an array-like display. (D) The display conforming to a curved surface.

Figure 5-13. Interacting with the kinematic material display - the display simulating (A) mud with four DOF and (B) stiff dried soil with 2 DOF.

Wearable Haptic Proxy

We demonstrate that ReCompFig mechanisms can also be used to create kinematic haptic proxies that simulate the hand feel of holding objects made of different materials (Figure 5-14A), similar to that of ElaStick [245]. The device is attached with a weight of 80 grams, which is allowed to swing in different directions depending on the tensioning cable configuration (Figure 5-14B). Specifically, the weight is immobile when all cables are tightened and is free to rotate or translate along five DOF when fully unlocked. This design allows us to simulate the weight shift of holding

different objects, including liquids, elastic sticks, and rigid bars (Figure 5-15A, B, and C, respectively). It is worth noting that the mechanical simplicity of our design led to a substantial reduction of assembly demand and device weight: it consists of only three printed parts, two mini-motors, and weighs 101.3 g, whereas the device was designed with more than four parts, four motors, and weighs 814.4 g in ElaStick [245]. For this reason, we hypothesize that ReCompFig can enable us to create lightweight haptic proxies. Moreover, the device's design frees the user's fingers to interact with other possible interfaces (e.g., texture, vibration), which makes room for a more immersive experience. While not explored in this work, we believe this design can also be integrated with weight-changing interfaces [211] to become a more versatile and immersive haptic proxy in virtual or mixed reality.

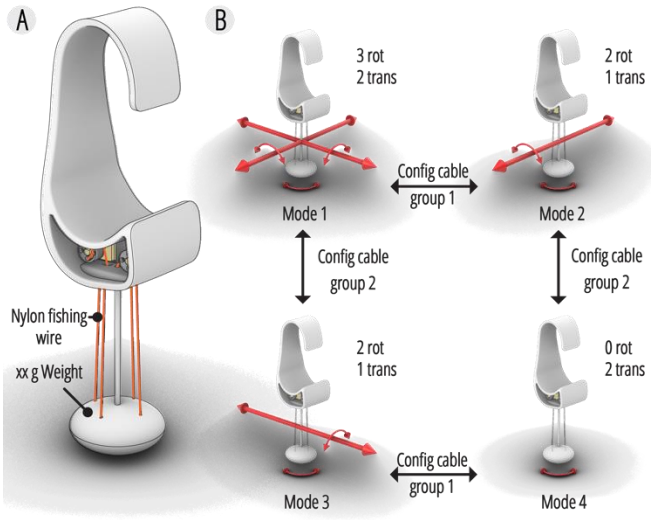

Figure 5-14. Kinematic haptic proxy - (A) its structure and (B) kinematic modes controlled by a pair of motors.

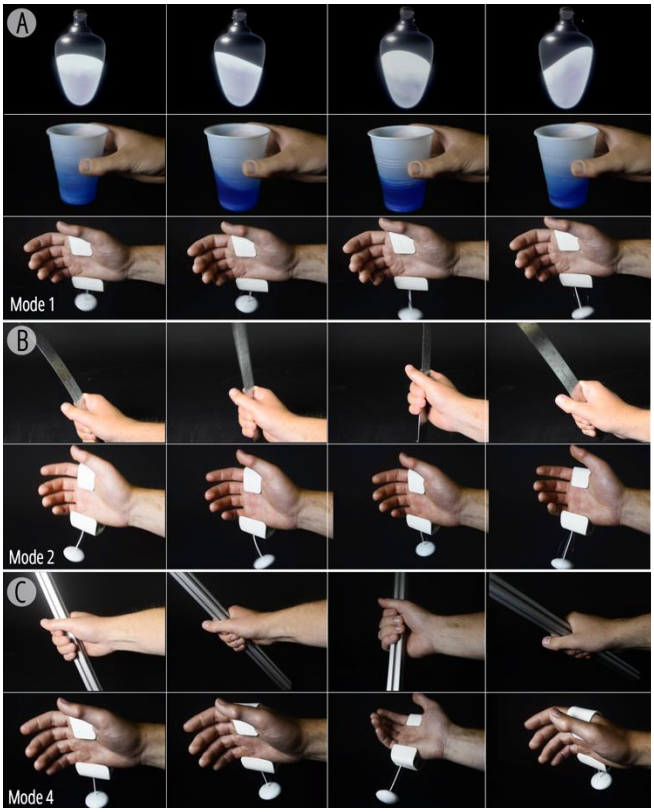

Figure 5-15. Simulating the haptic feedback of (A) liquids, (B) elastic sticks, and (C) rigid bars with the kinematic haptic proxy.

5.8. Algorithm

Each kinematic mode's constraint and freedom space can be calculated based on the motion axis specified by the user in step 1. The line itself creates a directional vector, and either of the endpoints can be used as a reference point on the axis. Plugging these components into eq. 9-1 yields a twist vector of the desired motion. However, since the motions assigned under a kinematic mode may contain linear redundancies that may complicate the calculation, it is important to simplify the freedom space prior to computing its complementary constraint space. Given a matrix $[T_i{}']$ that collects all twist vectors under a kinematic mode i, its non-redundant freedom space $[T_i]$ can be found by finding the kernel of its nullspace. The outcome $[T_i]$ can then be used with (eq. 9-7) to compute the kinematic mode's constraint space $[W_i]$.

Shared Flexure Placements

Given a collection of constraint spaces associated with each kinematic mode, the shared flexure placements [W_{shared}] can be identified by finding their intersection (Figure 5-16A), i.e., the wrench vectors shared by all modal constraint spaces. Given a matrix [T_{all}] that contains all twist vectors under all kinematic modes, the intersection can be found by finding the nullspace of [T_{all}].

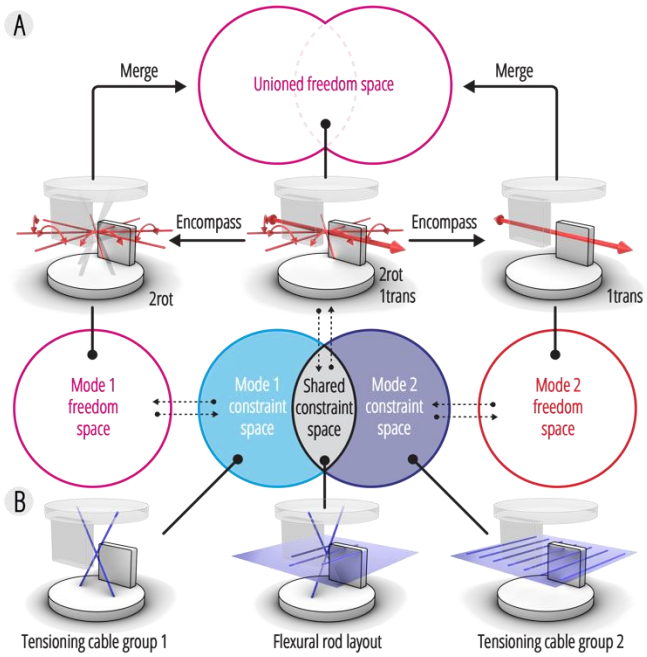

Figure 5-16. (A) Computing shared flexural rod constraint space - the shared rod placement is computed by intersecting the constraint spaces of all kinematic modes. The dashed lines with arrows denote the algebraic mapping as described in Eq. 3. (B) The rods and tensioning cables' constraint subspaces are also computed based on the Venn diagram using linear subspace intersection and difference.

Detecting Over-Constraining Elements

If the current design's axial or positional component's rank is higher than that of the target, then the system is considered over-constraining, and the tool should prompt the user to reduce the degree of constraints by removing the respective flexural elements. Over-constraining flexural rods can be identified by checking whether they are linearly spanned by the target constraint space (i.e., a subset or subspace). Similarly, if the current design's rank is lower than that of the target, then the system is under-constrained, and the tool will prompt the user to add more flexure.

Tensioning Cable

The tensioning wire placement [W_{cable}] for a kinematic mode [W_i] can be found by finding the difference between the modal constraint space [W_i] and the shared flexures [W_{shared}] (Figure 5-16B). This can be done by finding the kernel of the concatenation of [W_{shared}] and [W_i]'s nullspace. It is worth noting that some combinations of [W_{shared}] and [W_i] may lead to a [W_{cable}] with all-zero directional components, which, geometrically speaking, results in invalid rod placements (i.e., axis-less rods). In this case, we can take the directional components from [W_{shared}] and/or [W_i] to generate a valid constraint space. A cable's coerced twin in the pair can be found by rotating its wrench vector 180 degrees around the motional axis they are constraining (Figure 5-17A, B). For translational constraint cables, their directions must not be perpendicular to the translational axis, whereas for rotational constraints, the cables must not be parallel to the rotational axis nor pass through the rotational axis. Additionally, after rotation, the corresponding endpoints in the cable pairs must land on different rigid bodies, which makes sure the cables experience opposite axial forces under external loads (Figure 5-17C).

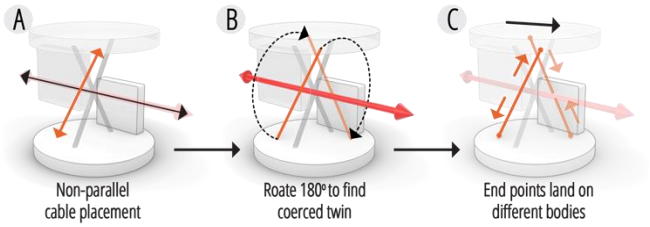

Figure 5-17. Tensioning cable pair placement of a rotational joint. (A) The cable must not be parallel to the rotational axis. (B) The coerced twin of a cable can be found by rotating the cable 180 degrees about the rotation axis, but (C) the corresponding endpoints must land on a different rigid body, which leads to opposite loads when deformed.

The tightening and loosening of cable groups can be determined by checking their corresponding constraint space with that of a kinematic mode. If the cable group's constraint space is a subset of a kinematic mode, then the cables should be tightened to enable and exactly constrain the said kinematic mode. Conversely, if the cable group is not a subset of the kinematic mode's constraint space, they would overconstrain the system and should thus be loosened.

Stretchable Sensor Placements

The stretchable sensors follow the same placement rules, but they would not have any constraining effect due to their extensibility. To determine whether a sensor is responsive to a DOF, we can move the free end of the cable by a finite amount along the motional direction. If the distance

between the two ends increases after the finite movement, then the sensor will be stretched under the respective motion and produce a signal.

Textual Prompts

The textual prompts guide users to design flexure placements that satisfy and complete the freedom and constraint spaces. This can be done by comparing the current design with the targeted constraint spaces. The directional and positional components of the constraint space can be evaluated individually, and each is considered complete when the current design's respective screw vector parts have the same rank as the targeted constraint space. Finally, the minimally needed and existing number of unique flexural rods can be calculated as the rank of the non-redundant constraint spaces.

5.9. Physical Implementation

Flexure Dimension

We acknowledge that there is currently no analytical way to design flexure dimensions without using numerical methods like finite element analysis, which is computationally expensive and may render the design process less interactive. Moreover, different application scenarios may also have different structural demands. For this reason, we recommend users iteratively find out the ideal flexural dimensions through physical prototyping and performance evaluation (see Validation section). As a rule of thumb, the flexural rods are recommended to have an aspect ratio between 0.1 and 0.05. Larger aspect ratios may cause the rod to become less compliant, whereas smaller ratios may make them more susceptible to buckling but more deformable.

Fabrication Method

In this work, we use a desktop 3D printer (Ultimaker S5) and off-the-shelf filaments (Ultimaker PLA) to fabricate our prototypes. The flexural rods and rigid stages were printed as separate parts and assembled to form the work prototypes (Figure 5-18A, Figure 5-3C). Yet, generally speaking, CMs can be made using any fabrication method and material given adequate resolution and structural properties (e.g., resilience, stiffness, deformability), and other additive manufacturing methods like laser sintering (metal or plastics) or digital light processing (resin) can also be used

to fabricate CMs, potentially as a single piece and further reducing the assembly labor. Nylon fishing lines were used for the tensioning cables.

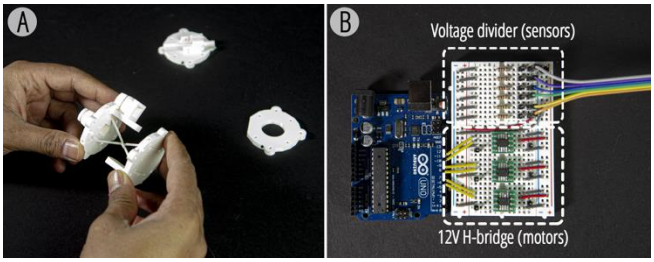

Figure 5-18. The fabrication and control method using the multimodal input device as an example. (A) the kinematic devices are printed in parts and assembled together. (B) The electronics and circuits used to operate the devices.

Control and Sensing

The cables are modulated using geared motors (ROB-12285, SparkFun Electronics) and winches (Figure 5-18A), which are controlled by an Arduino board and an H-bridge (BD62130AEFJ-E2, Rohm Semiconductor). Our geared motors have a gear ratio of 298:1, and due to their high reduction ratio, they only require power during reconfiguration, thus making them power-efficient when idle. We calibrate the cables manually and procedurally tighten them until the corresponding DOF becomes sufficiently stiff. During operations, the cables can be modulated by running the motors in the corresponding directions for a set amount of time and travel (i.e., open-loop control). In our case, it takes 0.5 seconds for the motors to reconfigure the cables.

The stretch sensors (product Id: 519) are purchased from Adafruit and have a resistance of 350 ohms per inch and a maximum strain of 70%. The sensor's resistance increases as a function of its strain and can be mapped to a CM's deformation (Figure 5-18B). Yet, we take a digital approach and detect deformations by checking the resistance deviation from the relaxed state. I.e., a signal is detected if the resistance change exceeds a certain threshold.

5.10. Validation

Testing Setup

We evaluate the effectiveness of our proposed framework in terms of prescribing DOF and the cable-driven reconfiguration. Five samples were designed using our design tool and methods. The passive CMs (i.e., A1, B1, and C1) in Figure 5-19A are used to verify the desired DOF, while the

CMs with cables (i.e., A2 and B2) are tested for their reconfigurability (Figure 5-20A, Figure 5-21A). We use a 3D-printed jig to adapt our CM prototypes to the testing system (Instron 5969), as shown in Figure 5-20C and Figure 5-21C. The samples are fixed on a slider on one end, and the loads are applied to the other. The slider ensures the point of force application always stays lined up with the machine. A maximum tensile load of 5 N is set for all tests, and the loads were gradually applied at a rate of 5 mm/minute. The test system measures the samples' deformations as extension over load. For samples with rotational DOF, the extensions can be trigonometrically converted into bending angles by plugging in the distance between the load application point and the rotational axis, which is part of the sample's geometry. The tests were repeated five times, and the plots were produced by averaging the results.

Designing Desired DOF

The test samples A1 and B2 are designed to validate that the framework can produce the targeted rotational and translational DOF: they are designed with a rotation about the x-axis and a translation along the x-axis, respectively. The mobilities along all six DOF were evaluated for both samples and can be examined by their deformation over load, i.e., the slope of the curves. A steep curve suggests that the mechanism is less mobile along that direction, whereas a more gradual curve indicates freedom. In Figure 5-19B, C, we can see that both samples are much more compliant along their prescribed motional axes. Compared to their constrained motions, both of the samples had a 5-6x higher deformation under the same load along its DOF. Specifically, A1 had more than 30 degrees of rotation about its mobile x-axis, whereas the same deformation was almost indiscernible (smaller than 6 degrees) in other directions. Similarly, B1 translated more than 6 mm along its mobile axis, while the deformation was less than 0.5 mm in the other directions.

On the other hand, sample C1 is designed with a rotational DOF along the z-axis and a translation DOF along the x-axis, and it was used to validate that the design tool can produce a single mechanism with multiple DOF. Based on the results, we can see that the sample is much more mobile along the two prescribed DOF compared to the others, thus proving that the sample was successfully made mobile in the two targeted DOF and is constrained in the other four. These results show that the proposed framework and design tool can indeed lead to kinematic mechanisms with the desired DOF.

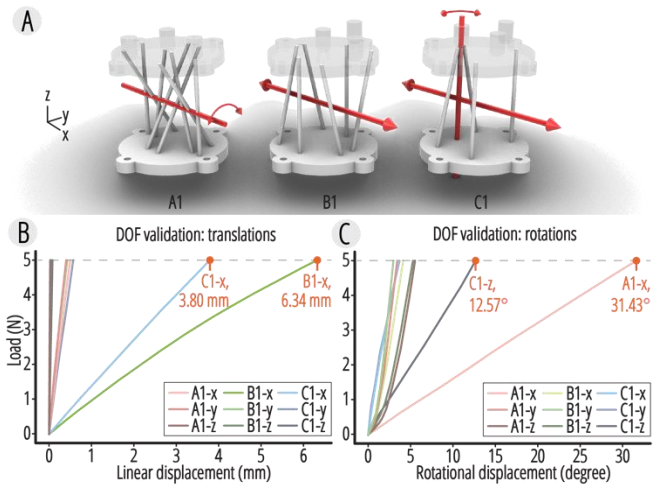

Figure 5-19. DOF design validation - (A) test samples and results for (B) translational and (C) rotational DOF.

Reconfiguration

The test samples A2 (Figure 5-20A, C) and B2 (Figure 5-21A and C) are designed with the same DOF and flexural rod layout as their counterparts in the earlier experiments, A1 and B2, respectively. However, their DOF can be enabled or disabled by loosening or tightening the cables, which were placed per the design tool's suggestions. Figure 5-20B shows that under the same load (5N), the maximum rotational angle of A2 was 31.43 degrees when it was unlocked, but the mobility dropped to 2.89 degrees when the cables were tightened. B2 also showed a similar trend: under the same load, B2's translation along the x-axis dropped from 6.34 mm to 0.49 mm when it was locked (Figure 5-21B). These results indicate that the cable-driven reconfiguration was able to modulate the DOF's stiffness by more than a magnitude, thus showing that the design tool and method produced truthful designs.

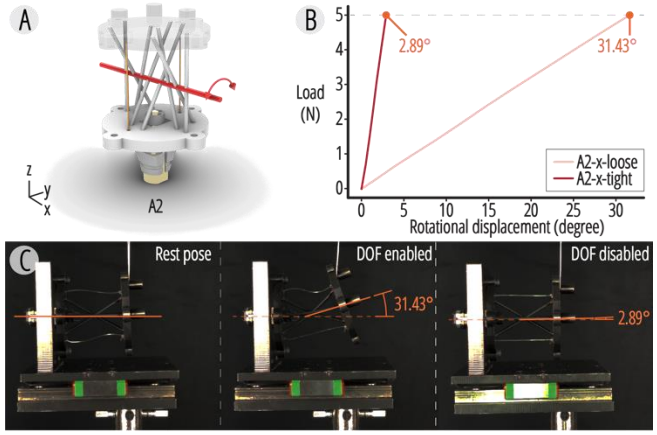

Figure 5-20. Rotational DOF reconfiguration test - (A) the test sample, (B) results, and (C) comparisons.

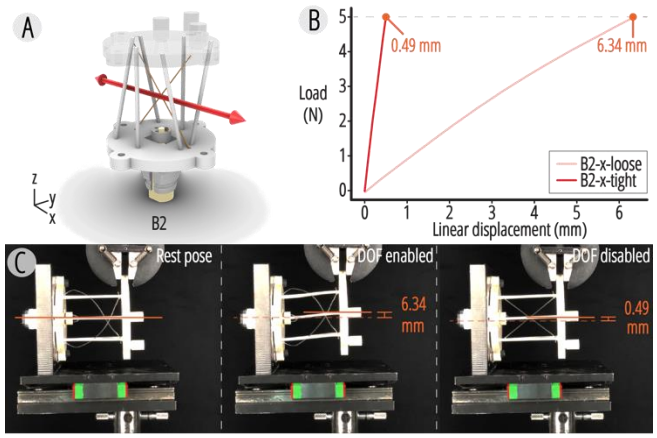

Figure 5-21. Translational DOF reconfiguration test - (A) the test sample, (B) results, and (C) comparisons.

5.11. Discussion and Limitation

Single Joint Design

Although our evaluations and design examples show that the proposed framework can enable and assist users in designing a wide range of reconfigurable and multimodal kinematic mechanisms, certain types of kinematic joints are still unproducible. For instance, it is mechanically impossible to create a single CM joint that affords translation along all three principal axes [97, 98]. However, it is still possible to produce such a joint by serially connecting two or more joints. I.e., a joint is valid as long as it has less than three translational DOF, and the free end of a serial kinematic joint will have the combined DOF of all joints.

Mechanical Design Considerations

On the other hand, compliant mechanisms also have certain mechanical limitations. Since the DOF is enabled by the flexibility of flexures, it is impossible to create devices that allow continuous rotation. Translational CMs are also accompanied by parasitic displacements - as the free end translates in one direction, it will also creep in a perpendicular direction. That said, we did not find this to be a major concern when developing the demonstrations, and users can add an orthogonal translational joint to compensate for this unwanted motion if needed. Structural fatigue has also been identified as a potential pitfall of CMs [185], but we did not observe this to be an issue even after months of repeated usage.

It is worth noting that CM devices are inherently deformable even along the constrained DOF as observed in Figure 5-20B and Figure 5-21B. This slight mobility is due to the materials' inherent deformability, and the tests represent an extreme case where all loads were concentrated along the constrained DOF. In reality, the stiffnesses in these directions are magnitudes higher than that of the DOF. Thus, the device would mostly move along the DOF when subjected to an external load, and the displacements in the constrained DOF are often negligible. That said, the undesired mobilities can be further attenuated by adding more flexures to increase the joint's stiffness, but this property can only be captured by a physically-based simulator.

Device Kinetics Design

The framework focuses on designing the devices' DOF behaviors, that is, kinematically speaking, the derivative of motions. We acknowledge that this is both an advantage and a limitation. Designing devices through their DOF provides both force and dynamic responses to the user [300]. In this work, adding the reconfigurable and multimodal dimension navigates an even larger design space. However, the current state of the design tool and framework does not support other aspects of a device design, such as their kinetics - stiffness, motion range, and trajectory [110, 182]. Designing these properties calls for an interactive and physically-based simulator, and future work may consider adopting advanced or accelerated numerical simulations [300, 337] to provide these functions.

Device Scalability

Practically speaking, ReCompFig works best for a hand- to body-scale. While the design principles and algorithms are applicable across scales, the only limitation comes from the material and components. There is a tradeoff between the joint's stiffness and weight-carrying capacity when scaling the designs. Based on the Euler-Bernoulli beam theory, as a device scales up, the weight increases cubically while its load-carrying capacity increases quadratically, creating a need to thicken the flexures. Yet, a thicker flexure is structurally more vulnerable to deformations (scales linearly with the thickness). Therefore, users are advised to instead add more flexures while keeping the thickness constant or source for a stiffer material to circumvent the need to thicken the flexures. On the other hand, when scaling down the designs, the cables and motors were the

bottlenecking factor as they are difficult to miniaturize and assemble. In this case, a novel method or material system is required.

Computational Interactivity

ReCompFig's design tool supports the design of kinematically reconfigurable CMs through a suggestive design workflow. The tool takes kinematic modes as input and guides the user to advance toward achieving the desired functions using textual and visual prompts. The prompts are procedurally generated at each design modification, informing the user what is needed to complete the design and how to add flexures to satisfy the required layout. Should the user stray away from the design objectives and use invalid flexure placements, the tool also prompts the user to delete them. The step-wise suggestions are also smaller in size, making them easier for users to understand and follow through.

The ReCompFig tool supports forward processes embedded in an inverse design workflow. The tool acknowledges the user's intentions and navigates the user toward their goal while also preventing the user from straying away. Powered by the FACT method's algorithmic simplicity, feedback and prompts could be computed in real time to provide just-in-time feedback. We intentionally designed the tool not to generate flexure layouts for the user but instead to provide a summary of what is needed, already satisfied, and possible ways to complete the design (i.e., using the visual prompts to show where flexures could be placed) to promote reflection. Such *guided*, as opposed to *automated*, interaction also informs users of critical information and allows them to navigate design tasks while taking additional constraints or design considerations in mind, thereby avoiding target fixation [241] and promoting design branching and variation.

In addition to navigating the design space, step-by-step modeling and feedback are crucial to fostering user intervention and reflection. The user has direct control over the design throughout the design process, and the user can interpret and respond to the modeling instructions in any way they like and incorporate any other considerations. In a realistic CM design problem, the consequences of each user action stretch beyond just kinematics design. Each design change may also affect the device's fabricability and lead to functionally satisficing but difficult-to-implement designs. CM design through the FACT method is an increasingly complex process. As more flexures are added, the joint becomes increasingly cluttered, making adding further flexures more

difficult. As a result, users often have to revert their actions and think carefully before taking action. The plan to incorporate additional mechanical components, device ergonomics design, and fabrication strategies must also be updated by the user in real time and reflected in device modeling, creating a feedback loop. The ReCompFig tool provides the information needed to make these decisions, making the tool a reflective medium.

On the other hand, when formulating the prompts, we opted for a combination of visual and text prompts to better explain the tool's suggestions. CM flexure layout design is inherently a geometric problem, and text prompts alone could not adequately help users understand where the flexure should or should not be placed. Conversely, the FACT method's visual repertoire could also be arcane to novice users and require textual prompts to explain the meanings behind the visual prompts. When the user falsely adds a flexure to the design that would compromise their objectives, the textual prompts also help explain why the flexure is invalid and why the tool blocked the user from making further actions to provide clarity.

5.12. Future Work

Designing kinematic devices is challenging due to the involvement of expert mechanical knowledge, and it is even harder to design these devices using compliant mechanisms as it involves nonlinear physics and topological kinematics. Yet, CMs also offer advantages that are unattainable by conventional mechanisms, such as their simplicity, ease of fabrication, precision, and scalability. We have only explored and utilized some of these properties in this work, and it would be exciting to see future design tools that are more powerful and enable/engage a wider range of users to adopt CMs in their design. For instance, can we produce design tools that automatically generate CM designs that have different aesthetic and functional qualities? Can we develop intelligent fabrication software and tools that make CMs monolithically to further reduce their fabrication complexity and assembly demand, making them even more accessible to makers, researchers, and designers? Due to its complexity, CM design tools also make a valuable playground for human-computer collaboration studies to take place.

Our demonstrations exemplified how ReCompFig devices can be leveraged to enable haptics in artificial reality. Yet, further evaluations are still needed to validate their applicability. We also

speculate that it is possible to combine the proposed framework with other interaction design methods to provide even more diverse, immersive, and augmentative experiences. On the one hand, using the proposed framework and compliant mechanism design methods would allow future researchers to design shape-changing interfaces that have even more complex or dynamic tangible responses. On the other hand, incorporating shape- or stiffness-changing materials [220, 313] and other haptic modalities (e.g., texture [163], weight [211]) into kinematic devices may also provide more realistic and genuine sensations for artificial reality or produce perceptually and emotionally evocative interfaces.

5.13. Conclusion

In this work, we have introduced tension cables into compliant mechanisms to create multimodal and reconfigurable kinematic mechanisms and devices. Based on the screw theory of compliant mechanisms, we develop several design principles to govern and inform the design process. Technical contributions, including computational algorithms, design tools, and fabrication methods, are also provided. In particular, the design tool assists users in designing two aspects of a reconfigurable CM device - its prescribed DOF configurability and the cables used for dynamically switching their DOF modes. The design tool provides procedural and open-ended guidance to assist users in creating mechanisms with the desired kinematic modes and sensing capabilities. Our evaluations also show that the design framework can assist users in creating devices that have multimodal and reconfigurable kinematic behaviors. Design examples, including material displays, haptic proxies, and a multimodal input device, are also presented to showcase the mechanisms' application opportunities. Beyond enriching the kinematic device design toolbox, we also believe ReCompFig further expands the tangible interaction design space and facilitates the development of interactive haptics.

5.14. Computational Toolmaking Remark

The suggestive design tool implemented in this work helps users to iterate designs toward their goal without directly solving and presenting a solution. The designer's intention initializes the design task, and the tool helps the user take a ground-up approach and iteratively add structural elements until the design is satisfied. By involving them throughout the modeling process,

designers could reflect on the task and their plans to create a design solution. This interaction modality also allows the user to directly manipulate the design in response to factors that emerged during the process, allowing co-steering to incorporate design qualities not considered by the tool.

The design tool relies heavily on visual and textual prompts to communicate with the user. Due to its innate design challenge (i.e., nonlinearity and discontinuity), compliant mechanisms design is traditionally difficult to navigate ground-up with humans in the loop. However, the ReCompFig design tool showed that textual and visual prompts could effectively communicate the tool's suggestions and inform the user's action toward the design goal. The design tool's suggestions are also piecemeal and actionable, making it easier for users to follow through.

Still, we note that ReCompFig's inner working makes a naïve assumption about the strategy to create a reconfigurable CM design. The multimodal kinematics algorithm is deterministic while calculating a single strategy to complete a reconfigurable CM design. While it allows users to freely model flexural joints under an identified design strategy, it does not allow users to explore alternative strategies to create kinematic reconfiguration, causing certain designs to be unachievable using the current tool. This limitation is then addressed in the next chapter of a follow-up research study.

Chapter 6. Compliant Metastructure Reconfigurable at Six Degrees of Freedom

6.1. Computational Design Motivation

This work is a continuation of ReCompFig. We build upon the previous tool and transition from the notion of "helping users solve design problems" to "helping users find different ways to solve design problems", which in turn allows users to handle more complex design tasks that require not only reasoning about the modeling itself but also exploring different strategies to complete the task. While we did not implement a new CAD tool in this work, we expanded the design algorithm and toolkit to better guide users in designing a satisficing reconfigurable compliant mechanism that targets realistic, contextualized scenarios.

Using wearable haptics and kinesthetic devices as an example, we highlight several challenges designers may face when designing for contextualization, including device conformation, stiffness rendering, and more. The ReCompFig's solver could not address these challenges, but they could be accounted for by extending the algorithm and incorporating additional computational tools to create a rationalized design pipeline. These additions help users navigate different strategies in producing a satisficing design, as well as informing effective design modifications toward a numerical, functional objective, allowing users to rapidly iterate wearable device design toward realistic, contextualized design requirements.

6.2. Technical Motivation

Compliant mechanisms with reconfigurable degrees of freedom draw increasing attention in the development of mechanical transmission stages, kinesthetic haptic devices, robotic systems, and mechanical metamaterials. However, available devices have limited DOF programmability, often lack customizability, and are limited to specific form factors, therefore restricting versatility for diverse contexts. To address this gap, we propose a tailored metastructure concept with a rational

design strategy for devices with reconfigurable DOF and stiffness tunability. Such devices can also be tailored for different form factors and use cases. The devices consist of passive and actively stiffness-changing flexural rods that can alter the devices' kinematic DOF on the fly. The rational design pipeline informs the flexures' topological arrangements, geometric parameters, and control signals given the targeted motional freedom. This enables us to program independent or combinatorial DOF reconfigurability in, and up to, six DOF within a single joint, creating a fundamental unit for kinematically reconfigurable devices. As application examples, we demonstrated a 2DOF reconfigurable wrist device has an effective stiffness of 0.370 Nm/deg (unlocked state, 5% displacement) to 2.278 Nm/deg (locked state, 1% displacement), with its locked state providing the stiffness required to restrain motion under a perceptually just-noticeable displacement (4 Nm/6 deg = 0.666Nm/deg), thereby enabling dynamic control of joint mobility freedom. A haptic thimble device (2.27-52.815 Nmm^{-1} at 1% displacement) mimics the touch sensation of physical materials ranging from soft gel to metal surfaces. By assembling designed joints, we demonstrate wearable devices tailored for the arm and hand that can kinesthetically reconfigure to constrain unwanted motions, provide resistance for muscle training, or augment haptic experiences in virtual realities.

6.3. Introduction

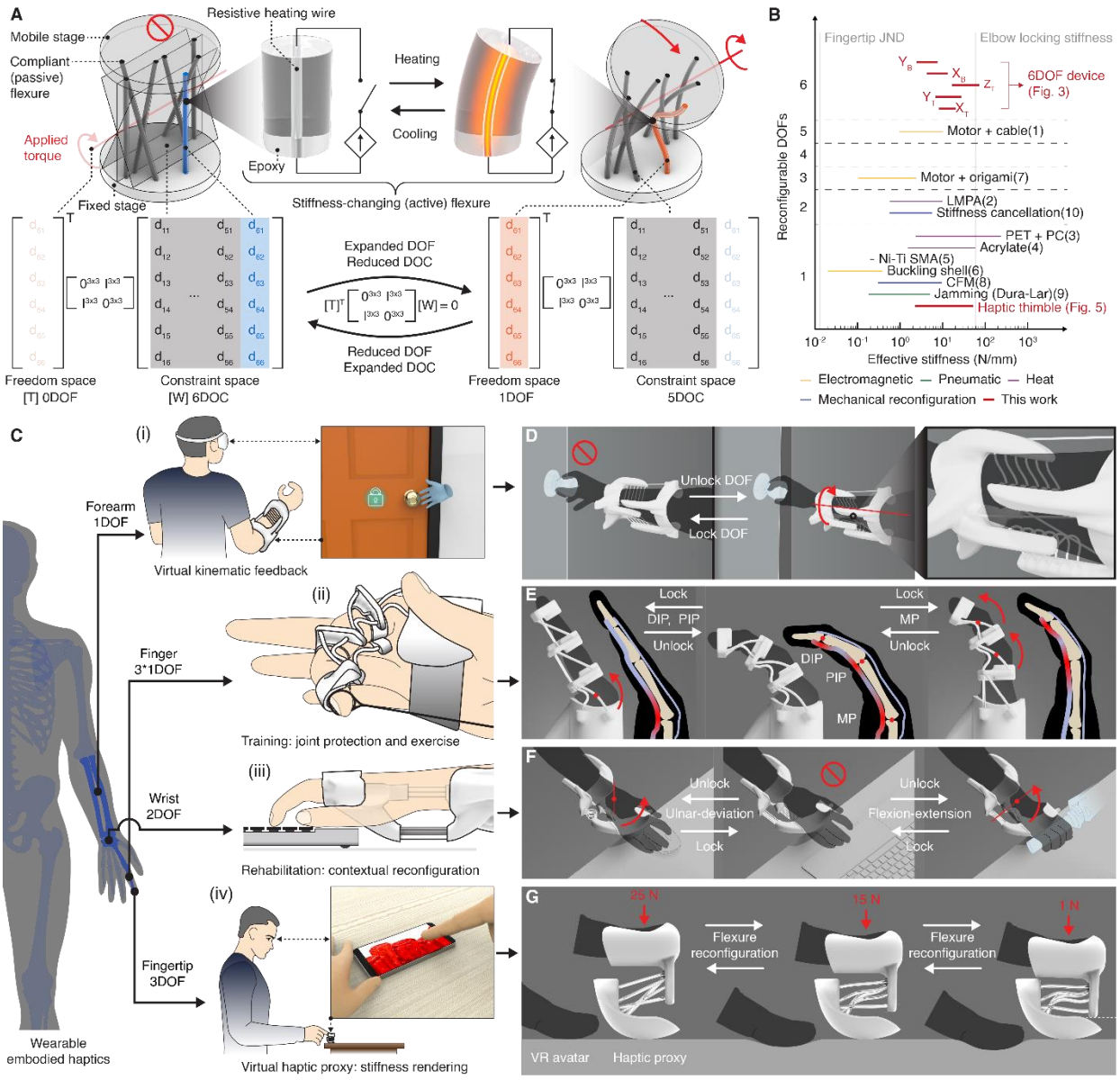

Figure 6-1. Kinematically reconfigurable compliant metastructure design and envisioned application examples. (A) Design of stiffness-changing materials for making compliant metastructures that can change their kinematics depending on the context of use and their screw algebra representation. (B) A benchmark of devices presented in this work and literature with respect to the number of programmable DOF and the range of afforded effective stiffness [40, 74, 140, 160, 183, 188, 197, 266, 335, 345]**. The range of stiffness needed for upper limb wearable kinesthetic haptic design is exemplified by the vertical dashed lines. JND, just noticeable difference; LMPA, low melting point alloy** [345]**; PET, polyethylene terephthalate** [197]**; PC, polycarbonate** [197]**; CFM, constant force mechanism** [140]**. (C) Exemplary design space enabled by reconfigurable kinesthetic haptic device leveraging DOF locking/unlocking and stiffness changes, including (i) virtual kinematic feedback, (ii) selective muscle group training, (iii) context-adaptive rehabilitation braces, and (iv) wearable haptic proxies. (D) The device can disable the forearm's rotation to, e.g., simulate the experience of turning a locked vs. unlocked doorknob (E) The kinematic reconfiguration can selectively constrain finger interphalangeal joints, allowing for targeted muscle group training. (F) The wrist device can function as a context-adaptive wrist brace (e.g., for alleviating wrist-tunnel syndrome) that can reconfigure its kinematic constraint to enable certain motions. (G) A haptic thimble device can proxy the haptic experience of pressing different materials by reconfiguring its stiffness.**

Literature has explored engineering methods for creating stiffness reconfigurable mechanisms or metamaterials [113, 230, 263, 289], as well as methods for designing stiffness reconfiguration along multiple targeted DOF [40, 74, 140, 160, 183, 188, 197, 266, 335, 345]. While such kinematic and stiffness reconfigurability could often be achieved through electromagnetic [40, 53, 77, 113, 188, 195, 335, 358], electrostatic [18, 59], or pneumatic jamming systems [74, 126, 220, 347], they are often limited to reconfigurability along few DOF due to the inherent mechanical complexity and integration [250, 271]. Alternatively, compliant structures incorporated with architected stiffness-changing materials [160, 197, 263] could afford reconfigurability without increasing the devices' mechanical complexity.

Although reconfigurable compliant mechanism designs affording binary modes have been explored [140, 266], a design method targeting multimodal (>2 modes) reconfiguration is needed but not available. To address this, we adopt a screw algebra-based model [97–99, 276] of conventional, non-reconfigurable compliant mechanisms, known as freedom and constraint topology (FACT), and extend it to account for multiple kinematic modes and reconfigurations. While such adaptation has been demonstrated in a recent study [335], it was a purely kinematic analysis and did not account for material properties, such as rod stiffness and buckling, in the design pipeline. Therefore, the previously designed devices can only reach a maximum stiffness of 0.79-10.2 Nmm^{-1}, navigating a much smaller space than the human kinesthetic perception range (0.013-59.342 Nmm^{-1}). Moreover, the prior work used passive flexures and tensioning cables for reconfiguration and had a maximum of 5-DOF programmability (i.e., a minimum of one flexure is needed, adding 1-DOC to the system). In comparison, the active flexures used in this work allowed for 6-DOF reconfigurability. On the other hand, we note that prior development in three-dimensional metamaterial and structures [52] had focused on actuation and proprioception along arbitrary DOF, and enabling kinematic and stiffness reconfiguration could further expand the design space.

In addition to the increased DOF of kinematic reconfigurability, our method also allows tailored design versatilities [22, 319] for different use contexts [224, 229, 350] in terms of their stiffness ranges and form factors. In summary, we set the following criteria for designing compliant metastructures to extend the real-world implications of such devices: i) the functions (kinematic freedoms and stiffness) should be actively reconfigurable to adapt to changing use contexts; ii) the

stiffness range should be tunable to accommodate target use cases; iii) the devices should have customizable form factors for uses in different contexts (e.g., worn on target human body areas).

To achieve these design goals, we propose a compliant metastructure design (Figure 6-1A) that is composed of both passive and active stiffness-changing flexural rods. Tailored design algorithms are presented that inform the topological arrangements, geometrical parameters, and control signals of these flexures based on target sets of reconfigurable kinematic modes. In this paper, we implemented systems that provide a large tunable stiffness changing ratio of up to 23.26x along a single DOF and a range of effective stiffness (2.445-73.785 Nmm^{-1}) tailored to the kinesthetic perception range (Figure 6-1B).

To demonstrate the large and versatile design space of our approach (Figure 6-1C), we (in collaboration with product designer Tate Johnson, a co-author of this research) implemented multiple wearable devices tailored to unique kinematic functions, body areas, and use contexts, including using DOF reconfigurability to provide kinematic feedback when interacting with virtual reality [277, 353] (Figure 6-1D), simultaneously locking/unlocking multiple joints to provide targeted muscle group training [274] (Figure 6-1E), context-adaptive rehabilitation and injury (e.g., carpal tunnel syndrome) prevention [47, 114, 216, 243, 250, 255, 262, 269] (Figure 6-1F), and proxying the haptic feelings of touching surfaces in mixed realities [212, 251, 335] (Figure 6-1G). To our best knowledge, we believe this is the first compliant metastructure design approach that enables algorithm-informed and user-prescribed compliance with up to all six DOF, which can be independently or combinatorically reconfigured.

6.4. Mechanisms of Reconfigurable Compliant Metastructure

Our fundamental design unit of the reconfigurable compliant metastructure consists of two rigid stages connected by parallel flexures (Figure 6-1A and Appendix 2). Multiple structural units can be serially connected to accommodate multiple motional freedoms at different locations (Figure 6-1C). The flexures can be passive or actively stiffness-changing; their topological arrangements, geometrical factors, and control signals will eventually determine the compliant metastructures' function and performance. Therefore, we used an algorithm-informed approach to design and control such metastructures.

We choose to engineer the actively stiffness-changing flexure with a resistive heating wire as the core and a thermoset epoxy resin-based cladding [31, 32] (Figure 6-1A and Appendix 2: Notes on Material Selection and Safety). When heated above its glass transition temperature, the resin's elastic modulus drops by 57 times from 1.14 ± 0.18 GPa to 0.02 ± 0.008 GPa. Due to their slender aspect ratio, in the cold state, both passive and active flexures have magnitudes higher stiffness against axial than bending or twisting loads, creating a degree of constraint along their axis. Yet, when the active flexures are softened, their stiffness and buckling loads are reduced proportionally to the elastic modulus, becoming soft and buckling easily against axial load. Therefore, stiffness-changing flexures can be used to create dynamic DOC, which in turn allows for kinematic reconfiguration.

Additionally, tactful flexure arrangements can instate distinct kinematic modes, each affording different mobilities and constraints (Figure 6-2A). Each kinematic mode is defined by its DOF represented as a screw vector space $[T]$ and a complementary DOC as a screw constraint space $[W]$ (Figure 6-1A). To instate a mode, the cold and stiff flexures should fully span the constraint space $[W]$, which ensures the mode is exactly constrained without allowing motions not spanned by $[T]$. The passive flexures create a permanent constraint subspace shared by all modes, whereas the stiffness-changing flexures are used to dynamically expand or truncate constraint spaces. The resulting device can then be reconfigured between kinematic modes by selective softening and stiffening of active flexures.

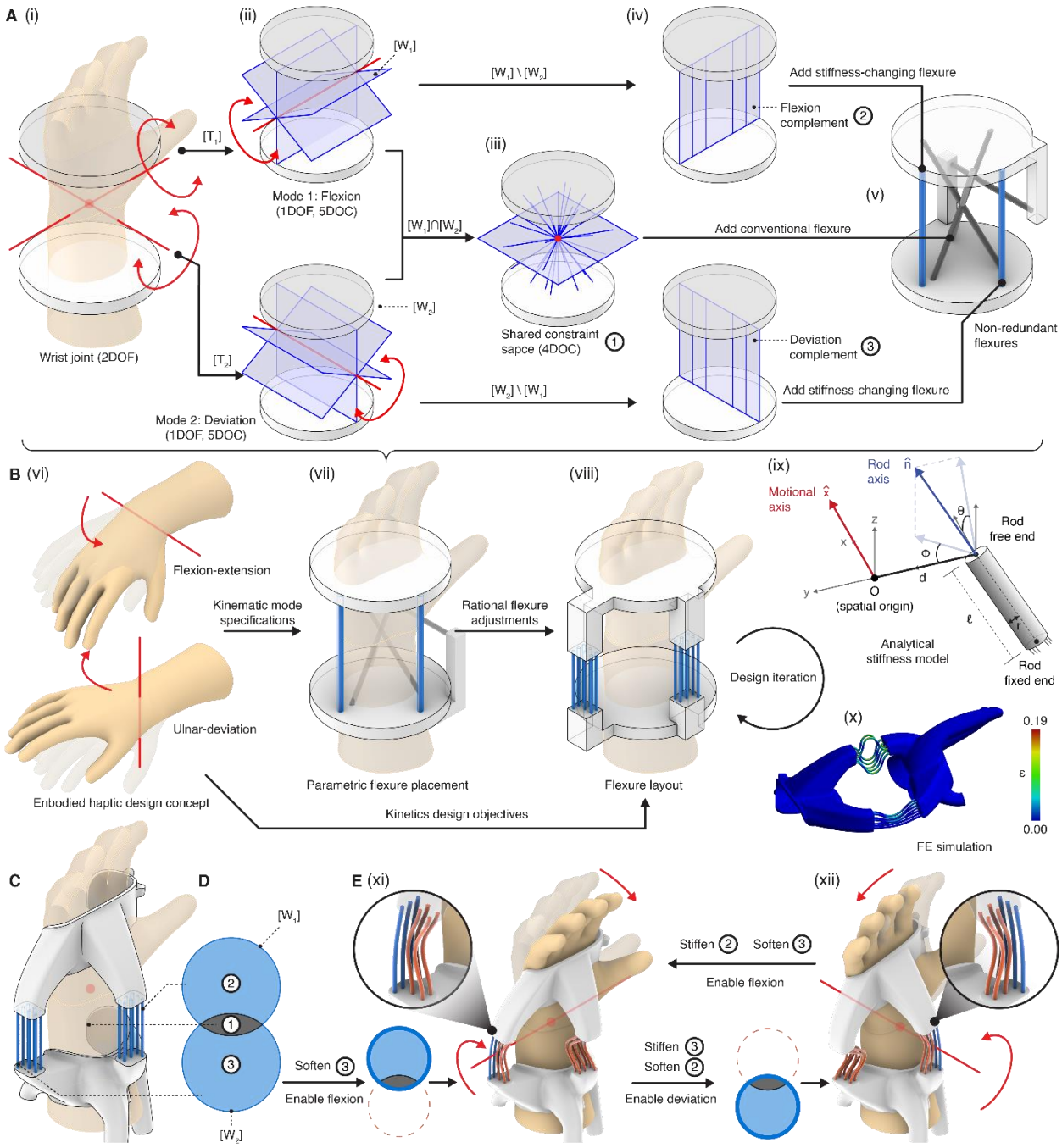

Figure 6-2. An algorithm for designing kinematically reconfigurable compliant mechanisms using the wrist joint as an example. (A) The algorithm starts by computing the (i) freedom and (ii) constraint space of deviation and flexion. (iii) The subspace for passive flexures is the intersection of all modal constraint spaces. In this design case, any rod flexure whose extended axis passes through the center point or lies on the plane spanned by the two DOF axes is permitted. (iv) The relative complements represent the rod placements required to exactly constrain the other mode. In this case, any flexure that does not pass through and is not parallel to the rotation axis is allowed. Next, minimal (v) non-redundant flexures should be added to span the constraint subspaces and exactly constrain each kinematic mode. In this case, four conventional flexures and two stiffness-changing flexures are used (one for each mode). (B) The overall design workflow for reconfigurable embodied haptic devices. (vi) The input kinematic specifications are supplied to the algorithm (a) to find (vii) the parametric flexure placement. Based on the kinetics design goals (e.g., stiffness), more flexures could then be added to the design and use (ix) an analytical stiffness model and (x) finite element simulation to validate and iterate the design (Color indicates the equivalent strain in ANSYS). In this sequence, the passive flexures from (v) are replaced by the wrist

joint's skeleton to produce (vii), then more flexures are added, and their geometric parameters are altered to produce a device with the targeted kinetic performance. (C) The final design was created by remodeling the rigid stages in (viii) to provide a good fit to the wearer and connections between flexures. (D) The design's Venn diagram representation and the membership of each space in (A). (E) Reconfiguration for the kinematic modes: the complement constraint subspace should be canceled by softening the flexures to enable (vi) flexion and (vii) deviation.

6.5. Rational Design Algorithm

We illustrate the steps to design kinematically reconfigurable devices (Figure 6-2), using the wrist joint device (Figure 6-1F) as an example. This device aims at two kinematic modes where mode 1 enables flexion-extension and mode 2 enables ulnar deviation. Under a mode, any other freedoms except the one(s) enabled should remain constrained. Mode 1 can be represented by its freedom $[T_1]$ and constraint spaces $[W_1]$, and mode 2 by $[T_2]$ and $[W_2]$. The intersection of two modes' constraint spaces $[W_1] \cap [W_2]$ are shared by both modes. Consequently, flexures placed within this intersection are needed to exactly constrain both modes and are not required to be stiffness-changing. On the other hand, the relative complements $[W_1]\backslash[W_2]$ and $[W_2]\backslash[W_1]$ are the subspaces required to exactly constrain one mode but not the other, and flexures placed in this space should be stiffness-changing. Specifically, rods placed in $[W_1]\backslash[W_2]$ are needed to exactly constraint $[W_1]$ and would resist motions in $[T_2]\backslash[T_1]$, establishing mode 1. Conversely, to instate mode 2, the flexures residing in $[W_2]\backslash[W_1]$ should be softened.

The screw subspaces parametrically describe the flexure placements that lead to the desired kinematics reconfigurability, allowing for rational and generative design and optimization toward design considerations (Figure 6-2B, C). Notably, the constraint subspaces may also have redundancies or be invalid. Therefore, only a subset (between k and 2^k-1, k denotes the number of unique kinematic modes) of complement constraint subspaces is needed to create an exactly constrained device. The stages can be modeled into any shapes that are sufficiently rigid (i.e., have minimal deflection) without altering the prescribed DOF modes [97], providing more design freedom for a customized fit or other functional purpose. In this work, we leverage finite element (FE) simulation to verify the generated designs' performance, and an analytical stiffness model was used to synthesize and adjust flexural rods' performance toward targeted values.

To identify flexural configurations, the kinematic modes and flexures of a reconfigurable device can be represented as a Venn diagram (Figure 6-2D), where each circle in the diagram represents the constraint space required to exactly constrain and instate a kinematic mode. The segments

correspond to the subspaces resulting from the algorithm and, hence, flexures. To instate a kinematic mode (Figure 6-2E), all flexures not included by the mode's constraint space should be softened while the ones included should be kept stiff. In other words, the constraint subspaces (flexures) that are not encompassed by the mode's circle should be softened to lift their constraints, while the ones located within should be kept stiff to constrain unwanted mobilities.

In summary, the rational design of reconfigurable kinematic devices can be summarized by the following five steps: (i) Compute each kinematic mode's kinematic freedom and constraint space; (ii) Compute the constraint space for placing non-stiffness-changing flexures; (iii) Compute constraint subspaces required to instate each mode; (iv) Selecting constraint subspaces for placing stiffness-changing flexures and add non-redundant flexures required to exactly constrain each mode; (v) Adding additional flexures to reach the targeted device performance and modify the rigid stages to connect to the flexures and for other functions.

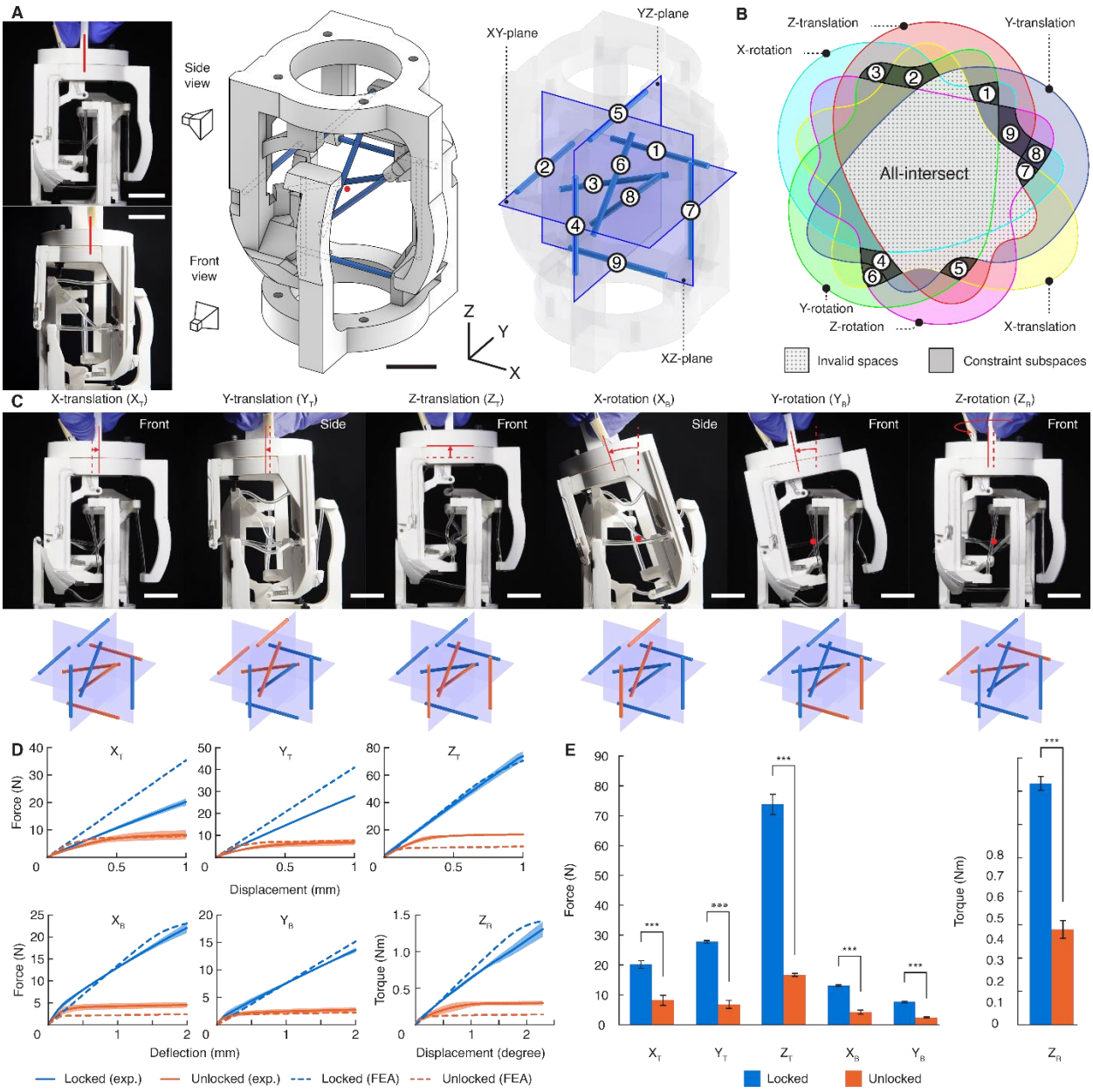

Figure 6-3. A compliant mechanism joint that can be reconfigured to provide mobility along each and any of the six DOF in the three-dimensional space. (A) The device consists of two identical stages connected by nine stiffness-changing flexural rods lying on three orthogonal planes passing through the center of rotation (red dot). Scale bar, 20 mm. (B) The Venn diagram showing the constraint subspaces calculated from the algorithm and the affinity of each rod (Methods). (C) The device's motion along each of the DOF (top row) and the corresponding flexural rod configuration (bottom row, orange: heated, blue: cold). The dashed and solid lines show the mobile stage's centerline position before and after displacement, respectively, with the arrow showing the direction of motion. Scale bar, 10 mm. (D) The load-displacement plots of each DOF in the locked (blue) and unlocked (orange) states. Data are means ± s.d. n = 3 samples. (E) 1% (1 mm) displacement loads for each DOF. Statistically significant differences were found between toggled modes using t-tests (*p<0.001). Data are means ± s.d. n = 3 samples.**

6.6. Generalized Design with 6-DOF Reconfigurability

To exemplify the designs afforded by our algorithm, we employed it to create a device that can be reconfigured to provide any of the six DOF in 3D space (Figure 6-3A). Each of the DOF was specified as a kinematic mode as input to the algorithm, leading to a total of 2^6–1=63 constraint subspaces (Figure 6-3B). Yet, several subspaces were invalid for being empty or led to unviable rod placements, leading to 56 viable subspaces to choose from for placing stiffness-changing rods. From this viable collection, we then picked nine subspaces that allowed us to place the rods on three planes through the targeted rotation center (Figure 6-3A). The resulting device consists of nine stiffness-changing rods and zero passive flexures. The device has zero degrees of freedom when all flexures are stiffened. Yet, by softening the rods according to the algorithm, the device can become mobile in each of the six DOF (Figure 6-3C).

The device was jigged and tested to reveal its distinctive load-displacement behaviors between the locked and unlocked states along each DOF (Figure 6-3D). The translational DOF was tested by linearly displacing the free end, whereas the X- and Y-rotational DOF were tested as bending deflections. Z-rotation was applied as a pure rotation by fixing the rotation center (Appendix 2: Mechanical Test Jig Design). Load curves were relatively linear for the locked states but displayed a plateau in the unlocked states. As a consequence, the loads at the end of tests were also statistically distinct between the two states (Figure 6-3E). The largest difference was observed in Z-translational (4.42x) and the smallest in X-translational DOF (2.47x). We note that the difference is bound to become larger with increasing displacements as the unlocked state has a stiffness close to zero due to the buckling of flexures.

6.7. Wearable Kinesthetic Haptic Devices for Mobility Reconfigurations

The proposed design approach allowed us to tailor wearable devices for human augmentation. Here it is demonstrated through the design of a device that can be worn on the arm and hand (Figure 6-4A). The kinematics and performance are both considered per joint, and the joints were designed individually and combined later to form the complete device. In the rest pose, the arm and finger are fully extended. The joints are designed through the rational design algorithm and

can be reconfigured to lock or unlock each of the afforded DOF. Note that when designing a wearable device, the skeletal structure can be considered a part of the passive flexures, which readily and exactly constrains the DOF. Therefore, it is optional to add passive flexures, though they may provide benefits such as maintaining the relative position between two stages.

In addition to kinematic modes, the device's stiffness with respect to the human body's performance and perception should also be considered for embodied haptics. We set our design criteria based on the torques exerted by each human body joint [89, 186] and the just noticeable difference (JND) of joint angle proprioception [239] (Table 6-1). To lock a DOF at a body joint, the device should displace less than the JND when subjected to the exertable torque, such that the wearer cannot perceive any movement. Conversely, in the unlocked state, the joint should be able to displace to and above the JND with a load lower than the exertable torque.

Table 6-1. The arm-wearable device design criteria.

Joint	Isometric Strength (Nm)[180, 186, 306]	Design criteria (Nm)[89, 186]	Just noticeable difference (degree)[239]
Forearm	10	5	8
Wrist flexion	10.92	4	6
Wrist deviation	8.46	4	7
Metacarpophalangeal (MP)	0.8	0.8	8
Proximal interphalangeal (PIP)	0.37	0.37	7
Distal interphalangeal (DIP)	0.14	0.14	9

The forearm joint's pronation/supination is defined as an axial rotation along the length of the arm (Figure 6-4B). The two stages are placed at the ends of the forearm and connected by twelve stiffness-changing flexures for reconfiguration and three passive flexures to maintain the spacing between stages. Based on the flexure placements suggested by the algorithm, we iteratively designed the device against the criteria (torque and JND) by changing the placement and number of rods in the system. Our FE simulation reveals that when locked, the device has a displacement of 0.83 degrees under the torque limit of 5 N-m, much lower than the JND of 8 degrees, suggesting

that the device is perceptually immobile. Yet, when the joint is unlocked, the device requires only 0.79 N-m to reach the JND, and the device becomes compliant against the wearer's motions.

The three finger joints each have a rotational DOF about the interphalangeal joint and an identical design (Figure 6-4C). The rigid stages are added at the phalanges and are connected by a stiffness-changing and a passive flexure, and the device is designed by changing the distance between the stiffness-changing flexures and the rotation axes. Similar to the forearm joint, our FE analysis also verifies that the device is perceptually immobile in the locked state but becomes mobile upon heating the stiffness-changing flexures.

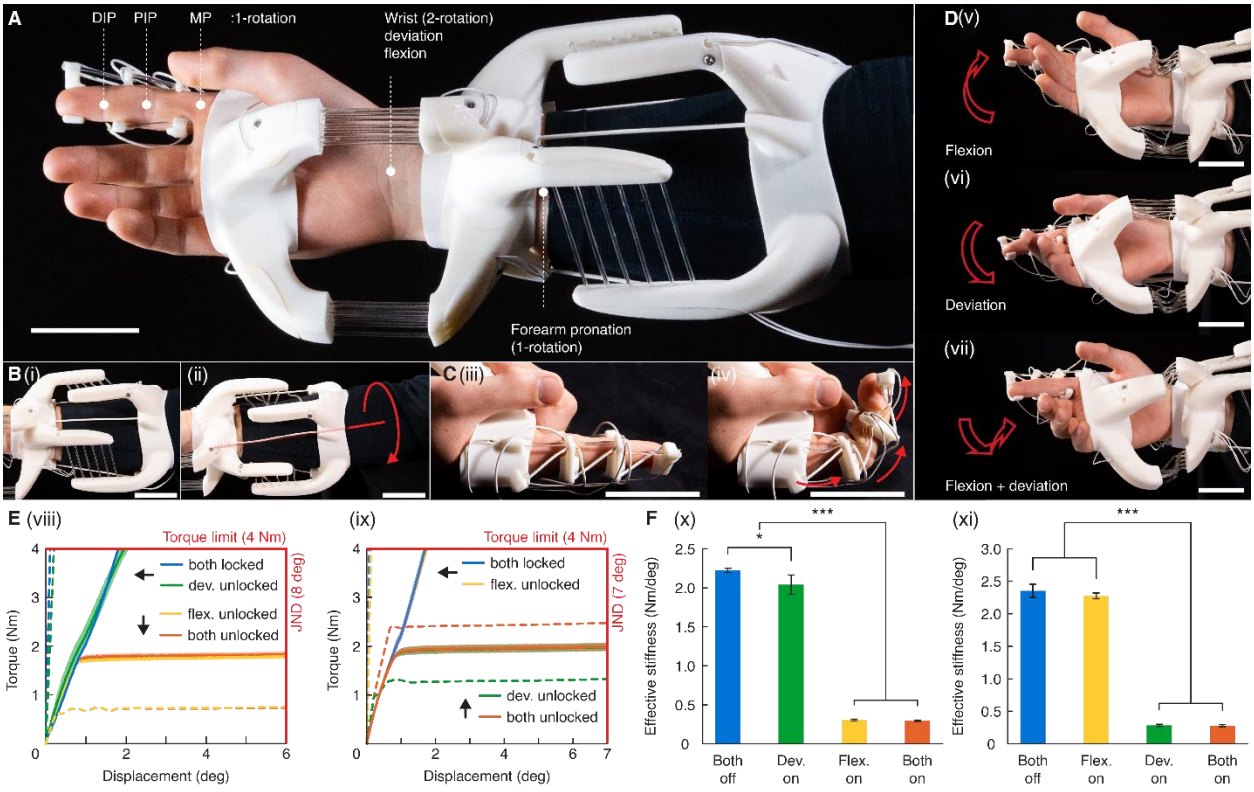

Figure 6-4. Tailored design for wearable kinesthetic haptics. (A) Picture of a device for the arm to toggle individual joint DOF. Scale bar, 50 mm. (B) Pictures of the unlocked forearm joint (i) before and (ii) after pronation. Scale bar, 50 mm. (C) Pictures of the unlocked finger joints (iii) before and (iv) after flexion. Scale bar, 50 mm. (D) Pictures of the wrist joint exercising along the unlocked (v) flexion, (vi) deviation, and (vii) both directions. Scale bar, 50 mm. (E) The load-displacement plots of the wrist joint device's (viii) flexion and (ix) deviation DOF under different configuration modes. Data are means ± s.d. n = 3 repetitions. Dashed lines are FE simulation results. (F) Stiffness comparison of the wrist joint device under different configuration modes against the (x) flexion and (xi) deviation DOF. The effective stiffness was calculated as load/displacement at the experiment criteria (i.e., torque limit load for locked states and JND limits for unlocked states). Data are means ± s.d. n = 3 repetitions.

Different from the finger and forearm joints affording single rotational freedom, the wrist joint allows rotation about two axes - flexion-extension and ulnar-radial deviation (Figure 6-4D). Each

rotation can be individually locked or unlocked, leading to four possible kinematic modes. The joint design consists of fourteen stiffness-changing flexures with three rods placed in-plane with each of the rotation axes. To enable a rotational freedom, all stiffness-changing flexures should be softened, except for those coplanar with the motional axis. Softening all flexures at the same time will allow both flexion and deviation to become unlocked. While the kinematics design of the wrist joints had been detailed in Figure 6-2, the exact numbers of flexures were further determined through an iterative design process comparing the stiffness values from FE simulations with our design criteria including the human wrist joint torque limit and JND.

We evaluated the wrist joint design through mechanical tests. We isolated the device's wrist joint, mounted it on an articulated testing jig, and loaded it along each of the DOF to measure its responses (Figure 6-4E and Appendix 2: Mechanical Test Protocol). When both DOF were locked, the joint displaced by 1.81 ± 0.02 and 1.71 ± 0.07 degrees at the torque limit when loaded with flexion and radial deviation, much lower than the JND of 8 and 7 degrees, respectively. Effective DOF locking was observed when the other DOF was unlocked: the displacement along flexion increased to 1.76 ± 0.03 degrees when radial deviation was enabled, indicating that the flexion DOF remained perceptually immobile. The same increase was also observed for the radial deviation DOF when flexion was unlocked (1.97 ± 0.12 degrees), but the DOF remained locked. On the other hand, when both DOF are unlocked, the device requires 1.82 ± 0.06 and 1.98 ± 0.09 N-m of torque to displace to the JND threshold along flexion and radial deviation, respectively, indicating the device can move past the JND with a torque lower than the limit and is perceptually mobile. The effective stiffnesses calculated at the JND and torque limit intercepts (Figure 6-4F) also showed no statistically significant differences ($p > 0.5$) between identical modes except for when flexion is locked ($p = 0.33$). Additionally, a strong statistical difference ($p < 0.001$) was observed between the locked and unlocked states, showing the device had distinctive perceived stiffness between locked and unlocked modes within human kinesthetic limits.

6.8. Wearable Haptic Thimble Device for Stiffness Tunability

In addition to mobility reconfiguration, the compliant metastructure design also enabled us to create a haptic device that renders a wide spectrum of stiffness. A device is designed to be worn on the fingertip to proxy the haptic feedback of pressing a surface to explore its material elasticity

(Figure 6-5A). In such an exploratory task, the finger applies forces across a 10 mm square area with a maximum force of 25 N, and the stiffness JND is 20 N/mm [360]. The device should be perceptually immobile when fully stiffened and could displace by up to 5 mm in the softest mode with a fraction of the maximum force.

The device is designed to have zero DOF in the stiffest and six DOF in the softest mode. Yet, while the kinematic modes are binary, we added stiffness-changing flexures with redundancy to create different levels of resistance against compression (Figure 6-5B). Four pairs of stiffness-changing flexures (1.5 mm diameter) are added in mirror symmetry. The flexure pairs have slightly different orientations and positions. When selectively softened, the flexures create different levels of stiffness, buckling plateau, and hence haptic response. A 2 mm diameter stiffness-changing flexure is added to provide higher stiffness in the fully locked mode. Each group of rods can be heated or cooled separately, leading to 2^5 reconfiguration modes.

We tested the device under a subset of configuration modes to find its afforded range of stiffnesses (Figure 6-5C-D). Between the stiffest and most compliant modes (Figure 6-5D), the effective stiffness calculated at the JND and force limit varied by 172.26 times from 0.27 ± 0.02 N/mm to 46.51 ± 2.42 N/mm. This range of stiffness corresponds to an effective modulus of 54.4 kPa to 10 MPa considering the device's design parameters. Perceptually speaking, this range of elasticity is identical to the sensation of touching jelly and rubber. Yet, in the stiffest state, since the thimble's compliance is lower than the JND, the device is virtually undeformable to human perception and, therefore, can be used to proxy the haptics of pressing stiffer materials, such as aluminum. Further repeatability tests also revealed the thimble device performed relatively consistently over 100 loading cycles (Appendix 2: Device Repeatability).

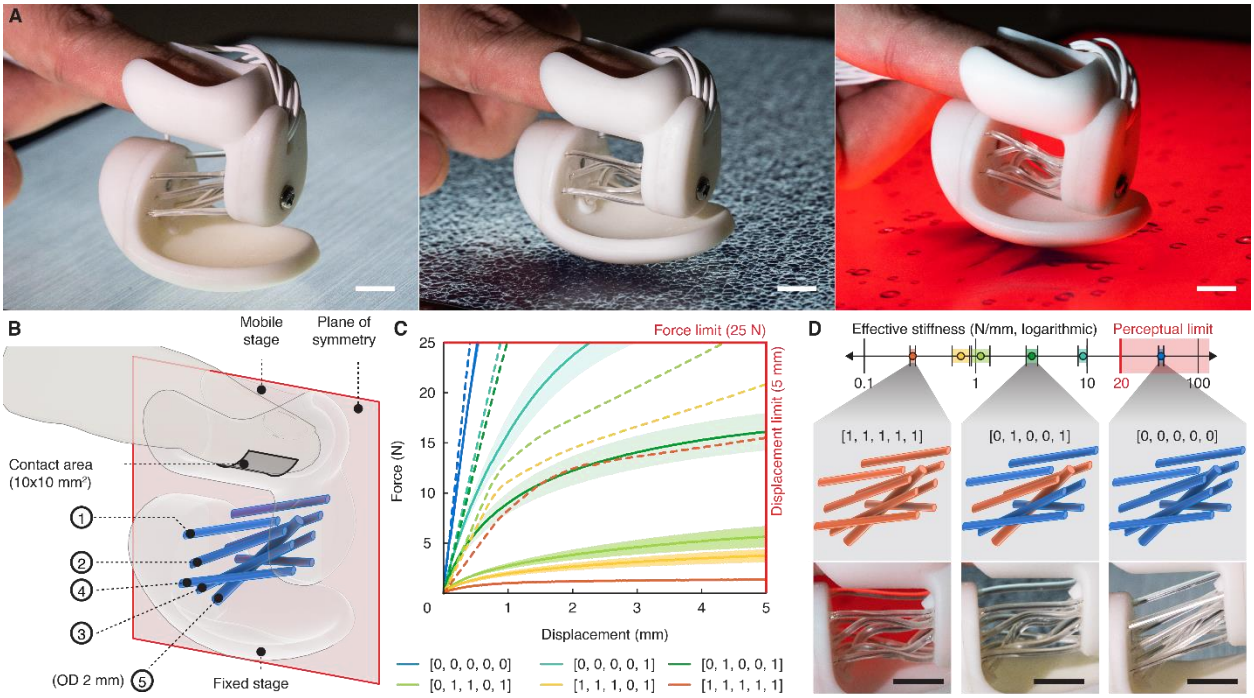

Figure 6-5. A tailored design of embodied haptics proxy. (A) Picture of the haptic thimble simulating the stiffness of pressing on a block of (i) aluminum stock, (ii) polyurethane foam, and (iii) a block of Jello. Scale bar, 10 mm. (B) The device design schema. All rods are OD 1.5 mm unless specified otherwise. (C) The load-displacement plots of the thimble under different configurations. The five numbers making up the name of each sample indicate the states of flexure groups 1-5 in (B) with 1 indicating softening and 0 for softening. Data are means ± s.d. n = 3 samples. Dashed lines are FE simulation results. (D) The device simulates different levels of stiffness within the design parameters, spanning two orders of magnitudes. Data are means ± s.d. n = 3 samples. The lower two rows show flexure configurations (middle row) and pictures of deformed flexures (bottom row) under three modes: [0, 0, 0, 0, 0] being fully rigid, [0, 1, 0, 0, 1] being partially softened, and [1, 1, 1, 1, 1] being fully softened. Scale bar, 10 mm.

6.9. Algorithm for Designing Reconfigurable Devices

The goal of this algorithm is to design a kinematic device that affords distinct kinematic modes (i.e., DOF) by selectively softening/stiffening flexural rods within the device (Figure 6-6). The algorithm takes k numbers of kinematic modes $\boldsymbol{T} = \{[T_1'], \cdots, [T_k']\}$ as input, each described by a set of twist vectors defining their motional freedoms. Next, given the targeted modes, the algorithm finds the allowed placements of non-reconfigurable and stiffness-changing flexural rods. Step 2 finds the shared constraint spaces between kinematic modes, where rods are not required to be actively stiffness-changing, whereas steps 3 through 5 identify and create the minimal actively stiffness-changing constraint topology to achieve kinematic modal reconfigurations. Step 6 further allows users to add redundant flexures to achieve targeted kinesthetic performances.

We note that steps 2 through 5 are extensions made from the FACT method [97, 98] to handle modal reconfigurations, which have also been discussed in a previous work [335]. However, previous work only presented the high-level design concept without rigorous and detailed formulation for implementation. More importantly, steps 4 through 6 incorporated newly introduced rules for augmenting constraint spaces to create more placement options, which are critical to wearable device design where available flexure placements are often confined by the user's body. Therefore, introducing constraint space augmentation rules navigates a larger design space and provides more freedom for flexural rod placements while achieving the targeted kinematic reconfigurations. Finally, the previous work used tensioning cables for kinematic reconfiguration, which could be subjected to tensile and compressive loads in the relaxed (unlocked state) without failure. Yet, the actively stiffness-changing flexures used in this work could only be compressed, not extended, and a later section further introduces an orientation check to make sure the active flexures experience a legal (compressive) load in their unlocked state.

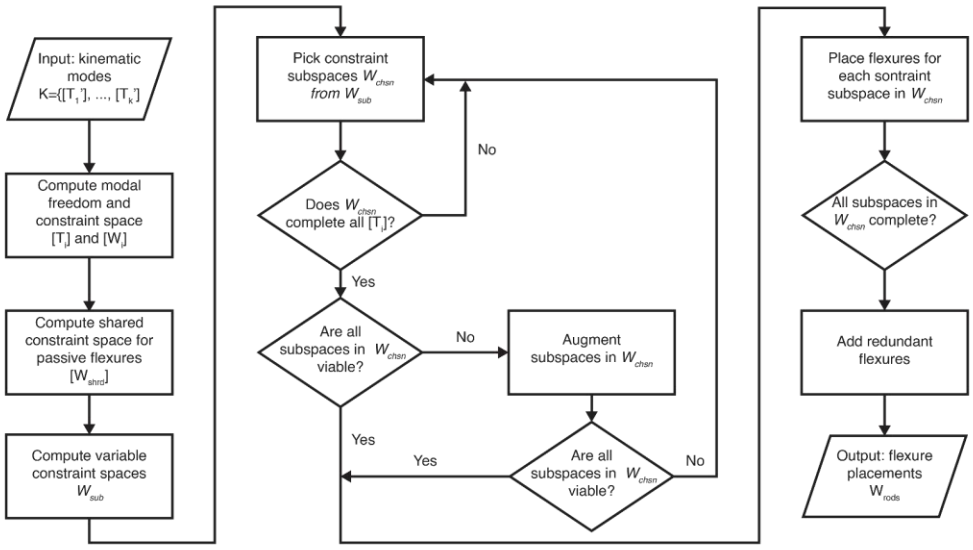

Figure 6-6. Algorithm for designing kinematically reconfigurable compliant metastructure joints.

The design process is divided into the following steps:

Step 1. Compute kinematic modes' freedom and constraint spaces.

Calculate each kinematic mode's corresponding freedom and constraint space using eq. 9-5 and eq. 9-7. Given a kinematic mode $[T_i'] \in T$, the freedom space $[T_i]$ can be found by computing:

$$[T_i] = \mathcal{N}(\mathcal{N}([T_i']))$$

eq. 6-1

and the corresponding constraint space can be calculated using

$$[W_i] = \mathcal{N}\left([T_i]^T \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}\right)$$

eq. 6-2

The derived spaces $[T_i]$ and $[W_i]$ will be used in later steps.

Step 2. Compute conventional flexure placements.

The all-intersection represents a constraint subspace $[W_{shr}]$ shared by all kinematic modes. Rods lying on this subspace are needed to enable all kinematic modes and, thus, are not required to be reconfigurable. The all-intersection of a device containing k kinematic modes are computed by

$$[W_{shr}] = \cap_{i=1}^{k} [W_i]$$

eq. 6-3

Noticeably, certain combinations of kinematic modes may lead to a shared constraint space that is unviable (e.g., the 6-DOF device). In this case, no conventional flexures should be used, or the kinematic modes will be over-constrained, and $[W_{shr}] = [0]$ is superimposed. In addition to the flexures added during the design process, any existing flexures and articulated joints (e.g., the wearer's wrist in Figure 6-2A) also counts toward this constraint space. Thus, it is possible to complete the shared constraint space without adding additional flexures, which is common in the design of wearable devices where the human skeleton readily and exactly constrains the kinematics.

Step 3. Finding variable constraint subspaces

To enable and exactly constrain kinematic mode i, stiffness-changing flexural rods should be placed in the subspace difference $[W_{dif\_i}]$ between $[W_{shr}]$ and $[W_i]$, such that the stiffened rods, together with the conventional flexures, complete $[W_i]$. I.e.,

$$[W_i] = [W_{shr}] \cup [W_{dif\_i}] \therefore [W_{dif\_i}] = [W_i] \backslash [W_{shr}]$$

eq. 6-4

Additionally, several different subspaces may also share a constraint space larger than $[W_{shr}]$; stiffness-changing flexures placed in such spaces can be shared among several kinematic modes, and the reconfigurable device can be completed without adding stiffness-changing flexures to each and every difference subspace. The constraint subspace $[W_{\boldsymbol{A}}]$ shared by kinematic modes $\boldsymbol{A} \subset \boldsymbol{T}$ and no others (i.e., exclusively shared by modes in $\boldsymbol{A}$) can be found by the following equation:

$$[W_{\boldsymbol{A}}] = \cap_{i\in\boldsymbol{A}}^{k}[W_i] \setminus \cup_{j\notin\boldsymbol{A}}^{k}[W_j]$$

eq. 6-5

For a device with k kinematic modes, there are potentially $2^k - 2$ constraint subspaces shared by different subsets of kinematic modes (minus the all-intersect and the constraint subspace that overconstrains all modes). However, some subspaces may produce an empty intersection and thus can be omitted. We call the set of all intersected subspaces $\boldsymbol{W_{sub}}$ in the following steps.

Step 4. Selecting variable constraint subspaces.

A minimum of k shared constraint subspaces should be chosen for placing stiffness-changing flexural rods. The chosen subspaces $\boldsymbol{W_{chsn}}$ should meet two conditions: for a kinematic mode i,

those included by $[W_i]$ should complete $[W_i]$:

$$[W_i] \leftrightarrow \cup_j[W_j]\ \forall[W_j] \in \boldsymbol{W_{chsn}} \rightarrow [W_j] \subseteq [W_i]$$

eq. 6-6

Those not included by $[W_i]$ should complete $\mathcal{N}([W_i])^T$:

$$\mathcal{N}([W_i])^T \leftrightarrow \cup_j[W_j]\ \forall[W_j] \in \boldsymbol{W_{chsn}} \rightarrow [W_j] \nsubseteq [W_i]$$

eq. 6-7

The former condition ensures a kinematic mode is exactly constrained when it's enabled, and the latter condition makes sure its kinematic freedom $[W_i]$ can be completely disabled in the other kinematic modes. Additionally, $\boldsymbol{W_{chsn}}$ should always include $[W_{shr}]$ if it is viable.

It is worth mentioning that any combination of $\boldsymbol{W_{chsn}}$ is valid as long as eq. 6-6 and eq. 6-7 are met, which provides flexibility when designing the device (e.g., avoiding rod cluttering and collision, achieving targeted stiffness, and aesthetics). More information is provided in Section 2.6.

Step 5. Placing nonredundant flexures

Flexural rods should be added according to and complete each subspace in $\boldsymbol{W_{chsn}}$. Note that some selected constraint subspaces may be unviable. In that case, such constraint subspace $[W_j]$ shared by kinematic modes $\boldsymbol{A_j} \subset \boldsymbol{T}$ can be augmented (unioned) with another constraint subspace $[W_{aug}]$ in $\boldsymbol{W_{chsn}}$ if the kinematic modes $\boldsymbol{A_{aug}}$ that intersected into $[W_{aug}]$ is a strict superset of $\boldsymbol{A_j}$ (i.e., $\boldsymbol{A_{aug}} \subset \boldsymbol{A_j}$).

Step 6. Adding redundant flexures

In addition to the nonredundant flexures required to exactly constrain and enable each targeted kinematic mode, additional flexures may also be added to the device to achieve targeted performances.

Figure 6-7 provides an exemplary design process following the algorithm, using the wrist joint design as an example. Each input kinematic mode is represented as screw linear spaces in steps 1 to 4. Then, the design was modeled in steps 5 and 6 by placing flexural rods and replacing passive rods with the human skeletal structure.

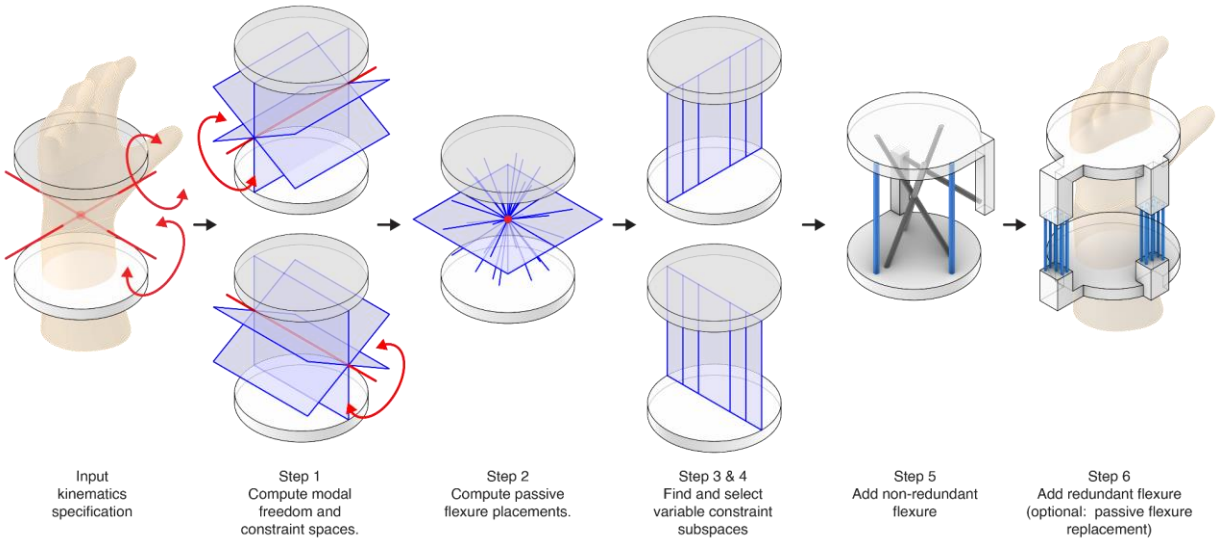

Figure 6-7. A step-by-step design example of the kinematics design algorithm using the wrist joint as an example.

Determining Flexure State for Kinematic Modes

A stiffness-changing flexural rod's state under a kinematic mode i can be determined by comparing its wrench vector $\widehat{W}_{rod}$ against the kinematic mode's constraint space $[W_i]$. $\widehat{W}_{rod}$ should be softened if it is not spanned by $[W_i]$. This check can be performed using the following formula:

$$\mathcal{N}([W_i])\widehat{W}_{rod} = \hat{0}$$

eq. 6-8

If the formula evaluates to false, then the wrench vector is not included in $[W_i]$ and is an Over-constraining element, it should be softened to enable the targeted kinematic mode. Conversely, if the formula is evaluated to be true, the rod can be left stiffened without compromising the targeted kinematics. Such flexures can also be softened to provide a lower stiffness along the enabled DOF, but it should be noted that the constraint space formed by stiff flexure $[W_{stf}]$ should complete $[W_i]$ to exactly constrain the mode. Thus, the following conditions are required:

$$\mathcal{N}([W_i])[W_{stf}] = [0]$$

eq. 6-9

$$\mathcal{N}([W_{stf}])[W_i] = [0]$$

eq. 6-10

Representing Constraint Subspaces Using Venn Diagrams

The constraint subspaces in steps 2 and 3 of the algorithm can be represented as a Venn diagram to show their relationship with kinematic modes. Such a diagram is helpful during the rational design process to help users identify and strategize flexure placements and modal reconfigurations. The constituent shapes (e.g., circle) represent the kinematic modes' constraint spaces, and subspaces intersected by shapes are shared among them. In particular, the center of a Venn diagram is the all-intersection and represents $[W_{shr}]$, whereas other segmented regions represent constraint spaces shared by the overlapping kinematic modes (i.e., $\boldsymbol{W_{sub}}$).

The spaces encircled by a kinematic mode i are spanned by their constraint space and can be stiffened under mode i. Conversely, the subspaces not included by a kinematic mode will overconstrain it and should be softened to enable the mode. In step 4 of the algorithm, the two conditions for choosing $\boldsymbol{W_{chsn}}$ can also be interpreted on the Venn diagram: the union of chosen subspaces encircled by kinematic mode i should complete $[W_i]$, and the ones outside should complete $\mathcal{N}([W_i])^T$.

Principles of Selecting Variable Constraint Subspaces

The conditions described in step 4 of the algorithm are the minimum requirements for designing a multimodal kinematic device. We note that some design properties or objectives (e.g., reducing the number of rods) can be obtained through tactful choosing of $\boldsymbol{W_{chsn}}$ and to avoid certain mechanical design issues.

Designing a device with fewer rods can be achieved by prioritizing $\boldsymbol{W_{chsn}}$ on most-shared subspaces, such that rods placed in those can be shared by as many kinematic modes as possible and require fewer stiffness-changing flexures to complete the design. For a device with k kinematic modes, prioritizing picking subspaces shared by a descending number of modes (i.e., $k-1$, $k-2$, …, 1.) will be an efficient way of choosing constraint subspaces. Each descending level is shared by fewer kinematic modes and should only be chosen when the preceding ones do not complete the modes' constraint subspaces.

Constraint space intersections may also yield unviable subspaces (e.g., without a directional component) or cause mechanical design issues (e.g., rods cluttering and overlapping). In this case, the subspaces can be augmented to provide more design freedom. An unviable constraint subspace $[W_{\boldsymbol{A}}]$ intersected by a subset of modes $\boldsymbol{A}$ can be expanded into $[W'_{\boldsymbol{A}}]$ by lifting the difference operator in eq. 6-5:

$$[W'_{\boldsymbol{A}}] = \bigcap_{j\in\boldsymbol{A}}^{k}[W_j]$$

eq. 6-11

such that $[W'_{\boldsymbol{A}}]$ navigates a larger linear subspace. This augmentation, in turn, provides more freedom for placing rods. Additionally, we may also re-write $[W'_{\boldsymbol{A}}]$ as the intersection of $[W'_{\boldsymbol{A\_1}}]$ and $[W'_{\boldsymbol{A\_2}}]$ with their corresponding intersection mode subsets, $\boldsymbol{A_1}$ and $\boldsymbol{A_2}$, respectively, meeting the following condition:

$$[W'_{\boldsymbol{A}}] = [W'_{\boldsymbol{A\_1}}] \cap [W'_{\boldsymbol{A\_2}}], where\ \boldsymbol{A}_1 \cup \boldsymbol{A}_2 = \boldsymbol{A}, \boldsymbol{A}_1 \subset \boldsymbol{A}\ \&\ \boldsymbol{A}_2 \subset \boldsymbol{A}$$

eq. 6-12

This can be interpreted as dividing $\boldsymbol{A}$ into two subsets with lower cardinality (i.e., the number of modes sharing a constraint subspace), each containing fewer kinematic modes. After applying eq. 6-12, the products $[W'_{\boldsymbol{A\_1}}]$ and $[W'_{\boldsymbol{A\_2}}]$ navigate a larger constraint subspace than $[W'_{\boldsymbol{A}}]$ since it is the intersection of fewer kinematic modes. This way, both $[W'_{\boldsymbol{A\_1}}]$ and $[W'_{\boldsymbol{A\_2}}]$ should be softened to enable a kinematic mode $i \notin \boldsymbol{A}$, and if mode $i \in \boldsymbol{A}$, $[W'_{\boldsymbol{A\_j}}]$ should be softened if $i \notin \boldsymbol{A_j}$. More, $[W'_{\boldsymbol{A}}]$ can be re-written into the intersection of any number of constraint subspaces, and the condition described in eq. 6-11 is expanded into

$$[W'_{\boldsymbol{A}}] = \bigcap_i [W'_{\boldsymbol{A\_i}}], where\ \bigcup_i \boldsymbol{A_i} = \boldsymbol{A}\ \&\ \boldsymbol{A_i} \subset \boldsymbol{A} \forall \boldsymbol{i}$$

eq. 6-13

Finally, constraint subspace augmentation can also be addressed on a vector level. If a constraint subspace $[W_{\boldsymbol{A}}]$ is missing or lacking directional components, it is advised to augment it using eq. 6-11 and eq. 6-12 to produce constraint subspaces that afford valid rod placements. Alternatively, if $[W_{\boldsymbol{A}}]$ leads to cluttered or mechanically unviable rod placements, it should be augmented with a positional component.

Principles of Placing Flexural Rods

When placing flexural rods, in addition to common mechanical design considerations (e.g., avoiding collision, delivering targeted stiffness), the rod's direction should also be taken into notice. Flexural rods undergo different types of loads depending on whether they are in (i.e., as conventional flexures) or outside of (i.e., as kinematically locking elements or stiffness-changing flexures) a constraint space with respect to a motion. Rods placed in a kinematic mode's constraint space will dominantly bend to enable that motion. By contrast, rods placed outside the constraint space are chiefly subjected to axial loads. Therefore, the orientation of these rods should be considered when they exhibit drastically different extensions and compression deformability, such as the epoxy rods used in this work. In particular, the stiffness-changing rods can only compress, not extend. Thus, an additional check is required to ensure all rods are subjected to compressive loads under a prescribed motion. This check can be performed by using

$$D = \left(\hat{n}_{axis} \times (\hat{r}_{rod} - \hat{c}_{axi})\right) \cdot \hat{n}_{rod}$$

eq. 6-14

for rotational and

$$D = \hat{n}_{axis} \cdot \hat{n}_{rod}$$

eq. 6-15

for translational motions, where D is an indicator of a rod's direction with respect to the motion, $\hat{n}_{axis}$ and $\hat{n}_{rod}$ are vectors along and $\hat{c}_{axis}$ and $\hat{r}_{rod}$ are reference points on the motional and rod axes, respectively. In particular, $\hat{n}_{rod}$ should always point from the fixed base stage to the free-moving stage. The rod is subjected to compression if $D < 0$ and extension if $D > 0$. $D = 0$ indicates that the rod lies in the motion's constraint space. Note that this check is only required by flexures that are not extensible. If the flexures are instead allowed to deform in both directions, the check can be omitted.

6.10. Analytical Stiffness Model

The device's stiffness is also a pivotal part of the design process especially when designing for locking motions under an expected load. For this reason, we provide a summary of design

parameters that affect a compliant mechanism's stiffness with respect to a motion. More detailed analysis can be found in the literature [276].

A flexure's deformation twist $\hat{T}$ and reaction wrench $\hat{W}$ are related by

$$\hat{W} = [K]\hat{T}$$

eq. 6-16

For a compliant mechanism joint comprising parallel flexures, the joint's stiffness matrix $[K_{joint}]$ can be modeled as the sum of its parallel flexures:

$$[K_{joint}] = \sum_i [K_i]$$

eq. 6-17

Where $[K_i]$ is the stiffness matrix of flexure i within the parallel flexure joint. Note that all $[K_i]$ must share the same reference frame in space.

Assuming a cylindrical flexural rod of radius r and length l (Figure 6-8). Its relative position with respect to a motional axis $\hat{a}$ can be described with three parameters d, θ, and φ, where d is the minimal distance between the rod's axis $\hat{n}$ and the motional axis (Figure 6-2B-ix). The angular parameters θ and φ describe the rod vector's direction with respect to the motion axis. The material constituting the rod is assumed to have an elastic modulus E and shear modulus G. The flexure's second moment of inertia $I = \pi r^4/4$ and torsion constant $I = \pi r^4/2$ are then functions of its geometric parameters.

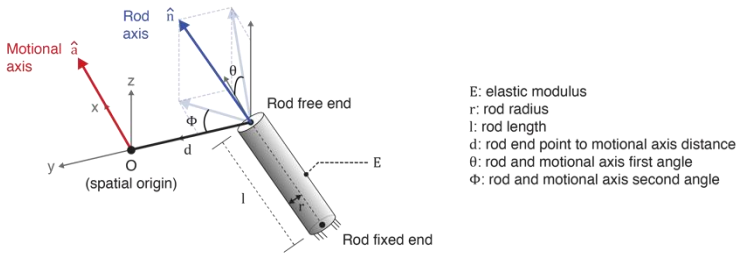

Figure 6-8. The flexural rod and motional axis parameterization of the analytical stiffness model.

Based on the Euler beam theory, the flexure's stiffness against a motion is proportional to r and inversely proportional to l. On the other hand, the relative position also affects a flexure's stiffness

contribution to a motion of interest. Following the model established by Su et al. [276], an x-axis-aligned flexural rod's stiffness matrix $[K_c]$ at the center middle of the flexure can be expressed as:

$$[K_c] = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} \begin{bmatrix} K_T & 0 & 0 & 0 & 0 & 0 \\ 0 & K_B & 0 & 0 & 0 & 0 \\ 0 & 0 & K_B & 0 & 0 & 0 \\ 0 & 0 & 0 & K_A & 0 & 0 \\ 0 & 0 & 0 & 0 & K_L & 0 \\ 0 & 0 & 0 & 0 & 0 & K_L \end{bmatrix}$$

eq. 6-18

Where $K_A = EA/l$ and $K_L = 12EI/l^3$ are the rod's stiffness against axial and lateral translations, respectively, and $K_T = GJ/l$ and $K_B = EI/l$ are the rod's torsional and bending stiffness against rotations about and perpendicular to its axis. The swap on the left operator was added because Su et al. [276] used a different definition for wrench vectors (i.e., force vector preceding moment vector). Given $[K_c]$, the stiffness matrix $[K]$ at the free end of the flexure can be found by applying adjoint transformation:

$$[K] = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} [Ad_c][K_c][Ad_c]^{-1}$$

eq. 6-19

Where $[Ad_c]$ is the adjoint transformation matrix [276] from the rod's middle to the free end frame, which can be expressed as:

$$[Ad_c] = \begin{bmatrix} I & 0 \\ D_C & I \end{bmatrix}, D_C = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{l}{2} \\ 0 & -\frac{l}{2} & 0 \end{bmatrix}$$

eq. 6-20

After substitution, we then get:

$$[K] = \begin{bmatrix} 0 & 0 & 0 & K_A & 0 & 0 \\ 0 & 0 & -\frac{K_L l}{2} & 0 & K_L & 0 \\ 0 & \frac{K_L l}{2} & 0 & 0 & 0 & K_L \\ K_T & 0 & 0 & 0 & 0 & 0 \\ 0 & K_B + \frac{K_L l^2}{4} & 0 & 0 & 0 & \frac{K_L l}{2} \\ 0 & 0 & K_B + \frac{K_L l^2}{4} & 0 & -\frac{K_L l}{2} & 0 \end{bmatrix}$$

eq. 6-21

Moreover, assuming the motion is x-axis-aligned and passes through the spatial origin. The flexure's endpoint is located at $(0, -d, 0)$, and the flexure's stiffness matrix $[K']$ at the motion axis frame can be found by

$$[K'] = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} [Ad'][K][Ad']^{-1}$$

eq. 6-22

Where $[Ad']$ is the adjoint transformation matrix between the flexure's free end and the motional axis frame, defined as:

$$[Ad'] = \begin{bmatrix} R & 0 \\ DR & R \end{bmatrix}, R = \begin{bmatrix} \cos\theta\cos\varphi & -\cos\theta\sin\varphi & \sin\theta \\ \sin\varphi & \cos\theta & 0 \\ -\sin\theta\cos\varphi & \sin\theta\sin\varphi & \cos\theta \end{bmatrix}, D = \begin{bmatrix} 0 & 0 & -d \\ 0 & 0 & 0 \\ d & 0 & 0 \end{bmatrix}$$

eq. 6-23

Consequently, in the motional axis frame, the reaction wrench $\widehat{W}'$ resulted from a displacement twist $\widehat{T}'$ is then found by plugging $[K']$ and $\widehat{T}'$ into eq. 6-16.

Note that both $\widehat{W}'$ and $\widehat{T}'$ are also defined in the motional axis frame. Therefore, one could substitute $\widehat{T}'$ with a unit rotation or translation vector to calculate reaction forces. The moment M_x reacting about a unit rotation is then computed as

$$M_x = (\cos^2\theta\cos^2\varphi)K_T + (1-\cos^2\theta\cos^2\varphi)K_B + d^2\big((\sin^2\theta\cos^2\varphi)K_A + (1-\sin^2\theta\cos^2\varphi)K_L\big) + \left(\frac{l^2(1-\cos^2\theta\cos^2\varphi)}{4} + dl\sin\varphi\right)K_L$$

eq. 6-24

From this definition, we can find that the rod primarily bends and twists in reaction to the rotation when its axis intercepts the motional axis (i.e., $d = 0$ or $\theta = 0$, see also Figure 6-9A-C). In this case, the rod also falls into the rotation's corresponding constraint space. Noticeably, when the rod is placed outside of the constraint space (i.e., $d \neq 0$) and not parallel with $\hat{a}$ (i.e., $\theta \neq 0$), the rod is subjected to additional translational displacements. In particular, the load increases quadratically proportional to d. Due to the high axial stiffness (i.e., $K_A \gg K_L, K_B, K_T$), the rod becomes acutely more resistant to rotation. On the other hand, θ and φ together determine the tradeoff between the rod's axial and lateral translations and rotations. The rod is primarily subjected to axial rotation (torsion) and lateral translation when it is more aligned with $\hat{a}$, and lateral bending and axial translation become more dominant as the rod's axis $\hat{n}$ deviates from the rotation axis $\hat{a}$. Given the rods' slender aspect ratio, the flexure becomes acutely stiffer against the rotation.

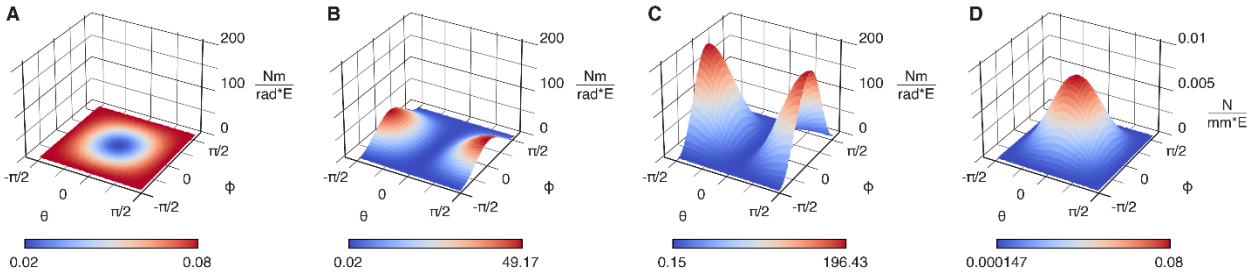

Figure 6-9. Stiffness and design parameter sweep using the analytical model, normalized by the material's elastic modulus. This study assumes the flexural rod to have $r = 1$ mm, $l = 40$ mm, and a Poisson's ratio of 0.3. Subfigure a-c shows the plots for a rotational motion at $d = 0$ (A), 25 (B), and 50 (C) mm, whereas subfigure (D) shows the stiffness plot against a translational motion.

Similarly, the force F_x reacting against a unit translation is computed as

$$F_x = (\cos^2\theta\cos^2\varphi)K_A + (1-\cos^2\theta\cos^2\varphi)K_L$$

eq. 6-25

Where d no longer determines the rod's deformation mode. This observation aligns with the intuition that the free stage's rigid body translation applies the same displacement regardless of the flexure's position. Still, θ and φ determine the rod's deformation modes. When θ and φ are small (i.e., rod aligned with translation axis), the flexure primarily undergoes axial displacement and is stiff (Figure 6-9D). However, when the rod is placed perpendicular to the translational axis (i.e., $\theta = \pi/2$ or $\theta = 3\pi/2$), the stiffness is minimized as K_A is canceled out.

The above equations and intuitions can be leveraged in the design process to adjust the kinematic devices' performance. In particular, taking partial derivatives of the above equations can help to find the parameters required to increase or decrease the device's stiffness. For instance, if the locking effects provided by a stiffness-changing flexure are lower than the design criteria, one may consider increasing its distance to the motional axis to increase the locking effect acutely. Similarly, to increase the locking stiffness against a translational motion, we may orient the rod more perpendicular to the direction of translation, reducing the contribution of the K_A term. In addition to adjusting the flexures' geometric parameters and relative positions, the stiffness of a parallel flexure joint can also be tuned by adding and removing rods. These design decisions are demonstrated in the Device Design section.

It is worth mentioning that a (stiffness-changing) flexure's buckling criterion is also affected by its placement with respect to a motion. When designing a mode to lock a motion, the buckling criterion must be higher than the expected loads. Yet, finding the buckling criterion of a flexure undergoing complex loads requires elliptic integrals [95], and its integration for screw algebra-based compliant mechanism design leads to a highly nonlinear optimization process, making analytical solutions nearly impossible. Hence, we use FE simulations (see next section) to validate a design's buckling behavior against expected loads.

6.11. Finite Element Simulation

We further perform finite element (FE) simulations to predict the nonlinear stiffness of the devices under large deformation and rotation to verify and iterate the design's performance before fabricating them for mechanical tests. For simplicity, the devices are assumed to be made of isotropically elastic material models (Table 6-2). The stiffness-changing flexures are also modeled

as homogeneous bodies without explicit heating wires and epoxy interactions. Yet, they are modeled with an effective modulus representing the combined stiffness of both components. To achieve this, we fabricated 50 mm long samples ($n = 3$) with $OD = 1.5$ mm and 2 mm and subjected them to axial compression under stiffened and softened states to acquire their load-displacement curves (see Appendix 2: Flexural Rod Characterization). The elastic modulus is then calculated from the pre-buckling linear region by normalizing against the rods' geometric parameters. The stages are modeled without through holes and simplified by removing mechanical features that provide minimal structural functions (e.g., wiring guides and service panels). Bonded contacts are applied to the interface between the flexures and the stages to model their connection.

Table 6-2. FE simulation material definitions.

Material	Elastic modulus	Poisson's ratio	Applied to
Polyacrylic acid (PLA)	3204 MPa	0.3	Rigid stages, passive flexures
Formlabs white resin	2800 MPa	0.3	Rigid stages
Aluminum	68 GPa	0.33	6-DOF device jig
Stiffness-changing flexures (OD 2 mm)	2958.4 MPa (RT), 152.02 MPa (54°C)	0.3	Stiffness-changing flexures
Stiffness-changing flexures (OD 1.5 mm)	2668.9 MPa (RT), 480.86 MPa (54°C)	0.3	Stiffness-changing flexures

We used Ansys Mechanical to conduct the FE simulations using the Static Structural implicit solver. We enabled large deflection and automatic time-stepping in the solver controls to accommodate flexure buckling. The device geometries are imported as STEP files and meshed in the Ansys Mechanical interface. Due to their distinct deformation behaviors, we used different mesh settings for the stages and flexures. The stages are meshed with default settings and an element size of 1 mm using tetrahedral (Tet10) elements. Conversely, the flexures are meshed by dividing the circumference into sixteen control points and with a 0.5 mm face-sizing on the cylindrical surface, leading to a hybrid mesh consisting of hexahedron (Hex20) and wedge prism (Wed15) elements. The loads and boundary conditions are applied according to the context of each device's design and mechanical test setup (Figure 6-11, Figure 6-10, Figure 6-12).

Table 6-3 provides a summary of the settings for each simulation model setup. In brief, all models

Device	Load type	Fixed condition	Displacement load	Load
6-DOF device	X, Y-translation	Interface between the jig and the Instron machine clamp: all six DOF	Interface between the jig and the load cell clamp: Fixed DOF: all except for load	Displacement: 1 mm (Unlocked: 0.2 mm/step, locked: 0.02 mm/step)
	Z-translation	Interface between the jig and the Instron machine: all six DOF	Interface between the jig and the load cell clamp: Fixed DOF: XT, YT, XR, YR, ZR	Displacement: 1 mm (Unlocked: 0.2 mm/step, locked: 0.02 mm/step)
	X, Y-rotation	Interface between the jig and the Instron machine: all six DOF	Interface between jig the rotational bearing at the device free end: Fixed DOF: XT, YR, ZR (X rotation); YT, XR, ZR (Y rotation) Free DOF: XR, ZT (X rotation); YR, ZT (Y rotation)	Displacement: 2 mm (Unlocked: 0.2 mm/step, locked: 0.02 mm/step)
	Z-rotation	Interface between the device and the jig: all six DOF	Interface between the device and the loading jig: Fixed DOF: XT, YT, ZT, XR, YR	Displacement: 2 mm (Unlocked: 0.2 mm/step, locked: 0.02 mm/step)
Wearable device (forearm)	Axial rotation (with remote point to specify axis of rotation)	Interface between the fixed stage and the wearer	Interface between the free stage and the wearer: Fixed DOF: XT, YT, ZT, XR, ZR Load DOF: YR	Displacement: 0.6 deg @ 0.02 deg/step (locked), 30 deg @ 0.2deg/step (unlocked)
Wearable device (wrist)	Flexion (with remote point to specify axis of rotation)	Interface between fixed stage and the wearer	Interface between the free stage and the wearer: Fixed DOF: XT, YT, ZT, XR, ZR Load DOF: YR	Displacement: 0.6 deg @ 0.02 deg/step (locked), 30 deg @ 0.2 deg/step (unlocked)
	Deviation (with remote point to specify axis of rotation)	Interface between fixed stage and the wearer	Interface between the free stage and the wearer: Fixed DOF: XT, YT, ZT, XR, YR Load DOF: ZR	Displacement: 0.6 deg @ 0.02 deg/step (locked), 30 deg @ 0.2 deg/step (unlocked)
Wearable device (finger joints)	Rotation (with remote point to specify axis of rotation)	Interface between fixed stage and the wearer	Interface between the free stage and the wearer: Fixed DOF: XT, YT, ZT, YR, ZR Load DOF: XR	Displacement: 0.6 deg @ 0.02 deg/step (locked), 30 deg @ 0.2 deg/step (unlocked)
Haptic thimble	All modes (with remote point to specify axis of rotation)	Interface between the device and the Instron machine: all six DOF	Interface between jig and load cell clamp: Fixed DOF: XT, YR, ZR Freed DOF: YT, XR Load DOF: ZT	Displacement (all load applied at 0.02 mm/step): 0.6 mm([0, 0, 0, 0, 0]), 2 mm ([0, 0, 0, 0, 1]), 5 mm ([0, 1, 0, 0, 1], [0, 1,1, 0, 1], [1, 1, 1, 0, 1], [1, 1, 1, 1, 1])

are applied with a fixed boundary condition at the fixed stage and prescribed displacements at the free end. The simulated force or moment reactions are recorded and compared with experimental

measurements. For the wearable device, the external kinematic constraints created by the human skeleton are modeled as remote rotational pivot points and used by the displacement loads. It is worth noting that during device design iterations, the rigid stages are not fully modeled for complete analysis. We apply a remote rigid body connection between the flexural rods' ends and the boundary condition surface to simulate flexural rod performances.

The FE simulations were used to iterate our designs to make sure they satisfy the targeted design criteria (e.g., stiffness, buckling loads). To iterate a design, we check the simulation results to identify the difference between the current and targeted performance and use the analytical stiffness model to identify the parameters affecting the performance and the gradient of changes. Once a design is modified, we subject the new design to FE simulations to evaluate its improved performance until the targeted criteria are satisfied. Additionally, the FE simulations were also used to iterate and make sure the rigid stages remained sufficiently rigid and stable against the expected loads.

Table 6-3. FE simulation boundary conditions.

Device	Load type	Fixed condition	Displacement load	Load
6-DOF device	X, Y-translation	Interface between the jig and the Instron machine clamp: all six DOF	Interface between the jig and the load cell clamp: Fixed DOF: all except for load	Displacement: 1 mm (Unlocked: 0.2 mm/step, locked: 0.02 mm/step)
	Z-translation	Interface between the jig and the Instron machine: all six DOF	Interface between the jig and the load cell clamp: Fixed DOF: X_T, Y_T, X_R, Y_R, Z_R	Displacement: 1 mm (Unlocked: 0.2 mm/step, locked: 0.02 mm/step)
	X, Y-rotation	Interface between the jig and the Instron machine: all six DOF	Interface between jig the rotational bearing at the device free end: Fixed DOF: XT, YR, ZR (X rotation); YT, XR, ZR (Y rotation) Free DOF: XR, ZT (X rotation); Y_R, Z_T (Y rotation)	Displacement: 2 mm (Unlocked: 0.2 mm/step, locked: 0.02 mm/step)
	Z-rotation	Interface between the device and the jig: all six DOF	Interface between the device and the loading jig: Fixed DOF: X_T, Y_T, Z_T, X_R, Y_R	Displacement: 2 mm (Unlocked: 0.2 mm/step, locked: 0.02 mm/step)
Wearable device (forearm)	Axial rotation (with remote point to specify axis of rotation)	Interface between the fixed stage and the wearer	Interface between the free stage and the wearer: Fixed DOF: X_T, Y_T, Z_T, X_R, Z_R Load DOF: Y_R	Displacement: 0.6 deg @ 0.02 deg/step (locked), 30 deg @ 0.2deg/step (unlocked)
Wearable device (wrist)	Flexion (with remote point to specify axis of rotation)	Interface between fixed stage and the wearer	Interface between the free stage and the wearer: Fixed DOF: X_T, Y_T, Z_T, X_R, Z_R Load DOF: Y_R	Displacement: 0.6 deg @ 0.02 deg/step (locked), 30 deg @ 0.2 deg/step (unlocked)
	Deviation (with remote point to specify axis of rotation)	Interface between fixed stage and the wearer	Interface between the free stage and the wearer: Fixed DOF: X_T, Y_T, Z_T, X_R, Y_R Load DOF: Z_R	Displacement: 0.6 deg @ 0.02 deg/step (locked), 30 deg @ 0.2 deg/step (unlocked)
Wearable device (finger joints)	Rotation (with remote point to specify axis of rotation)	Interface between fixed stage and the wearer	Interface between the free stage and the wearer: Fixed DOF: X_T, Y_T, Z_T, Y_R, Z_R Load DOF: X_R	Displacement: 0.6 deg @ 0.02 deg/step (locked), 30 deg @ 0.2 deg/step (unlocked)
Haptic thimble	All modes (with remote point to specify axis of rotation)	Interface between the device and the Instron machine: all six DOF	Interface between jig and load cell clamp: Fixed DOF: X_T, Y_R, Z_R Freed DOF: Y_T, X_R Load DOF: Z_T	Displacement (all load applied at 0.02 mm/step): 0.6 mm([0, 0, 0, 0, 0]), 2 mm ([0, 0, 0, 0, 1]), 5 mm ([0, 1, 0, 0, 1], [0, 1,1, 0, 1], [1, 1, 1, 0, 1], [1, 1, 1, 1, 1])

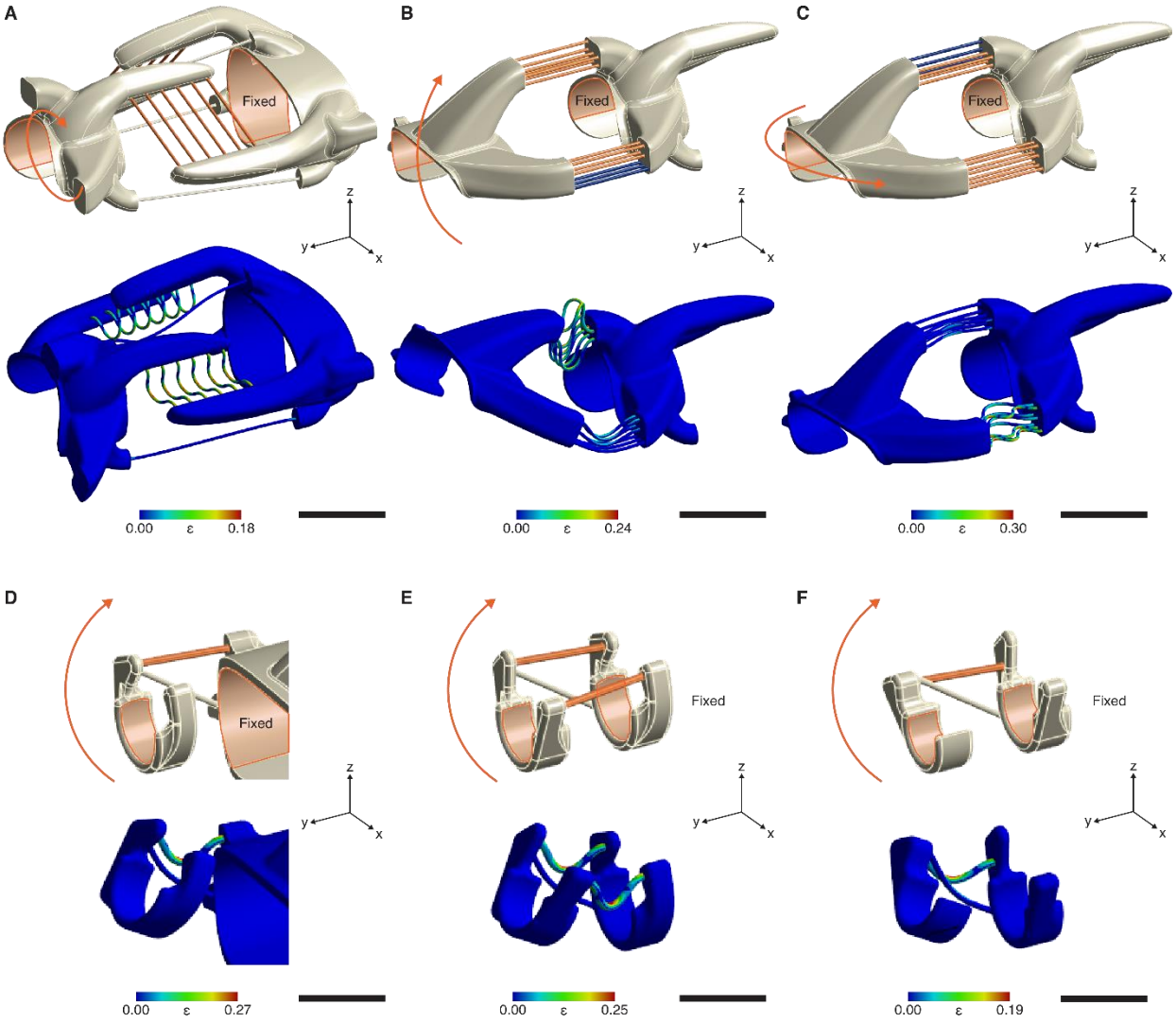

Figure 6-10. Arm-wearable device FEA setup and simulation result. The top rows in each subfigure show the FEA setup. The arrows show displacement load application sites. The bottom row shows the simulation results for each of the joints in their unlocked states: (A) forearm pronation;(B) wrist flexion; (C) wrist deviation; (D) MP flexion; (E) PIP flexion; (F) DIP flexion. Colors indicate material assignment: gray, Formlabs white resin; black, aluminum; blue, stiffness-changing flexures (RT); orange, Stiffness-changing flexures (54°C). Subfigure a-c scale bar, 50 mm. Subfigure d-f scale bar, 25 mm.

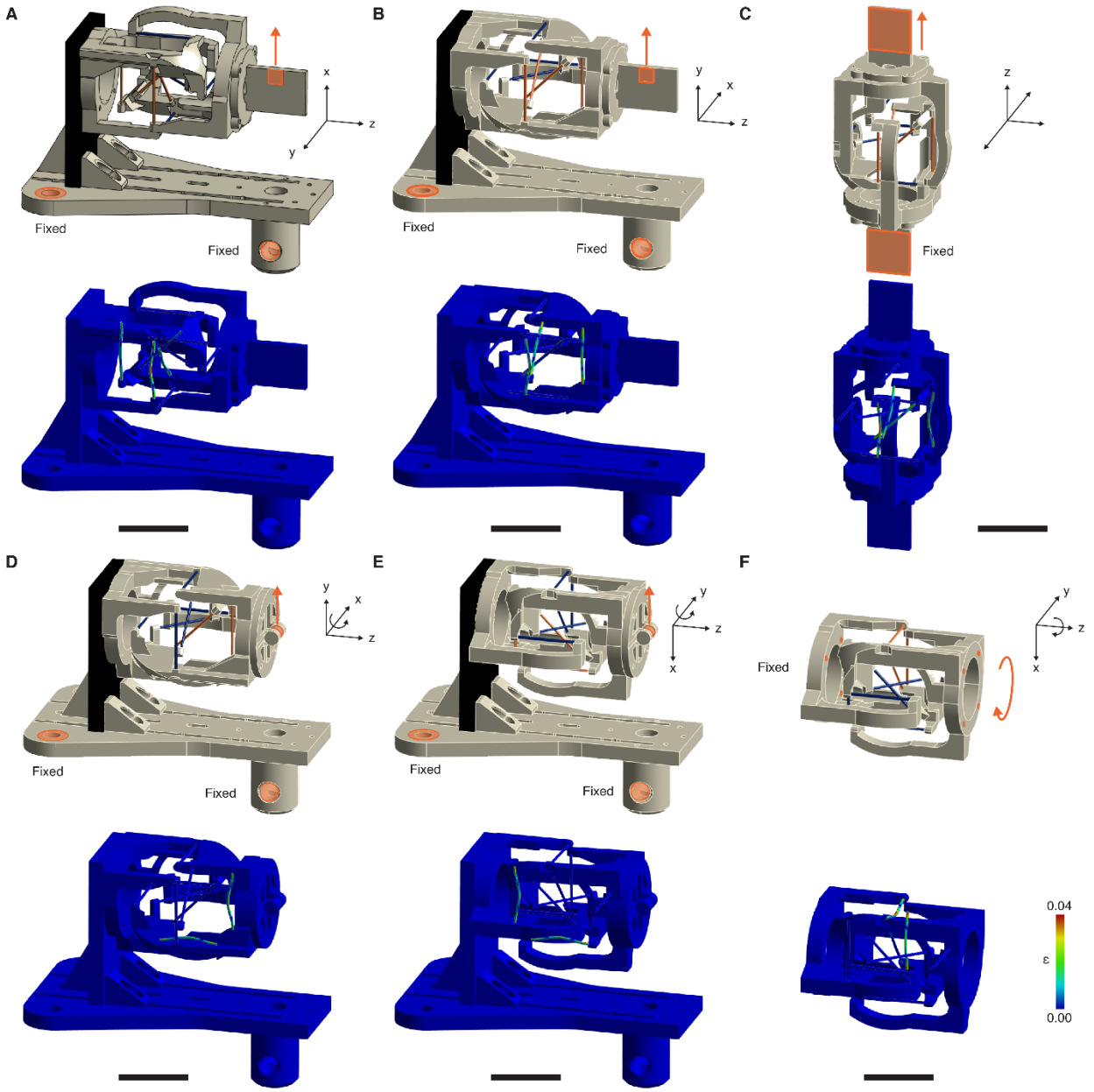

Figure 6-11. 6-DOF device FEA setup and simulation result. The top rows in each subfigure show the FEA setup. The arrows show displacement load application sites. The bottom row shows the simulation results for each unlocked state: (A) x-translation; (B) y-translation; (C) z-translation; (D) x-rotation; (E) y-rotation; (F) z-rotation. Colors indicate material assignment: gray, PLA; black, aluminum; blue, stiffness-changing flexures (RT); orange, Stiffness-changing flexures (54°C). Scale bar, 50 mm.

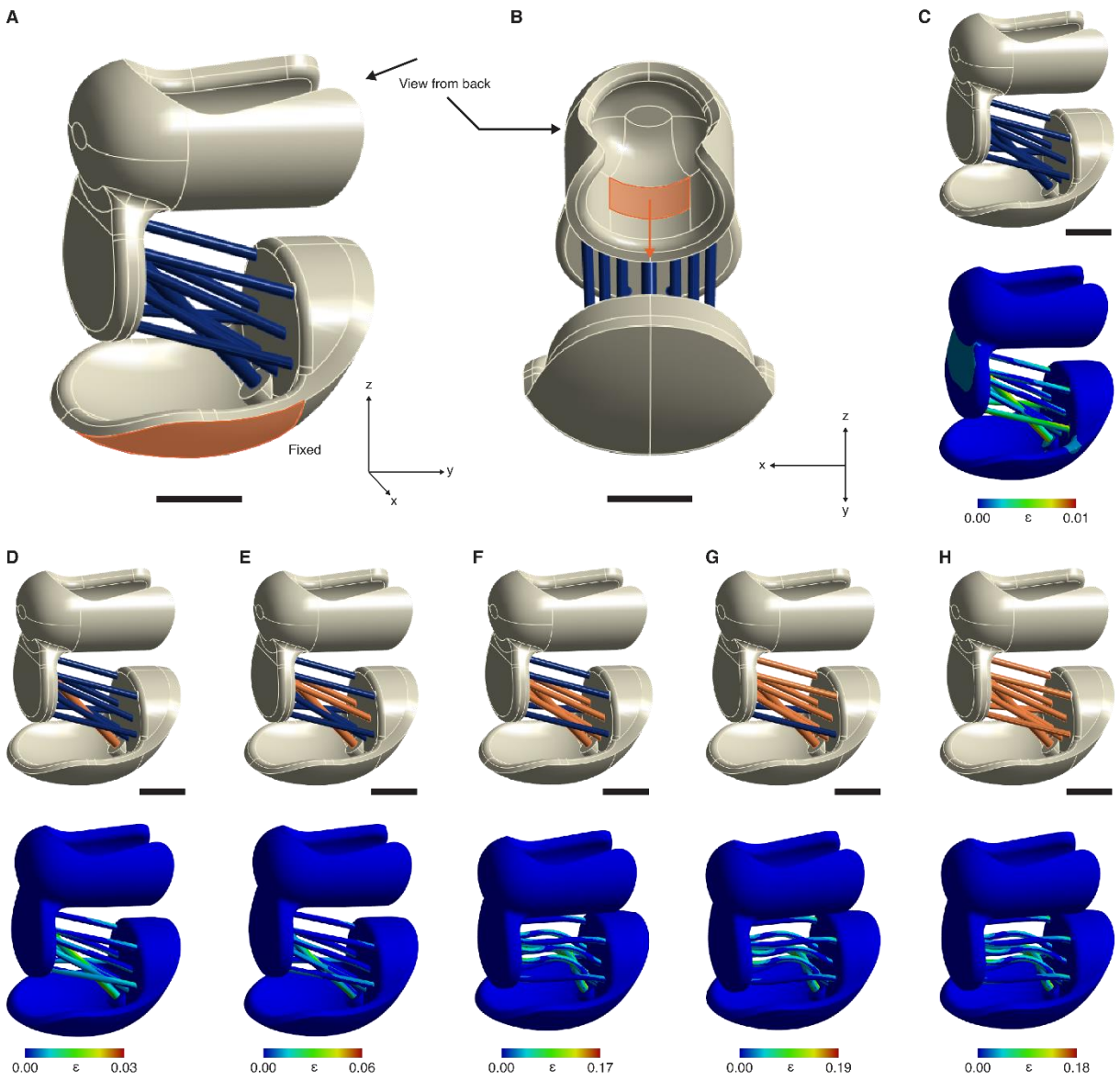

Figure 6-12. Haptic thimble FEA setup and simulation result: (A) the device's fixed-end application, (B) the device's displacement load (negative z) application shown at a different view angle than in subfigure (A). Subfigure (C)-(H) shows the FEA setup (top row) and simulation results (bottom row) in different device configurations: (C) [0, 0, 0, 0, 0]; (D) [0, 0, 0, 0, 1]; (E) [0, 1, 0, 0, 1]; (F) [0, 1, 1, 0, 1]; (G) [1, 1, 1, 0, 1]; (H) [1, 1, 1, 1, 1]. Colors indicate material assignment: gray, Formlabs white resin; black, aluminum; blue, stiffness-changing flexures (RT); orange, Stiffness-changing flexures (54°C). Scale bar, 10 mm.

6.12. 6-DOF Device Design

6-DOF Device: Design Summary

The 6-DOF device was intended to demonstrate the effectiveness of the multimodal kinematics algorithm in identifying flexure placements. Each mode is defined by its kinematic specifications without a stiffness requirement. Therefore, this design example did not use the FE simulation and the analytical stiffness model in its iterations. However, we demonstrate that the algorithm and design rules can help designers create designs with certain qualities, such as avoiding flexure cluttering, symmetry, and more.

6-DOF Device: Rational Design of Flexure Placement

Step 1 & design specifications

The 6-DOF device was designed with a target length of 100 mm. Two rigid stage placeholders were added to signify screw hole positions for connection with test jigs. Six kinematic modes were specified as the input to the algorithm, each enabling a rigid body DOF in the 3D space (Figure 6-13A). The device's center point was set as the rotation pivot and the spatial origin for calculations.

Steps 2 & 3

Figure 6-13B shows the constraint spaces resulting from the rational design algorithm. The kinematic mode specification led to an empty shared constraint subspace per Maxwell's equation of rigid body constraints. I.e., no constraints can be added if all six DOF should be enabled simultaneously. On the other hand, the constraint subspaces shared by the three translational DOF are also invalid since they led to subspaces with zero directional components, and no flexural rods can be placed accordingly.

Step 4

When picking constraint subspaces for placing stiffness-changing rods, we favored designs that require fewer rods and rotational symmetry on the three planes. I.e., the two stages should have identical shapes. In particular, rotational symmetry reduces design complexity as both stages share

the same geometry. The configurations to enable each mode were also identical along the three axes. Moreover, we also intended to avoid flexure cluttering, which complicates the device's assembly and could cause the flexures to collide.

We started by picking the subspaces with the highest cardinality (Figure 6-13C). Yet, as noted, the subspaces shared by the three translations lacked a directional component and were invalid. Therefore, we applied the subspace replacement rule described in eq. 6-12 and divided the invalid subspaces into two with a lower cardinality (Figure 6-13D). This action led to redundancies in selected constraint spaces: the cardinality-5 constraint subspaces were strict subsets of the newly selected ones (cardinality of 4), as they were shared by fewer modes (Figure 6-13E). Therefore, we removed the cardinality-5 subspace that was redundant.

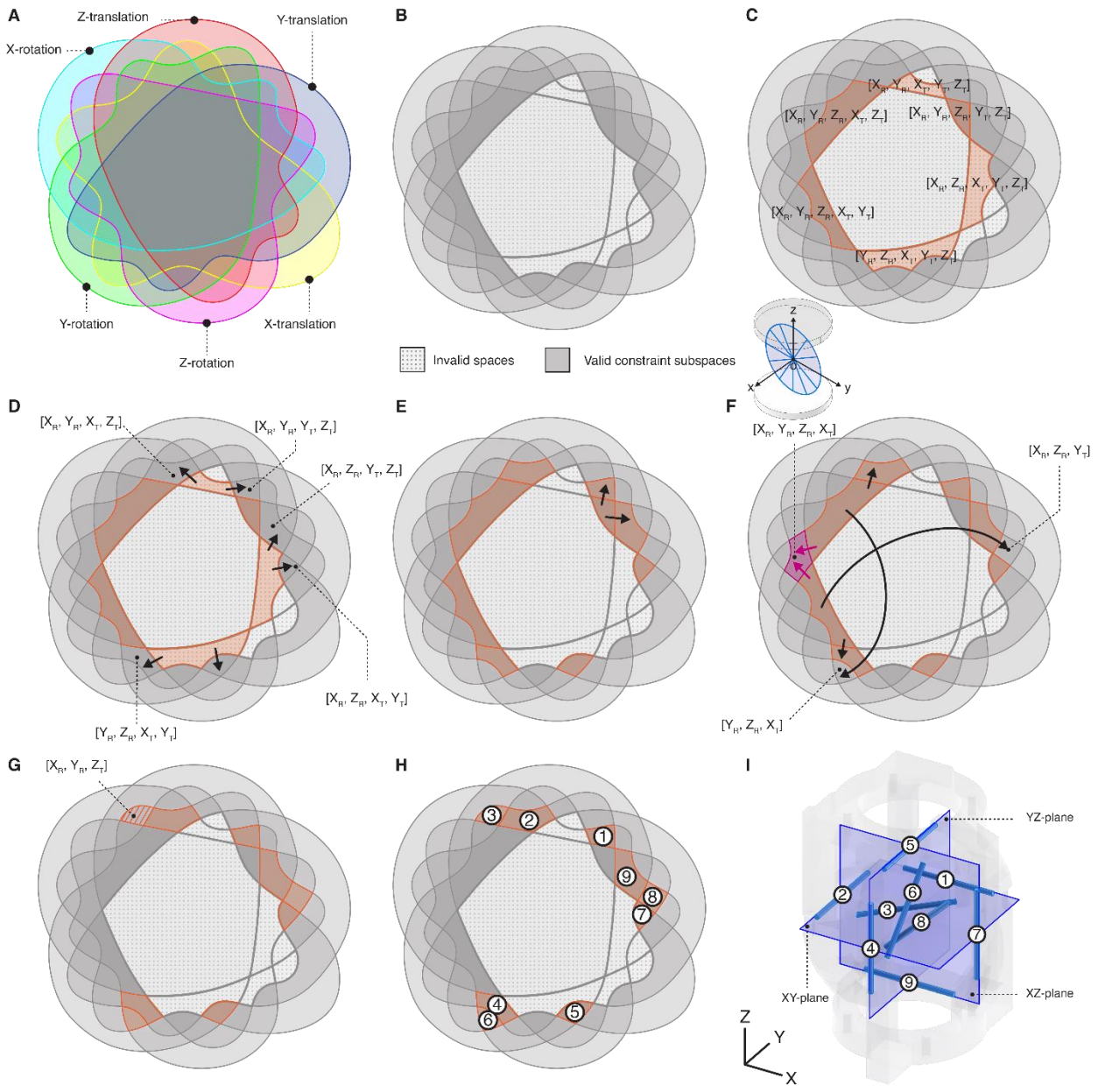

Figure 6-13. The constraint subspace selection process shown in Venn diagram representations. (A) The Venn diagram with labeled constituent constraint spaces. (B) The Venn diagram with colors removed and showing subspaces validity. (C) Initial constraint subspace selection - selecting those with the highest cardinality. (D) Constraint subspace substitution to avoid invalid ones. (E) Constraint subspaces substitution to remove redundancy. (F) Constraint subspace substitution to avoid cluttering. The option that leads to further cluttering is highlighted in magenta and visualized. (G) Constraint subspace selection to maintain symmetry. (H) Placing flexural rods to satisfy constraint subspaces. (I) Final flexural rod layout. The square brackets indicate the kinematic modes intersected into the marked subspace.

Next, we substituted the rest of the cardinality-5 subspaces with eq. 6-13 to maintain symmetry. However, the subspace highlighted in Figure 6-13F presents an undesired option as it may lead to flexure cluttering. I.e., requiring two flexures to pass through the spatial origin. Therefore, we selected two subspaces with a cardinality of three to satisfy the design (Figure 6-13F). Finally, an

additional constraint subspace was selected to maintain symmetry along all three planes (Figure 6-13G), leading to the Venn diagram in Figure 6-13H.

Steps 5 and 6

The constraint subspaces selected in the previous step led to an identical layout on the three spatial planes passing through the spatial origin. On each plane, the stiffness-changing flexural rods' extended axes formed a triangle at the compressive side of motions (Figure 6-13I). All rods were OD 2 mm and 40 mm in length. No redundant flexures were added to this design.

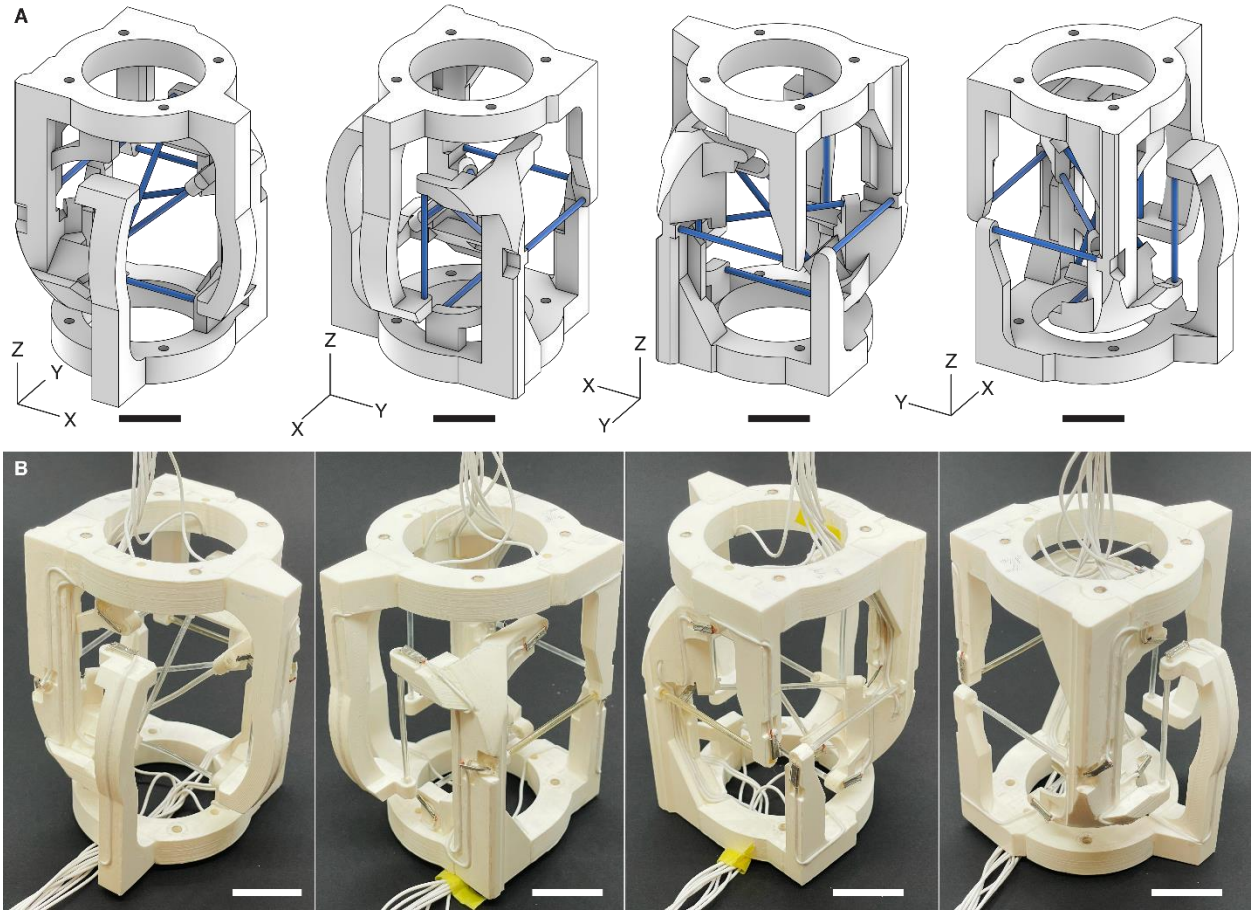

Figure 6-14. The 6-DOF device design. (A) The device design without mechanical features rendered in different view angles. (B) The final, assembled device shown in angles corresponding to subfigure (A). Scale bar, 20 mm.

Device Design

The fixed and mobile stages were designed after placing the rods and had identical shapes due to the flexures' rotational symmetry (Figure 6-14A). The models had pre-defined bolt holes (3mm diameter M3 screw) at the base to connect to mechanical testing jigs, and structural arms extended

from the base and met the stiffness-changing rods at the through holes housing them. All through holes had a depth of 5 mm. We added grooves to house the conductive wires connecting the rods' resistive heating wires. Specifically, the grooves had a diameter (0.5 mm) matching the 26-gauge conductive wires, and the wires were glued to the stages to prevent delamination.

The stages were printed using the FDM printer as three separate parts (Figure 6-14B). We oriented the parts to ensure the layering direction was perpendicular to the length of the structural arms, which helped avoid layer delamination when the devices were loaded.

6.13. Wearable Haptic Device Design and Iteration

Design Process Summary

The wearable devices were targeted at an application scenario where kinematics, stiffness, and motional resistance are essential to the device's performance. Therefore, the algorithm, FE simulation, and the analytical stiffness model were used to iterate the design. At a high level, we started by setting design criteria based on human perception and physiological literature. Next, we took the kinematic specifications and applied the rational design pipeline to identify flexure placements (i.e., constraint subspaces) required for each joint. Lastly, we used FE simulations to evaluate the design's performance and the analytical stiffness model to strategize flexure geometry modeling and adding redundant flexures. Specifically, we iterated between FE simulation and modifying flexure parameters until the targeted performance was satisfied. In short, there are three steps to design a reconfigurable joint:

Step 1: Defining design criteria based on human perception and physiology.

Step 2: Identify flexure placements using the multimodal kinematics design algorithm.

Step 3: Iterate designs using FE simulation and the analytical stiffness model until the targeted performance is reached.

Arm-wearable: Design Criteria

This design was aimed to create a device worn at the human forearm, wrist, and finger to provide individual DOF locking and locking depending on the application context. Table 6-1 summarizes

human perceptual and physiological performances at each of the joints. We opted these joints as they are located at the are the most prominent joints associated with kinesthetic perception[186, 239].

We based our arm torque specifications on Gupta et al.[89] The torques at each joint were set as a fraction of the human isometric strength. In particular, forearm supination and wrist rotations were set at 50% and 25% of a healthy adult's isometric strength. On the other hand, the finger joints' isometric strength was based on Milner et al.[186] and directly used as the device design criteria. Finally, the proprioception rotational just noticeable difference (JND) was based on Reissner et al.[239]

Arm-wearable: Rational Design of Flexure Placements

The arm-wearable device contained five joints: forearm pronation, wrist flexion and deviation, metacarpophalangeal (MP), proximal interphalangeal (PIP), and distal interphalangeal (DIP) joints. We designed the joints individually and combined them in series to form the final device. To design the arm-wearable device, we took measurements of the wearer and modeled the DOF axes in 3D model software. Next, placeholder geometries (i.e., plain shells) were placed at the opposite end of the joints, and the spaces between them were designated for placing flexures.

Forearm and finger joints (1-DOF rotation)

The forearm and finger joints have the same kinematic DOF, albeit with a different orientation. The forearm's rotation axis runs along the length of the forearm, whereas the finger joints' rotation axes run perpendicular to the finger. Still, we designed these joints to have two kinematic modes - locked (0-DOF) and unlocked (1-DOF of rotation). The constraint subspaces calculated by the algorithm are shown in Figure 6-15 as a Venn diagram. It is worth noting that the shared constraint subspace was the same as the unlocked mode; thus, no stiffness-changing flexures were needed to instate the unlocked mode. On the other hand, at least one stiffness-changing flexure was needed to instate the locked mode with 0-DOF. Precisely, the stiffness-changing flexure's extension line must not coincide with nor be parallel to the rotation axis.

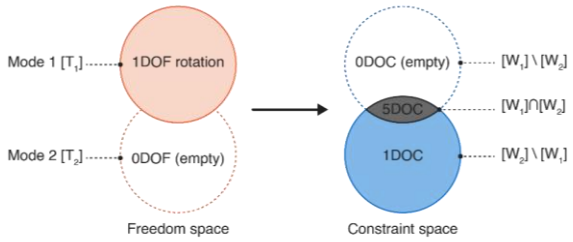

Figure 6-15. Arm-wearable device kinematic mode specification and Venn diagram (1-DOF rotation). This specification was applied to the design of the forearm and finger joints.

Wrist joint (2-DOF rotation)

The wrist joint comprises two rotation axes perpendicular to the length of the arm. Wrist flexion-extension denotes rotations where the palm's trajectory is perpendicular to itself, whereas the wrist's ulnar deviation denotes rotations where the palm's trajectory is in plane with itself. We defined two kinematic modes as input, each enabling a rotation axis. The resulting constraint subspaces are shown in the Venn diagram provided in Figure 6-16. We note that in this design, in addition to the two constraint modes, it was also possible to turn both rotations on or off simultaneously by softening or stiffening all stiffness-changing flexures, respectively. Therefore, the device was capable of a total of four kinematic modes.

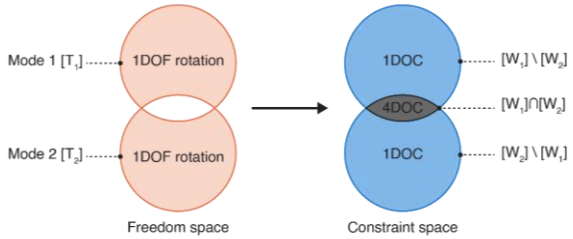

Figure 6-16. Arm-wearable device kinematic mode specification and Venn diagram (2-DOF rotation). This specification was applied to the wrist joint design.

Arm-wearable: Flexure Design Iterations

After identifying the flexure placements using the algorithm, we replaced the shared constraint subspace with the skeletal joint. The skeletal joint readily and exactly constrained the joint to have the fully unlocked DOF. Still, additional passive flexures were added to the forearm and finger joints to maintain the spacing between the two stages. We note that in these device designs, there was no need nor freedom to select different subsets of constraint subspaces since the ones resulting from the algorithm were all viable and efficient. Nevertheless, we added non-redundant flexures

to each device (V1 in Figure 6-17**,** Figure 6-19, and Figure 6-21) to initiate design iteration toward the design criteria.

Forearm joint

The forearm pronation joint design iteration (Figure 6-17) began with a stiffness-changing flexural rod between the stages (V1), which yielded a much lower buckling plateau than the expected load (5 Nm) in the FE simulation. We then added more flexures (V2) by rotating and copying the stiffness-changing flexure twelve times about the forearm pronation axis. Yet, the stiffness was still low as the rods were somewhat oriented parallel to the rotation axis (i.e., θ was too small). Thus, we oriented the flexures to become more perpendicular to the rotation axis to arrive at the final design (V3 and Figure 6-18).

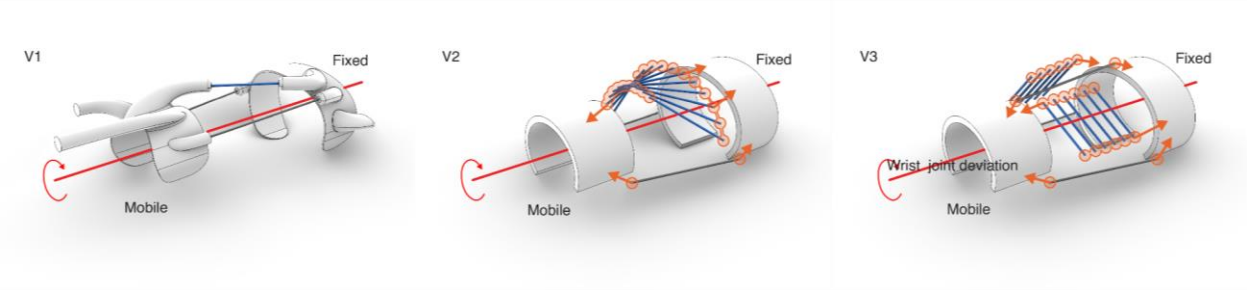

Figure 6-17. The forearm joint's design iteration. The red axis marks the kinematic DOF (forearm pronation). The orange marks and arrows indicate the remote rigid connection applied in FE simulations.

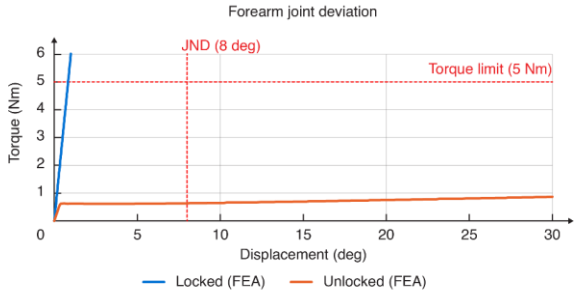

Figure 6-18. The final forearm joint design performance evaluation using FE simulation.

Finger joints

The finger joints shared an identical layout (Figure 6-19). Both passive and stiffness-changing flexural rods were added at an angle with respect to the axis to avoid cluttering around their endpoints, and a passive flexure was added between the stages to restrain the two stages' relative position (Base). Since each joint had a different max torque requirement, we modulated the

stiffness-changing rods' distance from the rotation axis to adjust their resistance against the motion, arriving at the final design. Due to the low torque limit at the finger joints, the passive flexures had a reduced diameter of 1.5 mm. Finally, the MP joint's torque limit is 2.35 times higher than the others, and we mirrored and copied the stiffness-changing flexure about the rotation center to create a design with higher stiffness (Figure 6-20).

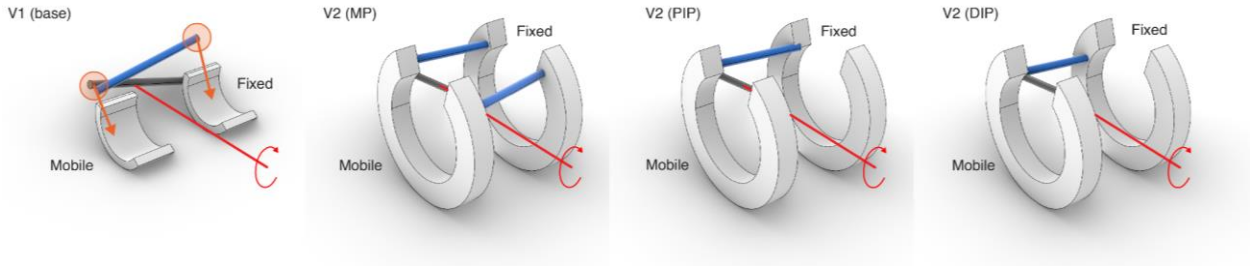

Figure 6-19. The finger joints' design iteration. The red axis marks the kinematic DOF. The orange marks and arrows indicate the remote rigid connection applied in FE simulations.

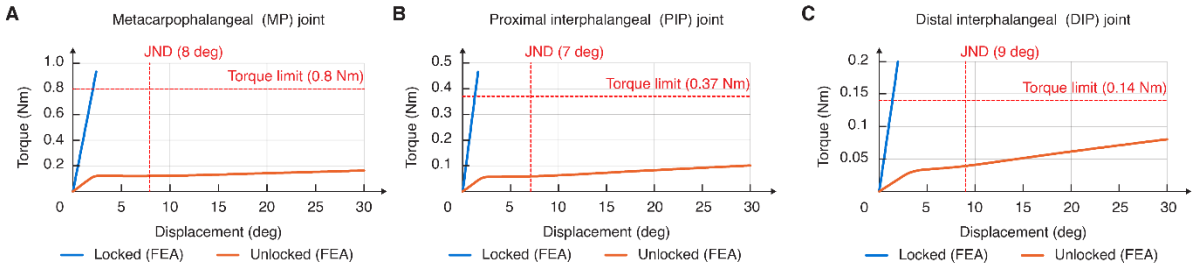

Figure 6-20. The final finger joints design performance evaluation using FE simulation. (A) the MP joint; (B) the PIP joint; (C) the DIP joint.

Wrist joints

We explored different flexural rod layouts according to the algorithm's output (Figure 6-21). No passive flexures were added to this design to avoid cluttering and collision. Several design variations (V1-V4) were explored, but they required long structural arms connecting the stages and rods, which led to increased material usage, weight, and mechanical integration challenges. Consequently, we opted for a design (V5) that required less structural connection. Next, we adjusted the number of rods to ensure the device had effective locking effects against the expected torque (V6, Figure 6-22).

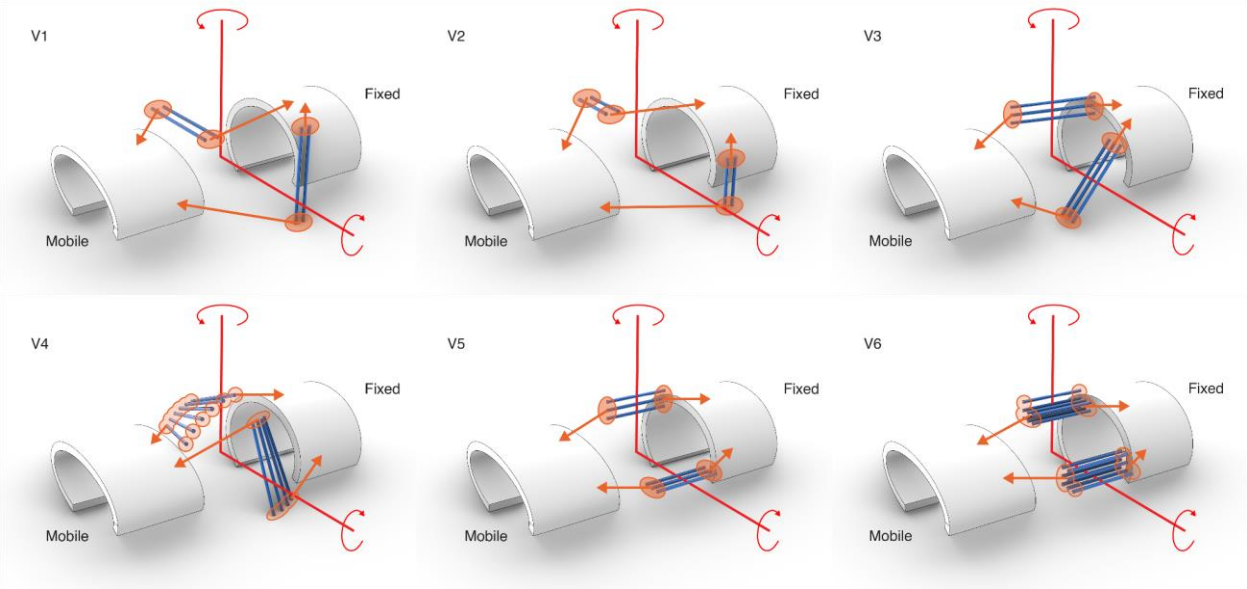

Figure 6-21. The wrist joint's design iteration. The red axis marks the kinematic DOF. The orange marks and arrows indicate the remote rigid connection applied in FE simulations.

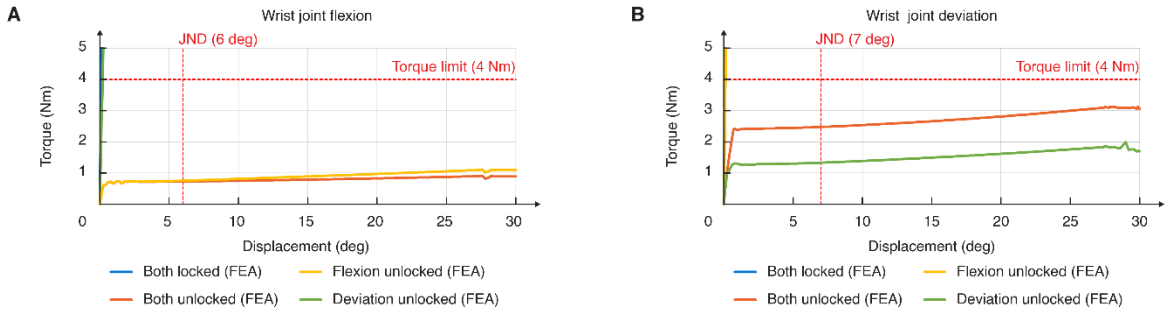

Figure 6-22. The final wrist joint design performance evaluation using FE simulation. A, evaluation against wrist flexion. B, evaluation against wrist deviation.

Arm-wearable: Device Design

Once the flexure layout at each joint had been decided, we collated all joints into one file to model the rigid stages (Figure 6-23A). Structural arms extending from the stages - modeled as cuffs around the arm - connect and house the flexure endpoints. The stages were modeled to be slightly smaller than the wearer's arm circumference so that the device would gently wrap around and cling to the wearer's body without requiring additional locking mechanisms. FE simulations were used to iterate the rigid stages to minimize their deformation within the joint torque limit. The device was fabricated using the SLS printer. Due to the printer's size constraint, the stages were divided into smaller parts for printing (Figure 6-23). Figure 6-24 through Figure 6-26 further shows the wearer exercising each joint under their unlocked modes.

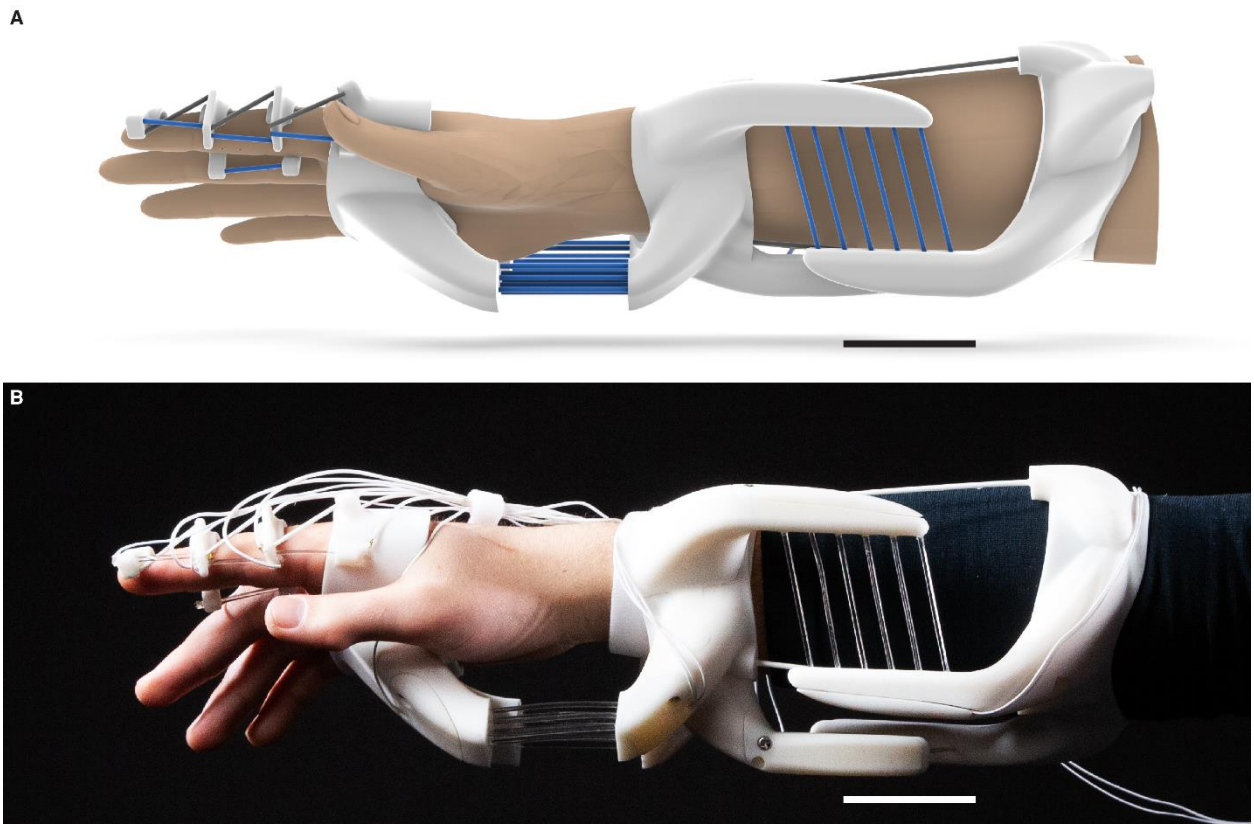

Figure 6-23. The arm-wearable device design. (A) A render of the device design without mechanical features. (B) The final, assembled device. Scale bar, 50 mm.

Figure 6-24. Pictures of the arm-wearable device's forearm joint in the unlocked state. Scale bar, 50 mm.

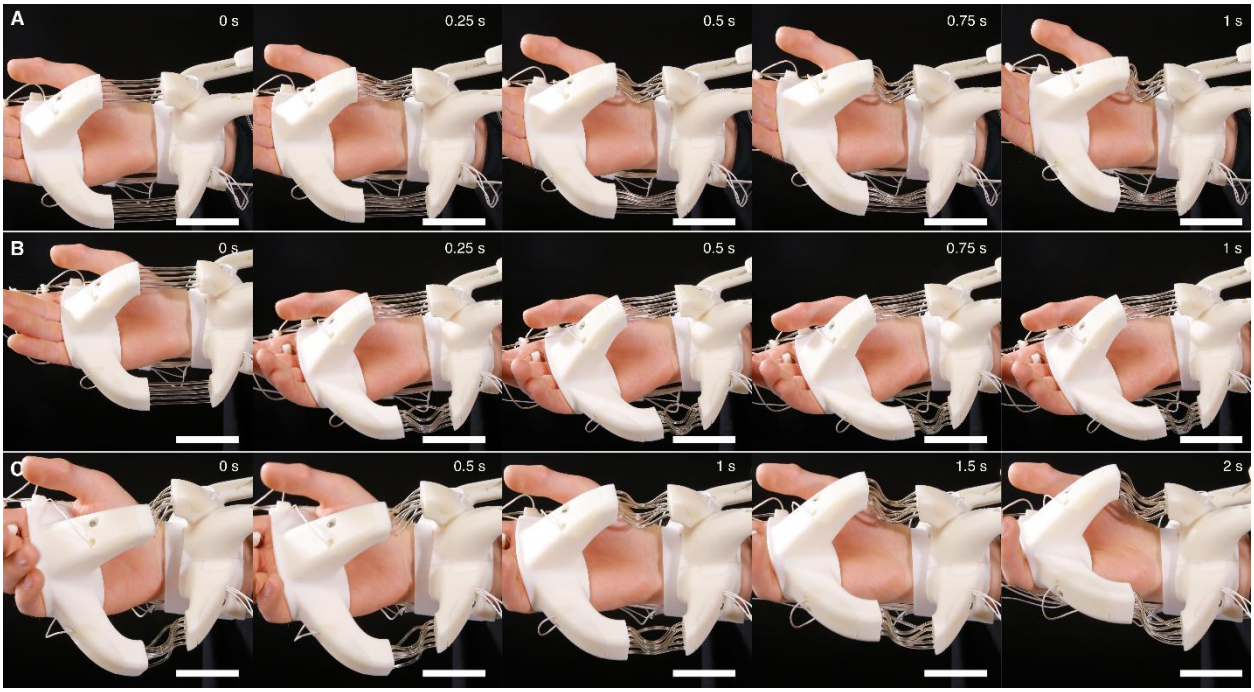

Figure 6-25. Pictures of the arm-wearable device's forearm joint in different unlocked modes. (A) The wearer exercises wrist flexion in the flexion-unlocked mode. (B) The wearer exercises wrist deviation in the deviation-unlocked mode. (C) The wearer exercises both DOF in the both-DOF-unlocked configuration. Scale bar, 50 mm.

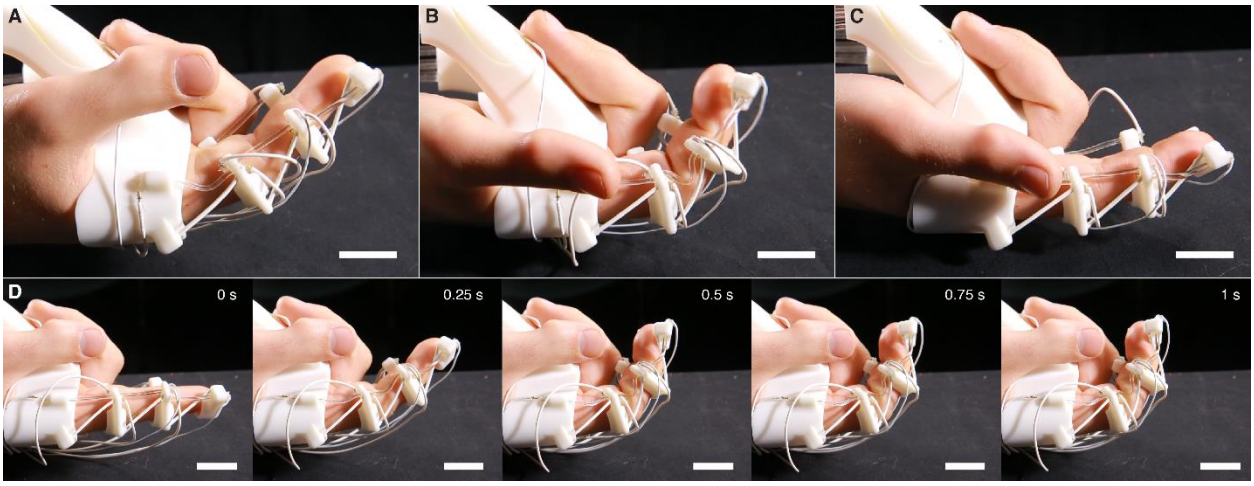

Figure 6-26. Pictures of the arm-wearable device's finger joints in different unlocked modes. (A) The wearer exercises MP flexion in the MP-unlocked mode. (B) The wearer exercises PIP flexion in the PIP-unlocked mode. (C) The wearer exercises DIP flexion in the DIP-unlocked mode. (D) The wearer exercises all finger joints with all joints unlocked. Scale bar, 20 mm.

6.14. Haptic Thimble Design

Haptic Thimble: Design Criteria

The haptic thimble device was designed to fit the index finger, and the design criteria were based on the forces and compliance discrimination of the fingertip in an explorative task (i.e., using fingers to explore the softness of a surface). Zoeller et al.[360] reported that a healthy adult's index finger could exert a compressive force of 25N in softness explorations, and the just noticeable difference in compliance is 0.05 mmN^{-1}, which converts to a stiffness of 20 Nmm^{-1}. In our design, we assumed the fingertip applied forces over a 10 x 10 mm^2 surface area at the mobile stage, and the flexures were confined in a 20 cubic millimeter space to ensure the device had a small footprint.

Haptic Thimble: Rational Design of Flexure Placements

The two kinematic modes were defined as one with 6-DOF and the other with 0-DOF (Figure 6-27). This specification led to an empty shared constraint subspace without passive flexures. However, the stiffness-changing flexures should create a full-ranked linear screw space.

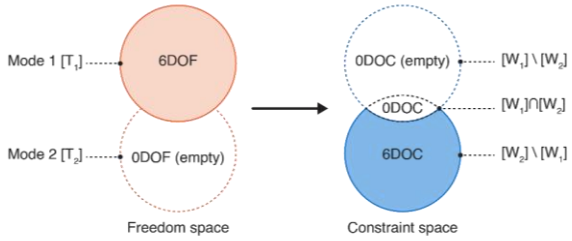

Figure 6-27. The thimble device's kinematic mode specification and Venn diagram.

Haptic Thimble: Flexure Design Iterations

Flexural rods (OD 1.5 mm) were added to the 20 cubic millimeter space in mirror symmetry about the central transverse plane. Figure 6-28 shows the flexural rod layout iterations. We started by adding six flexural rods to the design, yet the stiffness was insufficient to fully lock the device in the stiffest mode under a 25N load. Therefore, we added more rods in the following design iterations. Note that a pair of flexures with a small θ angle was added to provide high stiffness in the locked state. However, they also caused the flexures to become cluttered and prone to collision when bent (which may cause heater transfer). We arrived at the final design (V6) by spacing out

the flexures and replacing the vertical flexure pairs with a single rod located at the central plane with a larger diameter (OD 2 mm) to increase its stiffness against axial displacement loads.

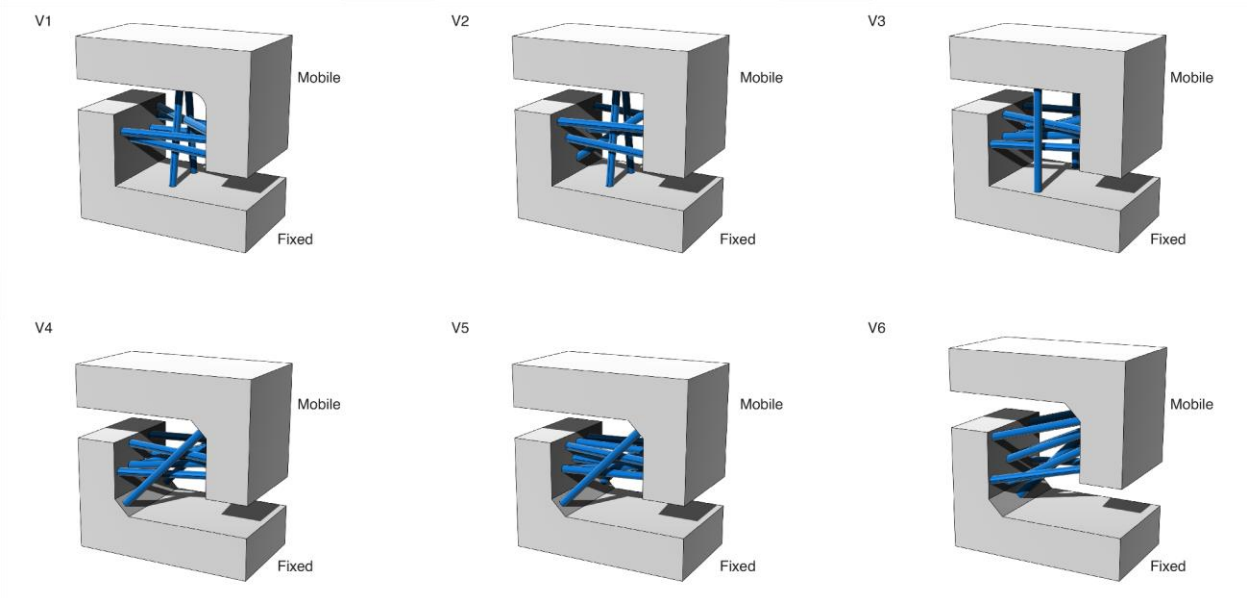

Figure 6-28. The thimble device's design iteration.

Haptic Thimble: Device Design

Housing through holes, wiring chambers, and service panels were added to the thimble following finding flexural rod placements (Figure 6-29). We added a cylindrical void to house the finger at the mobile stage. The fixed stage was modeled into a smooth, round surface so that the device could come into contact with an external surface at different angles. The device was printed using the SLS printer; both stages are printed as a singular part.

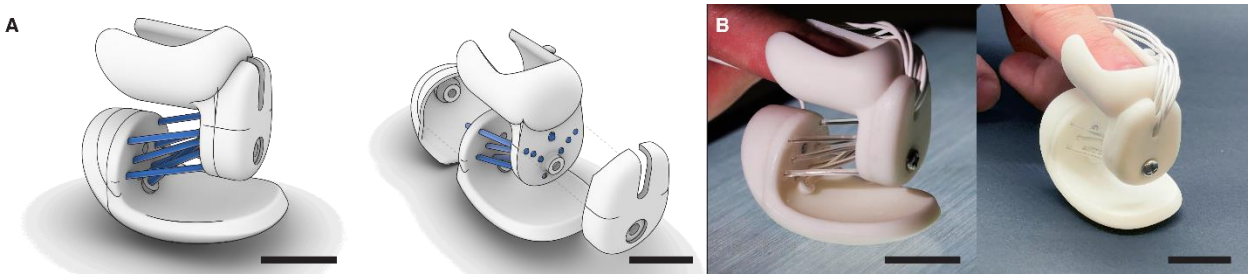

Figure 6-29. The Haptic thimble device design. (A) The device rendered in different angles. (B) The final, assembled device shown in angles corresponding to subfigure (A). Scale bar, 20 mm.

6.15. Computational Design Implications

The algorithmic extension enables users to explore strategies to *combinatorically* find compliant flexure placements that satisfy a targeted kinematic function. We highlight that this differs from the literature that explored design alternatives in a continuous numerical space [127, 259, 260]. While numerically varying designs could be conveniently clustered or filtered by Pareto fronts to reveal their design tradeoffs against numerical objectives, these methods do not apply well to combinatorial design problems that pertain to qualitative or binary metrics (e.g., viability, collision), such as the case for kinematically reconfigurable flexures, modes, and devices. Instead, the algorithm helps designers find alternative solutions by using production rules (heuristics) to attain new viable design plans that satisfice the kinematics design objectives.

Previously, ReCompFig's tool was implemented to find a single flexure layout plan that satisfies the functional goals. While this limitation is less apparent in design problems that involve few (i.e., less than three) kinematic modes, it becomes critical in cases where large numbers of kinematic modes are involved (e.g., the 6-DOF device). We note that the improvement is substantial as it allows us to create designs that would have been deemed impossible by the ReCompFig design tool, such as the 6-DOF device that could be reconfigured to provide any of the six kinematic freedoms in the 3D space. The heuristics provided here are essential for converting an unviable flexure layout into a valid, implementable solution.

Additionally, exploring alternative flexure layout plans helps navigate design problems involving device or mechanism integration and attachment. For instance, in the wearable device examples, exploring different flexure layout plans is critical to avoid collision or geometric conflict with the human body. Factors like ergonomics and flexible device placements on the human body are difficult to convey to a CAD tool and, thus, must be considered by the designers themselves. The heuristics presented in this work could help designers attain these qualities while circumventing invalid designs. Other heuristics also established rules for resolving flexure redundancy across different flexure groups, allowing designers to further simplify designs.

On the other hand, the analytical stiffness model and FEA aim to support designers to address factors not previously considered by the ReCompFig tool. While the ReCompFig tool helps users model CM flexures while respecting their kinematic goals, it does not help users reason about

flexure placement parameters (with respect to kinetic performance). Moreover, there is also a gap in knowing how the design performs and how to modify it toward targeted performance. These are critical challenges designers face while designing to meet realistic requirements.

The newly added stiffness model and FEA address these needs by helping users assess and improve their designs given kinetic goals. They each exemplify two distinct types of tools in a computational design process. While FEA is accurate and regarded as the ground truth in various engineering fields, their computational expense makes them less ideal in an inverse design process. Moreover, while FEA could help users assess their design's performance, they do not inform design modifications. Alternatively, the analytical stiffness model is fast to compute and parameterizes stiffness requirements with respect to flexure placements (i.e., using human-interpretable parameters), therefore effective at helping designers to understand and adjust flexure layouts (i.e., inverse design or (micro-)optimization design processes). However, they do not consider physical nonlinearity and should be used with more accurate tools, such as FEA, to verify designs.

6.16. Discussion and Conclusion

In this work, we present a reconfigurable metastructure design concept, as well as a rational design pipeline that enables customizing devices targeted for different use contexts, kinematic reconfigurations, and stiffness demands. The design pipeline first employs an enhanced FACT design algorithm for calculating flexural rod placements given multiple reconfigurable kinematic modes. FE simulations and an analytical model are then used in conjunction to evaluate and iterate the designs toward target kinematic performances. With this design strategy, we designed a generalized reconfigurable metastructure-based device that can selectively lock and unlock motions along any of the six DOF. Mechanical test results show that the devices have statistically distinct load-displacement performances between different modes. A wearable device tailored for the arm, hand, and fingers is also provided to demonstrate our method's effectiveness at addressing kinesthetic demands, as well as the enabled design space. In particular, the devices can provide stiffness levels appropriate for wearable kinesthetic contexts. The stiffness change between the locked and unlocked states along a DOF is also sufficiently large given human kinesthetic

perceptual ranges. Finally, a haptic thimble exemplifies two magnitudes of stiffness change, showing the system's ability to render distinct haptic feedback.

The presented design principles can be generalized and applied to different body locations to provide kinesthetic feedback. The devices could also be digitally controlled and interfaced with a computer to provide kinesthetic experiences when interacting with virtual or augmented realities [72, 212, 261], on-demand body training and assistance [274], or just-in-time personal care by adapting their functions [262, 271]. To scale up production, fabricating the entire device through a unified digital fabrication process (e.g., embedded printing of epoxy [8]) could help to reduce labor demands, assembly complexity, and navigate a larger design space. Combining the rational design pipeline with design optimization toward more diverse objectives such as size, weight, form factor, and energy consumption will also be important in future studies to make the devices more wearable, friendly to use, and resilient for daily life usage.

6.17. Computational Toolmaking Remark

In this work, we expand the kinematically reconfigurable compliant mechanism design algorithm to allow designers to explore different ways of planning flexural layouts to attain a satisficing design. This expansion, in turn, augments designers to tackle more complex design problems that would have been deemed impossible by the ReCompFig tool. Additionally, providing a computational method for negotiating alternate design strategies also allows designers to circumvent design problems that arise from designing for contextualization, such as collision issues in wearable device design. The added analytical stiffness model and finite element simulation further inform flexural modeling concerning device kinesthetics.

This research also prompts us to reflect on computational toolmaking for active materials design. In addition to their stewardship in designing toward satisfaction, design tools may allow designers to negotiate different ways of solving a problem. Prior iterations of suggestive design tools ([108, 144, 298, 302], also includes SimuLearn [337] and ReCompFig [335, 337]) are often focused on guiding the user through the modeling process but put less focus on the generation of suggestions themselves. However, design algorithms often present open-ended decision points when planning for guidance, creating an opportunity for users to intervene and trade between different ways of

completing a task. Exposing these opportunities in a design tool may give users more flexibility in finding a solution. This notion then informs the next chapter's computational toolmaking, where we allow designers to not only iterate toward a satisficing design but also provide means to negotiate different strategies for solving an even more complex CM problem.

Chapter 7. Interconnected Compliant Mechanisms with Active Materials Integration

7.1. CAD Toolmaking Motivation

In this work, we scale up the scope of compliant mechanisms design from designing single joints to designing assemblies of joints – a combinatorial design of combinatorial design problem. Such mechanisms could be designed to have certain computing capabilities by transmitting forces and displacements. Active materials could also be incorporated into these structures to make devices with input, output, and structural reconfiguration functionalities. However, the design complexity of such devices also increased with the interaction between connected joints, making the design space even more nonlinear and vast than single-joint design scenarios. Satisficing designs are scattered around a design space of variable dimensions. It could be unintuitive to iterate toward a viable solution, let alone iterate for varying solutions.

In this work, we develop a CAD tool for attacking such design problems. Given the design objective, the tool helps designers iterate toward an interconnected assembly topology as a "modeling plan," which could then be parsed into flexural modeling instructions for the user to follow. The design tool relies on a numerical solver to evaluate the viability of a given topology and uses several design heuristics to iterate the design. Naturally, such a design tool could afford both forward and inverse design workflows. However, in this work, we also expose the heuristics decision process to the user and allow them to negotiate different ways to modify the topology in a suggestive workflow. In these negotiations, designers could choose to directly iterate the design themselves, bypassing the design tool's suggestions, pick from the design tool's suggested actions, or alter the design tool's (i.e., the heuristic engines') decision-making behaviors to steer the formulation of design plans. This feature, in turn, provides designers more control when collaborating with a CAD tool to find design solution(s).

7.2. Technical Motivation

Interconnected Compliant Mechanisms as Computing Substrates

A single compliant mechanism joint consisting of two rigid bodies connected by a group of flexures could be regarded as an articulated mechanical joint, where the free end could displace against the fixed end in motional freedoms prescribed by the flexure layout. Forces and loads subjected to the free end could lead to different mechanical responses depending on their types and kinds with respect to the kinematic DOF and constraint DOC spaces created by the flexure arrangement. If a load is applied along the joint's DOF, the free end will conform to the load and displace with small resistance (i.e., stiffness). Conversely, CM joints have magnitudes higher stiffness against loads along their DOC, leading to minimal displacements and allowing loads to pass along the flexures. Using a CM knee joint replacement as an example [88], torsional loads about the DOF axis would cause the joint to rotate while loads (e.g., body weight) along its DOC lead to minimal displacement and allow forces to pass between rigid ends.

Such binary behaviors could be viewed and leveraged for signal processing and create mechanical computation. Multiple CM joints can be assembled into a network of joints to become the substrate for transmission and computation. In this scheme, loads and displacements applied to a rigid body in the structure are transmitted within the structures to drive a distal body's movement, creating an input-output correspondence. Moreover, signals could be relayed, negated, or transformed among the flexures to create complex relationships between multiple input and output points. For instance, Hopkins et al. used a network of joints to create a multi-DOF precision motion stage [100, 101]. The device is driven by multiple piezoelectric actuators, whose displacements are conducted through the flexural joints to exercise the end effector. Moreover, the flexures are arranged to isolate actuators from shear loads induced by asymmetric actuation (i.e., loads induced by other actuators moving at a different amplitude). In particular, while actuation loads are allowed to pass through the DOC of the "limbs" connecting an actuator to the end stage, the displacements caused by other actuators would be negated by displacement along the DOF contained within the limb and minimize shear load at the actuator end. Inspired by modern computer architecture, networks of CM joints could also be designed to mimic the functions of digital logic gates [273] or artificial

neural networks [146] to embody programs and algorithms, which in turn allow mechanical structures to dynamically respond to the environment and learn and adapt to its context.

In addition to the expanded functional design space, interconnected CMs also provide utility benefits, including reduced fabrication cost, better resilience and robustness against hazardous environments, and greater customizability of forms. Augmented by advances in additive manufacturing and digital fabrication, the ability to design and fabricate kinematic structures monolithically could reduce assembly efforts, manufacturing complexity, or even device weight and footprint by reducing the number of parts. These features make them an ideal mechanical design paradigm for contexts that are sensitive to weight (e.g., spacecraft thruster gimbal design [185]) or favor manufacturing simplicity (i.e., 3D-printable gadgets for open-source maker communities [301]). Notably, interconnected CMs are also more customizable for different form factors than a single joint. A single-joint CM design requires active material actuators and sensors to be placed directly at the joint, which could lead to mechanical design issues, including distal mass at limbs or components scattered around the kinematic structure. Alternatively, in an interconnected CM, actuators and sensors could be placed at a central location to avoid distal mass, and clustered actuators and sensors also simplify the wiring needed to connect components. The freedom of placing components at arbitrary places also provides designers the flexibility to adapt the structure for bounding volume or device footprint constraints.

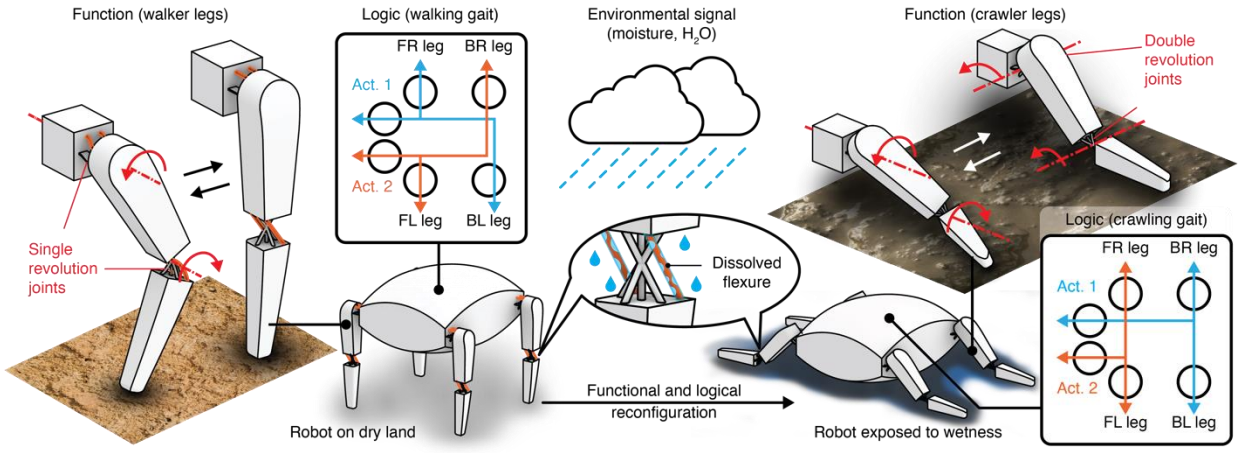

Figure 7-1. Integrating active materials into compliant mechanisms. (A) A robot integrated with such materials can adapt its functions according to the environment. The robot walks on four legs on dry land with one degree of rotation at each joint, and the gait actuates legs as diagonal pairs. When exposed to wetness, certain flexures will dissolve and change the limb functions and gait. The robot lies on a wet surface with increased contact area to avoid sinking and moves forward actuating legs in front-back pairs. The gait reconfiguration could be achieved by using materials that soften or harden when exposed to water, resembling clutches (i.e., transmit forces only when hardened/engaged). (B) Example actuator material:

shape memory alloy that contracts with heat or electric current to drive a CM gripper. (C) Example stiffness-changing material: hydrogel-foam composite that responds to moisture to lock/unlock CM joints on a linkage transmission system.

Smart Material Integration

Interconnected CMs also marry well with smart materials that can actuate, sense, and change their mechanical properties (e.g., stiffness). An interconnected CM could relay and transform the materials' actuation to displace distal parts (e.g., from linear contraction to rotation and vice versa [7]) or amplify, attenuate, or filter signals for sensing [21, 158]. As seen in previous chapters, stiffness-changing materials could be used to change a CM joint's DOF and DOC, which in turn changes how displacements and loads could be passed between rigid bodies. In a signal transmission and processing setting, they could provide the ability to dynamically control how signals (displacements and forces) are passed along and remixed between input and output nodes, allowing for the reprogramming of its functions (Figure 7-1).

Further Complicated Design Challenge

However, interconnected compliant mechanisms are difficult to design. While the kinematics of serially and parallelly connected CM joints can be modeled by Boolean operations over the twist and wrench spaces of their constituent joints, effectively reducing and simplifying them to a design problem identical to a single joint, interconnected CMs cannot be analyzed and simplified into independent sub-systems due to their net-like topology [278]. In particular, the adjacencies of rigid bodies and flexural connections create kinematic loop and compatibility constraints that must be satisfied on top of the transmission design goals, which leads to nonlinear dependencies between joints. Such designs also typically pertain to a large search space: its dimension scales in the power of two to the number of CM joints times six (i.e., each joint could afford a combination of the six DOF). This challenge is exacerbated by the integration of smart materials, whose functions and (re)configurations further add additional dimensions to the design space.

To this end, interconnected CM designs are often intractable to the human mind and could benefit from computer aids. Yet, while analysis algorithms for interconnected CMs exist [278], a design synthesis tool is still lacking. While established design methods such as topology optimization [145] and rigid body replacement [181] have been demonstrated in conventional, passive CM design, they each have certain limitations that render them inappropriate for the engineering problem. Topology optimization could be slow and ineffective in a 3D setting, in addition to their

shortcomings in handling materials with reconfigurable properties. Rigid body replacement methods, on the other hand, require linkage systems as input and convert them into a CM structure; therefore, they do not support ground-up synthesis from arbitrary specifications.

7.3. Overview

Problem Breakdown

An interconnected CM could be modeled as an abstract graph where the rigid bodies and flexural joints are nodes and edges, respectively. The nodes (stages) could be prescribed with boundary conditions such as fixed ends (grounded stage) or input/output nodes. A transmission is prescribed to the system by specifying an input motion at selected input and output stage(s). At the beginning of its design, the graph could be incomplete. I.e., only the I/O and ground stages are known, but their connectivity, including the need for intermediate stages, could be unknown factors in the design. Similarly, the DOF and DOC of individual joints are also unknown factors to be solved, as well as their flexural dimensions.

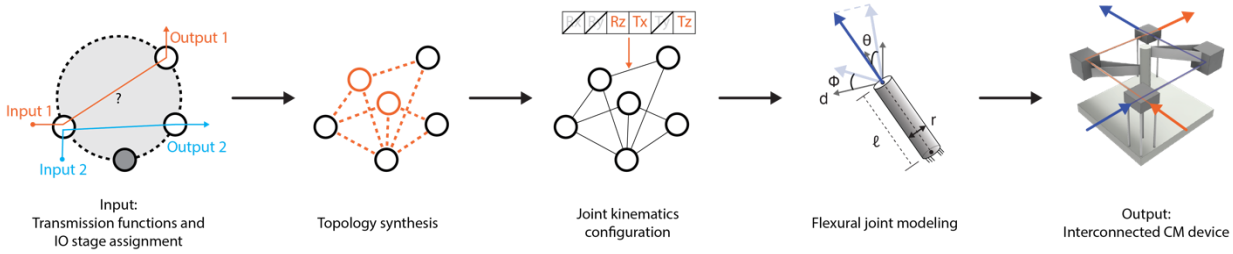

Figure 7-2. Interconnected compliant mechanism design problem decomposition.

Here, the design of interconnected CMs could be divided into three subsequent processes (Figure 7-2) - topology synthesis, joint kinematics configuration, and flexure joint modeling. The first two steps solve the problem globally to achieve the desired kinematic functions, while the third step models the CM joint-wise, independent from one another. Following this workflow, given the design specifications (i.e., I/O and ground stage assignments, desired transmissions, specific stage DOF or DOC), we seek to find an abstract graph whose joints could be configured to satisfy the desired transmission and stage DOF/DOC requirements. Note that only the connectivity and need for intermediate stages are determined at this step. Next, the joints within the graph are configured with specific combinations of DOF/DOC that satisfy the functional specifications. At this point,

the flexural arrangements (i.e., freedom and constraint spaces) at each joint are identified, and their dependencies are accounted for. Finally, given the DOF/DOC requirements from the previous steps, flexures could be added between stages to create a physically viable design. The flexures and their dimensions could also be designed to achieve, e.g., specific stiffness or range of motion requirements.

Algorithm

In this work, we use a combination of screw algebra and graph theory to develop an iterative heuristics-search algorithm. Our approach first discretizes the design domain (i.e., physical space where the interconnected CM occupies) into dense networks of rigid bodies connected by compliant joints, forming the input for iteration. The topology is represented as an abstract graph following F. Sun et al.'s analysis approach [278], where the stages are nodes, and compliant joints are edges. However, we note that there is a fundamental difference between an analysis and a synthesis setting. In the analysis setting of [278], the CM topology and the DOF/DOC of each joint are given, and the goal is to find the kinematic interrelations between the constituting rigid bodies. Conversely, in a synthesis setting, we aim to find the DOF/DOC that satisfies a desired kinematic interrelation between rigid bodies. The topology also changes over design iterations, creating a dynamic design problem where the design space dimension (i.e., the number of joints in the system) and performance landscape change with each update.

To address this dynamic design problem, our algorithm interlaces displacement estimation and heuristics-based modification to iterate the design toward user-specified transmission goals. A sparse linear solver takes the initial topology as input to estimate how rigid stages should displace against each other to achieve the specified kinematic transmission. Specifically, the displacement estimation encodes the motional freedom needed at each compliant joint to achieve the desired kinematic transmission, which could be directly translated into DOF and DOC requirements for flexure modeling using the FACT method. The displacement estimation may also expose topological redundancy and potential problems in the transmission gear train, such as kinematic decoupling between the input and output stage (i.e., when the input and output stages may displace independently as opposed to in synchrony). These issues are monitored and fixed by several design heuristics, which modify the CM topology and update the design constraints to remove topological redundancy while ensuring a valid gear train between the I/O stages. These two steps –

displacement estimation and update through heuristics – are iterated several times until arriving at a simplified topology and joint DOF/DOC assignment that produces the specified transmission functions.

Design Tool

We developed a computational design tool using the proposed algorithm to help designers navigate interconnected compliant mechanism design problems. The tool provides basic functions that allow users to model and represent design problems in a graphical modeling environment. While the algorithm to produce a satisficing interconnected CM is, at its core, an inverse design algorithm, we structured the design tool to support both forward, suggestive, and inverse design workflows. As mentioned, there could be many satisficing CM designs that satisfy a kinematic function, and an inverse design tool that operates on pre-defined logic to find a design solution may fall short of helping users navigate alternative satisficing design solutions in response to a contextualized design problem. For instance, there could be infinitely many ways to fix a decoupled transmission gear train, and fixes could be formulated with different prioritization (e.g., planar construction, small displacement and strain, and orthogonality) depending on the designer's preferences. These flexibilities are difficult to explore and navigate through a completely inverse design workflow.

Given a design problem (i.e., a CM topology and transmission specifications), the forward interaction provides a preview of displacement estimations and highlights design opportunities and needed changes according to the heuristics. The designers are then responsible for addressing these issues either on their own or with relevant information provided by the design tool. Alternatively, the design tool could also be configured to automate the design process with preprogrammed heuristics and prioritization. These heuristics and prioritization could also be modified during design iterations to "nudge" the tool to produce solutions that feature different qualitative values.

Contribution

The intellectual contribution of this work includes:

1. An algorithm for synthesizing interconnected compliant mechanism design using the freedom and constraint topology method and graph theory.

2. Design heuristics that help to search for simple and functional compliant mechanism transmission devices.

We also provide research artifacts and data to help readers better assess the applicability of our method and its impact, including:

3. A computational design tool based on the developed algorithm to support various workflows (inverse, suggestive, forward) to produce an interconnected compliant mechanism design.
4. Evaluations of the design tool's effectiveness by mechanically validating the generated design's performance against targeted transmissive functions.
5. Design examples demonstrating the enabled design space.

In the following sections, we will first discuss the technical implementation behind the algorithm, starting with the parametric representations of interconnected CMs, followed by the modeling of kinematic specifications and requirements of a viable design. Next, we discuss the sparse linear solver and optimization problem formulation to estimate compliant joint displacements for a given design problem, as well as the heuristics that the algorithm employed to simplify topologies and ensure design validity. In the latter half of this work, we document the interactivity afforded by the design tool and use several design examples to illustrate the workflow and applications of our proposed tool.

7.4. Design Representation

Topology representation

An interconnected CM could be represented as a directed graph with indexed edges and nodes, with n noting the number of nodes and e the number of edges. The graph's topology is encoded by signed adjacency ($[J]$) and incidence ($[C]$) matrices. The adjacency matrix $[J]$ is a square matrix whose dimensions are equal to the number of nodes (stages), where cell $[J]_{ij} = 1$ if an edge is directed from node j to node i, -1 if an edge is directed from node i to node j, and 0 if no edge is present. Similarly, the incidence matrix $[C]$ has a number of rows and columns equal to the number of edges and nodes, respectively. Each row in $[C]$ encodes the connectivity of an edge i: entry cell $[C]_{ij} = -1$ if the directed edge i starts from node j, 1 if edge i ends at node j, and 0 otherwise.

The graph has both feature-rich nodes and edges. The stages' volume centroid is used to identify their position within the design domain. The compliant joints are encoded by the midpoint between two adjacent stages; the midpoint is also considered the pivot for displacements afforded by the joint (i.e., rotations afforded by the joint is centered around this point). On the other hand, transmissions are prescribed as boundary conditions over the graph. For each kinematic *mode* we assign to the graph, a stage should be selected as the grounded node that has zero displacement and is the static reference point for all motion calculations. Similarly, input and output motions are assigned as twist vectors at scoped stages. For instance, $\widehat{T}_i$ denotes a displacing twist prescribed to stage (node) i. These boundary conditions are used in the latter steps for displacement estimation. $\widehat{T}_i$'s magnitude also describes how fast a stage should displace under the given transmission scheme.

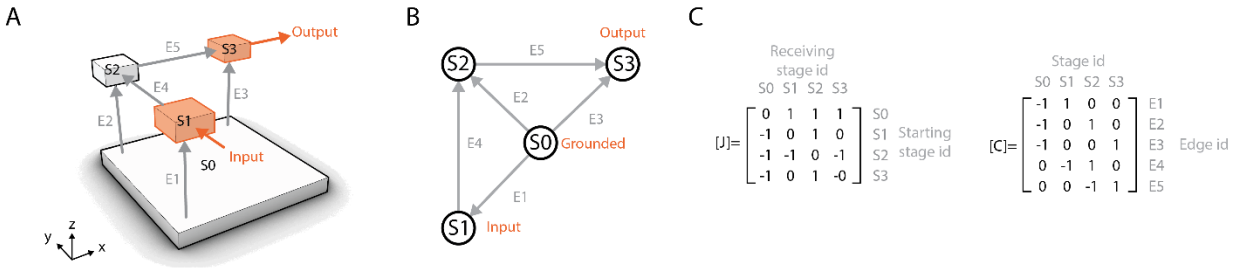

Figure 7-3. Graphical representation of interconnected compliant mechanism. (A) A mechanism consisting of four stages and five joints. Arrows show the direction of the joint. (B) The CM's abstract graphical representation. (C) The matrices representing the input topology.

Figure 7-3A provides an example of such representation. This example interconnected CM contains four rigid bodies connected by five compliant joints. With S0 as the grounded stage, a transmission is prescribed between S1 and S3 to convert a Y-axis-aligned translation at S1 into an X-axis-aligned translation at S3. The topology could be represented as an abstract graph shown in Figure 7-3B and matrices listed in Figure 7-3C. Note that the intermediate stage S2 was not prescribed with I/O motions or assigned grounded conditions. In this case, this indicates that we are not concerned with its displacements, and it could move in any (combinations of) DOF as long as the transmission relation is satisfied. If its motion was instead concerned and targeted, an output condition should then be assigned to S2. E.g., if we want to constraint S2 to have zero displacement in the transmission chain, we should instead prescribe S2 with an output displacement of $\hat{0}$.

In this representation scheme, there could be any number of input and output nodes under a transmission scheme. E.g., a transmission could map a single input motion to two or more output nodes; the input stage's motion will cause all output stages to move in synchrony with targeted velocities. Conversely, an output stage could be driven only by the synchronized movement of multiple input stages and motions. To achieve this, multiple inputs are mapped to a singular output.

A graph could also be prescribed with several transmission *modes* that are simultaneously afforded by the device. For instance, in the example of Figure 7-3A, we could prescribe an additional kinematic *mode* that transmits a x-aligned translation at S1 to a Z-axis aligned rotation at S3. The resulting device's behavior will then depend on how S1 is displaced. If S1 displaces along the Y-axis, then S3 will translate along the X-axis. If S1 is displaced along the X-axis, S3 will instead rotate about the Z-axis. The two *modes* could also be activated at the same time: if S1 displaces along the diagonal of the XY planes, then S3 will translate and rotate at the same time. This feature

allows us to design devices that have input-dependent behaviors. For instance, a device could respond differently depending on how a user is touching it (e.g., pressing or twisting).

Smart Material Integration

To incorporate smart materials into an interconnected CM design (Figure 7-4), the actuators could be represented and prescribed as the input of I/O specifications. An actuator could be placed between two rigid stages, in which one is considered the grounded stage, and the other is prescribed with a twist vector describing the actuator's motion. Similarly, sensors that sit between two stages could be modeled as the output of I/O specs, and the relative displacement between the two connected stages indicates how the sensor will be deformed during the event of interaction or actuation. It is also possible to prescribe signal amplification using this schema. E.g., to double the displacement of an actuation, we could assign an output displacement twist with twice the magnitude.

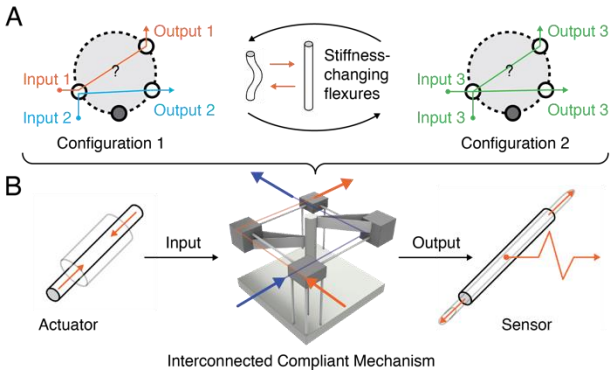

Figure 7-4. Active material integration in an interconnected compliant mechanism. (A) stiffness-changing materials could be used to stiffen/soften flexures to create different transmission modes. (B) Actuators and sensors may be considered as the input and output of transmissions to produce or detect displacements, respectively.

Here, we illustrate several example scenarios of integrating actuators/sensors:

1. Passive sensing. The actuation is carried out by an external stimulus (e.g., user manipulation, winds, etc.), and the design is concerned with picking up the interaction event using a sensing material. While prescribing the I/O function, we could use the input twist to describe how the input stage will move according to the interaction, thereby eliminating and filtering out undesired actuation "noises". The sensor placement could then be assigned by pairing an output motion that distorts a strain-responsive material with a

signal amplification factor, which in turn changes the material's, e.g., impedance, when passing a current through it.

2. Open-loop actuation. In this scenario, the I/O input describes how an actuatable material will rotate or translate an input body to produce a desired displacement at another body within the design domain.
3. Closed-loop actuation. This scenario connects an input actuation to multiple outputs, one of which describes a sensing material placement. This way, in the event of actuation, the remote output bodies will carry out their prescribed actuation, and the joint containing the sensor will displace and strain it to produce a signal. The signal can then be passed to a processing unit to modulate the actuation.

In addition to sensing and actuation, smart materials could also be used to reconfigure a device between different *configurations* each providing different sets of transmission *modes*. E.g., a CM robot could be designed to have two *configurations* that locomote in different environments (e.g., dry soil and water). Each *configuration* would then have one or more transmission gait *modes* that map an input actuation to targeted limb movements. To enable this, we could use stiffness-changing materials to dynamically adjust individual compliant joints' DOF, which in turn changes how actuations are transmitted or converted within the CM device, resulting in functional reconfiguration.

7.5. Iterative Design Algorithm

The iterative design algorithm takes a reductive approach to design an interconnected compliant mechanism. Given an initial topology and a set of transmission goals as input, the algorithm alternates between estimating displacement velocities and heuristically modifying the topology to produce a valid and simplified design. The heuristics pertain to reducing the topology's complexity and resolving any decoupling issues (i.e., when the input and output nodes can move independently as opposed to synchronously). To simplify a topology, a heuristic removes elements (stages or joints) from the graph, reducing the graph's complexity upon each application. Alternatively, to resolve decoupling issues, a heuristic prescribes additional displacements to a stage between the input and output stages.

Algorithm 7-1. Iterative Design Algorithm for Interconnected Compliant Mechanisms

```
def iterative_brute_force_algorithm(domain, goals, max_iteration):

	heuristics = topology_simplification_heuristics + decoupling_fix_heuristics
	graph = discretize(domain, goals)

	iteration_count = 0
	done = False
	do while not complete:
		iteration_count += 1
		is_modified = False
		velocity = velocity_estimation(graph, goals)
		for heuristic in heuristics:
			is_applicable = heuristic.check(graph, velocity)
			if(is_applicable):
				graph = heuristic.modify(graph, velocity)
				is_modified = True
				break
		if((not is_modified) or (iteration_count == max_iteration)):
			done = True
	end do

	return graph, velocity
```

It is worth noting that due to the changes to topology structure and displacement constraints, each heuristic modification will cause the previous velocity estimation to become obsolete thus requiring recomputation, thereby triggering the next iteration. A design is deemed complete if no heuristics were applied during an iteration, which indicates the topology could not be further

simplified and all decoupling issues have been fixed. The algorithm is also bounded by a max iteration limit that prevents the algorithm from oscillating between two actions (i.e., when a subsequent heuristic reverts the changes made by the previous action, causing the previous heuristic to refire). The algorithm then returns the simplified topology along with the velocity requirements at each compliant joint, which informs further flexural modeling steps to physicalize the design into a fabricable model. In this work, the velocity requirements are processed and visualized using the joint modeling function in ReCompFig [335] with minor modifications to aid users in modeling the flexural joints. The following sections provide implementation details for each part of the algorithm.

Initialization and Topology Manipulation

An initial graph is created by discretizing the space into densely connected stages and joints. To do this, a bounding rectangular box is computed for each design domain. Next, the box is discretized into a 3D array of nodes connected by edges of regular length along each principal axis. Finally, the array is trimmed by the input domain to remove nodes that fall out of the specified space. Users may also modify the topology (i.e., removal or addition of nodes and edges, modifying node positions) at any time during the algorithm to manually nudge the topology toward their visions.

It is worth noting that during initialization, the domain could be discretized with arbitrary resolutions (i.e., number of stages per direction). Higher resolution would afford more complex transmission functions due to the additional joints providing more design freedom, but the computational time also increases cubically. Conversely, lower resolutions would be less expressive than their counterparts but would be much faster to compute. Nonetheless, we surveyed literature (Library of Compliant Mechanisms in [105]) and report that a typical compliant mechanism design rarely requires more than five stages along any given dimension (Figure 7-5). Designs are also one- or two-dimensional in most cases. Thus, we recommend users start with a discretization resolution of five elements per direction if they are uncertain about the needed topological complexity, and redundant elements will be removed during the early stages of the design iteration.

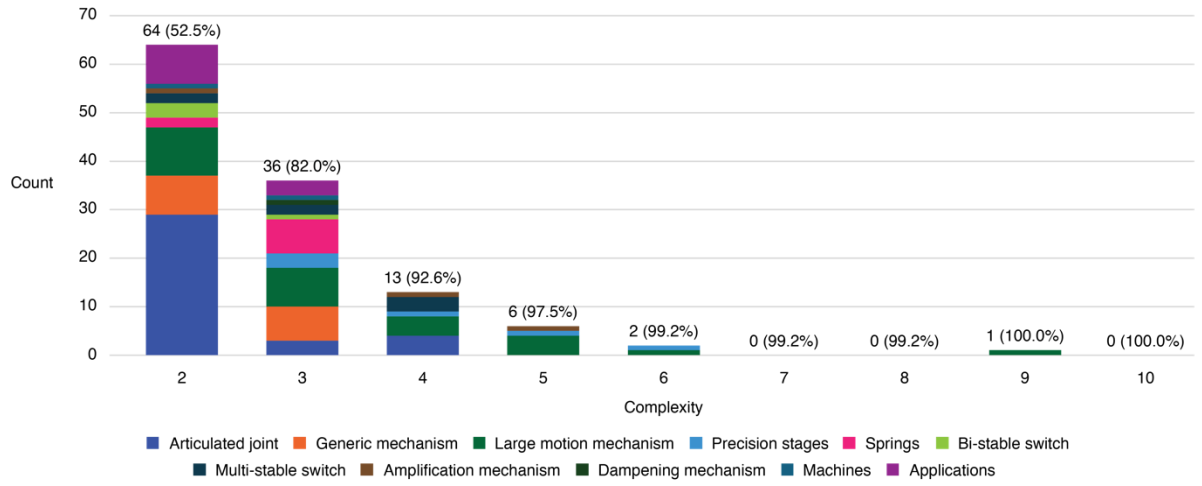

Figure 7-5.Typical compliant mechanism design's topological complexity measured by the number of joints between the input and output stages. The design examples are cataloged in [105]**. The number above each column shows the number of designs that appeared in** [105] **for a given complexity. The percentage shown in parenthesis shows the accumulated fraction of designs (the number of designs that appeared with equal or lower complexity divided by the total number of designs).**

Velocity Estimation

Given a topology resulting from either the initial discretization or the previous design iteration, the joint displacement velocities required to satisfy transmission objectives could be computed by applying the Joint Velocity Estimation Algorithm discussed in Section 7.10. The velocities are computed for each mode using the same topology but with varying constraining linear systems. In addition to the joint displacements, the displacements of each stage could also be calculated by constructing a path vector that points from the fixed and accumulates the velocities along the path factored by their respective directions.

Design Heuristics

This work uses two types of design modifications: Topology Simplification and Decoupling Fix. Topology simplification could be further broken down into three individual rules: merging synchronized stages, collapsing linear arcs, and removing redundant joints. The heuristics consist of two primary routines: a *check* that evaluates the topology and velocity to identify if a heuristic is applicable to the current design and a consequent *modification* that responds to the issue.

7.6. Design tool

The iterative design algorithm is implemented as a plug-in for the modeling software Rhinoceros 3D. The tool is implemented primarily in Python, and the interface is constructed using Grasshopper and UI+ by David Manns [54]The tool supports users in modeling design problems and working with the algorithm to create a design from end (conceptualization) to end (fabricable model) within the same environment, eliminating the need to switch between software packages.

The tool's usage follows the workflow established in Algorithm 7-1. The user approaches the tool with a design domain in mind, and the tool helps the user to model an initial graph. The transmission goals are then specified over the graph to establish the design problem, which is subsequently sent to the design solver and iterated with the design heuristics to until arriving at a valid and simple CM topology and joint DOF requirements. The tool then helps the user to model the flexural joints using a schema identical to that of ReCompFig [335] to produce a fabricable model. These steps are organized into different functional tabs in the design tool (Figure 7-6).

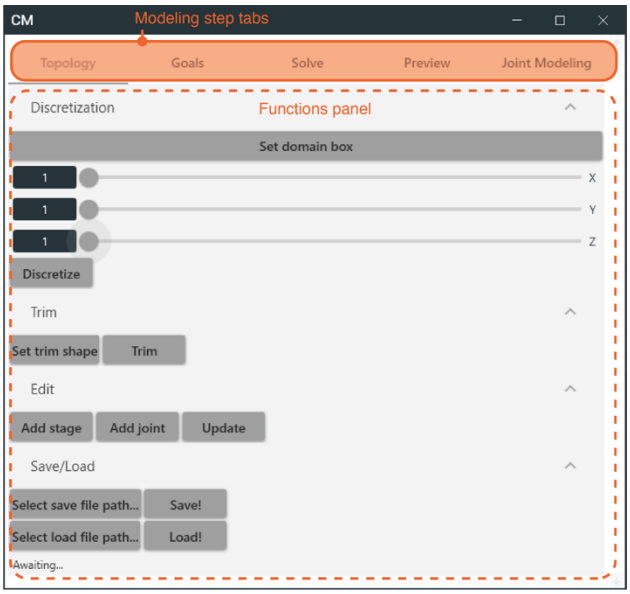

Figure 7-6. Design tool interface.

It is worth noting that the design tool also affords different workflows and modalities of interaction with the user. The tool could be set to find a design solution autonomously, creating an inverse design workflow that requires minimal user input. Conversely, the tool could also be used in a forward modeling manner, where the tool evaluates the design to find potential problems and

prompt correction. This mode of interaction gives the user complete control over the design while informing decisions to iterate toward the goal. Alternatively, the tool could also provide an interactive workflow that sits between forward and inverse – a suggestive design workflow where the tool assesses the current design and suggests potential modifications for the user to choose from. This way, the tool and the user could co-steer the course of design and arrive at a more satisficing solution. A user could also seamlessly switch between these design modes at any point while using the tool.

In the following part, we provide a walkthrough of the tool using an exemplary design task – an amphibian CM robot (Figure 7-7). The robot is projected to be 100 mm long and consists of two *configurations* to create gaits for on-land and in-water locomotion. We assume the movements are driven by two shape-memory alloy springs, and the *configurations* are adjusted by stiffness-changing flexures. In each modeling step, we also discuss different interaction modalities provided by the tool in context.

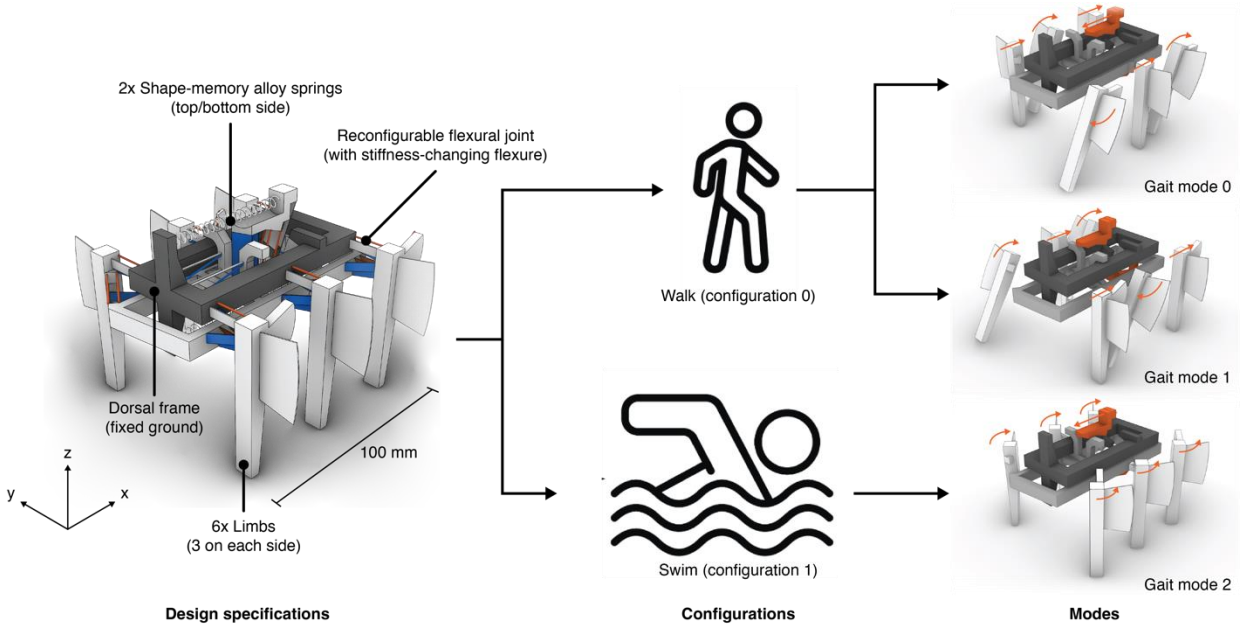

Figure 7-7. Amphibian robot design specifications from concept to configurations and modes. The images show the robot designed with the tool. Arrows on the right-column images show the input and output motion DOF.

Step 1. Topology Initialization

The design workflow starts with modeling an initial-guess topology. The user could either model the topology manually by adding rigid stages and compliant edges or specify a domain to the tool for discretization (Figure 7-8). When manually modeling the topology, the user should add points to the model that represent the rigid body center points, and edges are added by selecting the incident stage points (Figure 7-8B). On the other hand, if the user begins the workflow with a design domain represented by a volumetric shape, the design tool will allow the user to select a discretization resolution for each axis (Figure 7-8C). Once discretized, the user could then drag and move stages around to further modify the topology's form, and the edges are updated accordingly. Users could also manipulate the topology at any point of using the design tool – even while using the iterative solver. The solver would respond to these topology manipulations and update its velocity estimation and heuristics evaluations. Finally, the design tool also provides functions to export and import design models as a .json file.

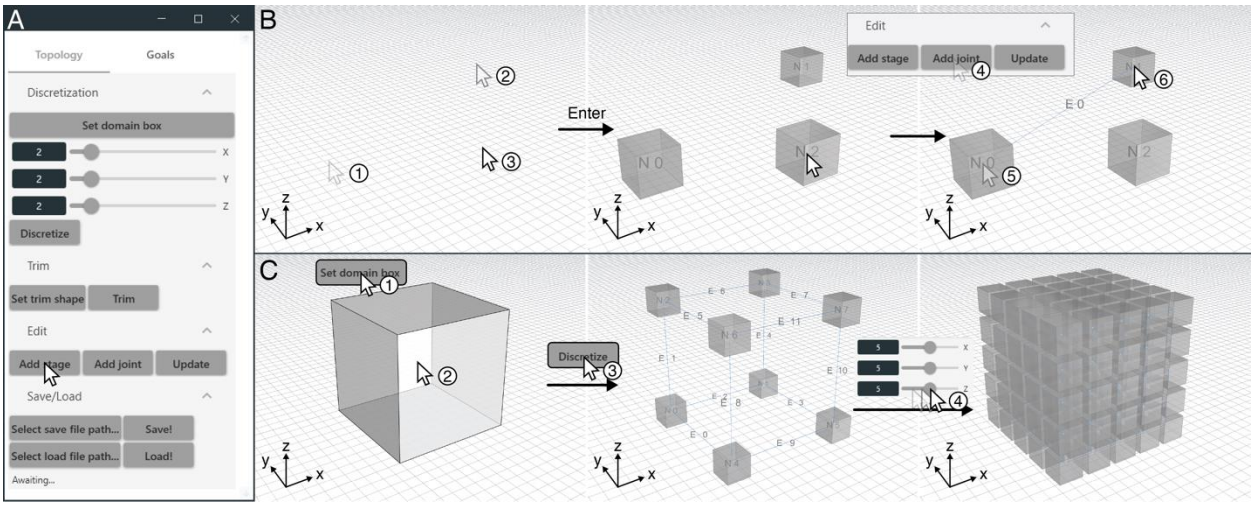

Figure 7-8. Topology initialization. (A) A screenshot of the design tool panel. (B) Manually modeling the topology. (C) Initializing the topology by domain discretization.

When designing the amphibian robot, the topology is initialized by discretization. Without an explicit robot model or outline to begin with, we used a bounding volume to specify the design domain. The domain is then discretized into a $5 \times 5 \times 5$ array of stages connected by orthogonal joints.

Step 2. Goals Modeling

Following topology initialization, the next step pertains to prescribing design goals over elements in the graph. The user should first specify the number of kinematic *modes* needed for the design task. The design tool then populates the interface with modeling panels for each *mode* (Figure 7-9A). In the panel, the user can specify a fixed stage for the mode and the *configuration* to which the mode belongs. To prescribe an input or output motion, the user should click on the "Add motion" button. A pop-up menu (Figure 7-9B) would then prompt the user to specify the scoped nodes, kind (i.e., input or output), and displacement. Once satisfied, the user then clicks the "Submit" button for the tool to register these prescriptions (Figure 7-9C). This step can be repeated multiple times to prescribe multiple I/Os transmissions to a *mode*. The design tool also visualizes the prescriptions for review, but only the prescribed stage's motions are captured. The unscoped nodes' motions are resolved in the latter steps.

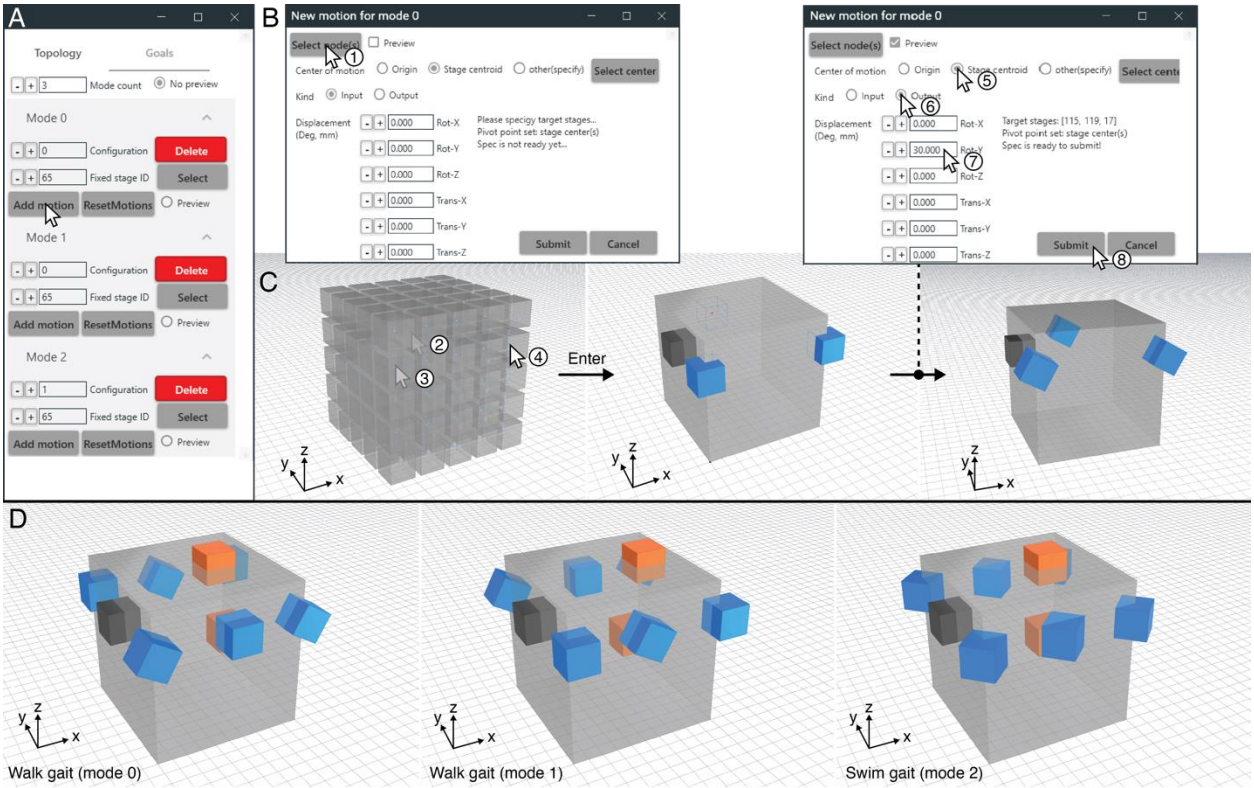

Figure 7-9. Goals modeling. (A) Goals modeling panel. Clicking on the "Add motion" button will bring up a (B) pop-up window that prompts the user to add transmission I/O. (C) Goals modeling steps. The images show the steps of adding leg rotations for walk gait mode 1. (D) Resulting gait mode assignments. Input and output stages are colored orange and blue, respectively.

We created three *modes* to specify the amphibian robot's behaviors (i.e., gaits) under two *configurations* (Figure 7-9D). A fixed stage was shared among all *modes*; two stages are selected as the transmission input, whose displacements will be driven by SMA springs. The robot's six legs are designed as the transmission output displacing along different DOF and speeds. In the walking configuration, the robot would lift half of its legs and propel its body forward with the rest. This gait alternates between two *modes*, each driven by one SMA spring. On the other hand, in the swimming gait, both SMA springs work in concert to turn the fins (built onto the legs) and propel the robot forward.

Step 3. Design Iteration

Once a design problem is established, the design tool then helps the user apply the iterative algorithm to produce a valid design with simplified interconnected CM topology (Figure 7-10A). The user could interact with the solver in two ways: automation and manual/suggestive mode. Users may also switch between these modes of design at any step. In the solver panel, several indicators are used to communicate the heuristics evaluations (Figure 7-10B). A green light indicates that a modification heuristic is not applicable to the current design, whereas a yellow light suggests an opportunity to apply a heuristic. A red light indicates an issue that needs to be fixed to produce a valid design. In particular, the yellow light pertains to topology simplification and the red light is associated with decoupling issues. The solver also allows users to redo/undo changes to explore alternative ways to modify a design.

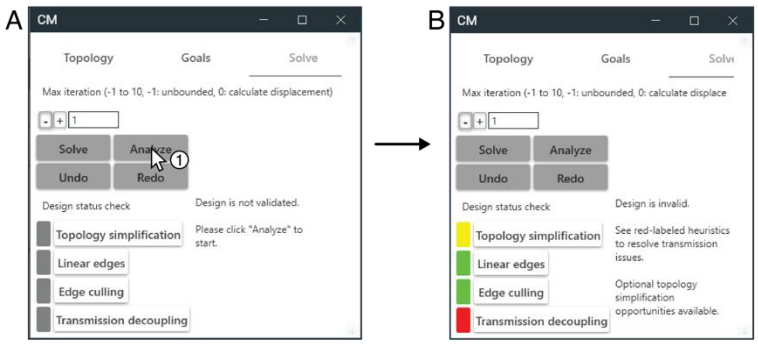

Figure 7-10. Design iteration panel. (A) In this example, the user clicks the "Analyze" button to initiate the suggestive mode. (B) The design tool then reports the design status as examined by each heuristic.

Automation Mode

In the automation mode, the user leaves the tool to finish a design task without manual input. On the top of the panel, the user could specify the tool to run for certain iterations or indefinitely until the design is completed. The solver may also terminate early if the design cannot be further simplified and presents no decoupling issues. The exit status and design completion status are then textually communicated to the user. In the finalized design, the user can expect to see a design where all heuristics indicator turns green, signaling no further changes are needed.

Manual or Suggestive Mode

The design tool also allows the user to modify the design manually or collaboratively with help from the heuristic engines. To achieve this, the user should click the "Analyze" button to generate a velocity estimation. The design tool then *checks* the topology and velocity estimation against each heuristic to generate their status report and modification suggestions. The status report updates the indicator signals, and the user can click on a heuristic to formulate a *modification* in a pop-up menu (Figure 7-11A). The formulation processes are introduced below.

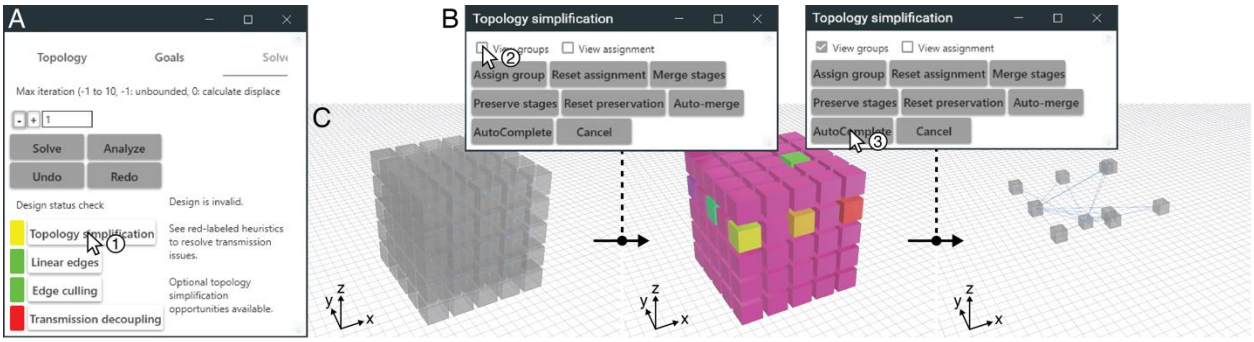

Figure 7-11. Topology simplification workflow. (A) The user clicks on the "Topology simplification" heuristic to bring up (B) a pop-up menu that helps to formulate a modification. (C) The user previews the solver-generated group assignment and accepts the change.

Topology Simplification

If a design presents an opportunity for merging synchronized stages, the pop-up panel will render the stages in color-coded groups to communicate a potential merge plan (Figure 7-11B). If the user decides to merge stages manually, they could use the "Assign group" button to specify a merge. The tool also double-checks the assignment to make sure the scoped stages are indeed synchronized, or it may prompt the user to correct their decision. Alternatively, the user could intervene by flagging stages for exclusion from merging. The tool would then proceed accordingly.

Lastly, should the user be satisfied with the suggested merge plan, they could also click the "AutoComplete" button to apply the merge (Figure 7-11C). These different levels of control allow the user to manipulate the course of design with varying levels of mixed-initiative collaboration. Similarly, the linear arc and saturated joints heuristics also allow the user to manually collapse arcs and remove edges, or they could allow the tool to automate for an iteration (i.e., apply the suggested *modification*).

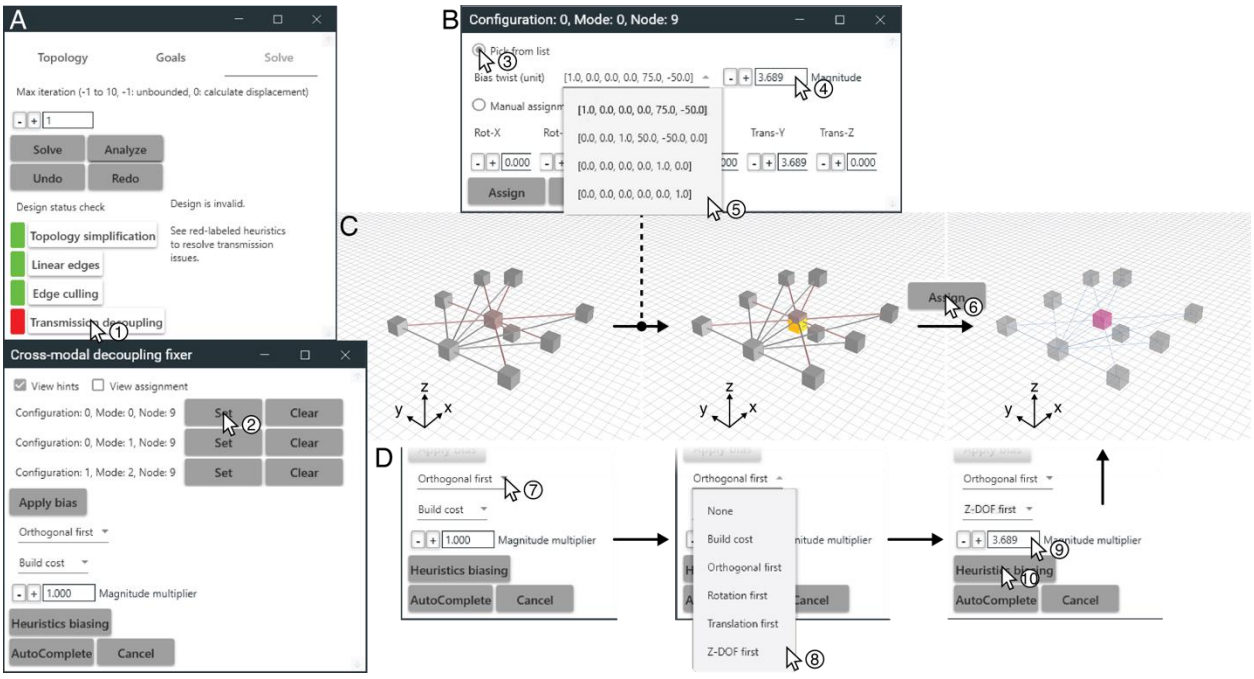

Figure 7-12. Decoupling fix workflow. (A) The user clicks on the "Transmission decoupling" button to bring up the heuristic's pop-up menu. Clicking on the "Set" (bias) button further brings up the (B) menu to specify a biasing twist. (C) The workflow of specifying a twist by selecting from the heuristic's suggestions. (D) Alternatively, the biasing steps could also be done by changing the heuristic's decision priorities and letting the solver pick twists accordingly.

Decoupling Fix

If the design tool identifies a decoupling issue, the design tool will highlight the decoupled edges and nodes for the user to examine (Figure 7-12A). There are a few ways the user could formulate a fix. By default, the user may allow the design tool to "Auto Complete" the design using its built-in heuristics. The pop-up menu also allows users to alter the heuristics engine's priorities when formulating *modifications* (Figure 7-12D). A drop-down menu allows users to prioritize different biasing twist selection logics and change the bias's magnitude using a scalar multiplier (see Decoupling Fix: Intra-modal), giving users more control over fixing transmission decoupling. Conversely, if the user decided to manually bias the gear trains, they could click on the associated

buttons in the interface to bring up a menu (Figure 7-12B) to directly specify the biasing twist. Here, the tool provides two ways to proceed. The user could either pick from a list of biasing twist suggestions (i.e., taking over from Step 2: Select unused DOF of Decoupling Fix) or manually key in the biasing twist themselves, superseding the heuristic engine's decision altogether (Figure 7-12C).

When iterating the amphibian robot design (Figure 7-13A), we used both the automated and suggestive modes to create a valid design in three iterations. The tool suggested merging synchronized stages in the first round, and the user accepted the *modification* (Figure 7-13A. i.e., automation). However, merging blocks resulted in an asymmetric topology (Figure 7-13B), and the user manually dragged the stages to re-obtain symmetry (Figure 7-13C). The design tool re-analyzed the updated topology and identified a lingering decoupling issue. The user then examined the biasing options one by one and picked a biasing twist for each mode-node combination from the design tool's suggestions (Figure 7-13D).

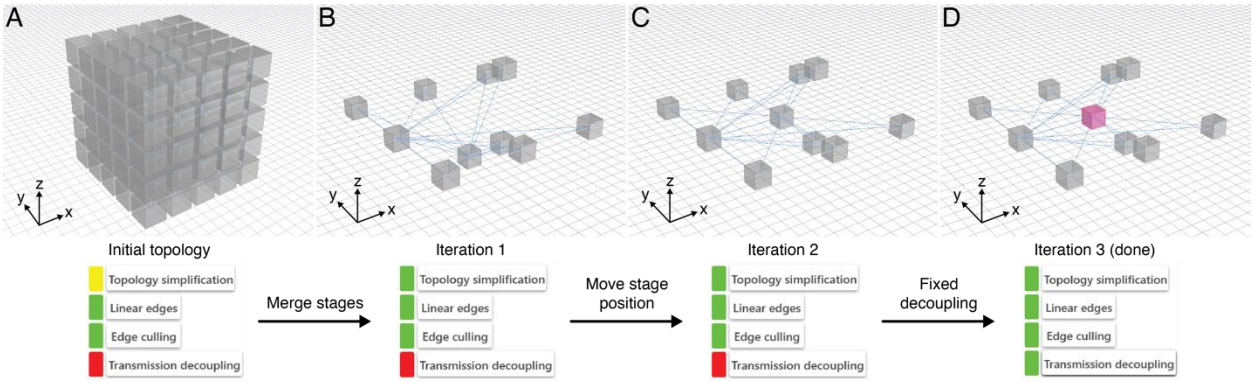

Figure 7-13. Amphibian robot design iteration history - (A) initial input, intermediate results (B) upon topology simplification and (C) stage position manipulation, and (D) final output after fixing decoupling issues.

Step 4. Transmission Preview

A separate tab allows the user to visualize the stages' displacements as estimated by the solver (Figure 7-14A). The estimations are not limited to the finalized design; users could also use this function to preview the unfinished design's stage displacements to inform their decisions. The preview tab provides one panel per *configuration*. Under each configuration, a slider allows the user to preview the *modal* motions from 0% to 100% displacements. The transmission *modes* under a *configuration* could also be actuated and visualized at the same to preview how they would behave in coherence (Figure 7-14B).

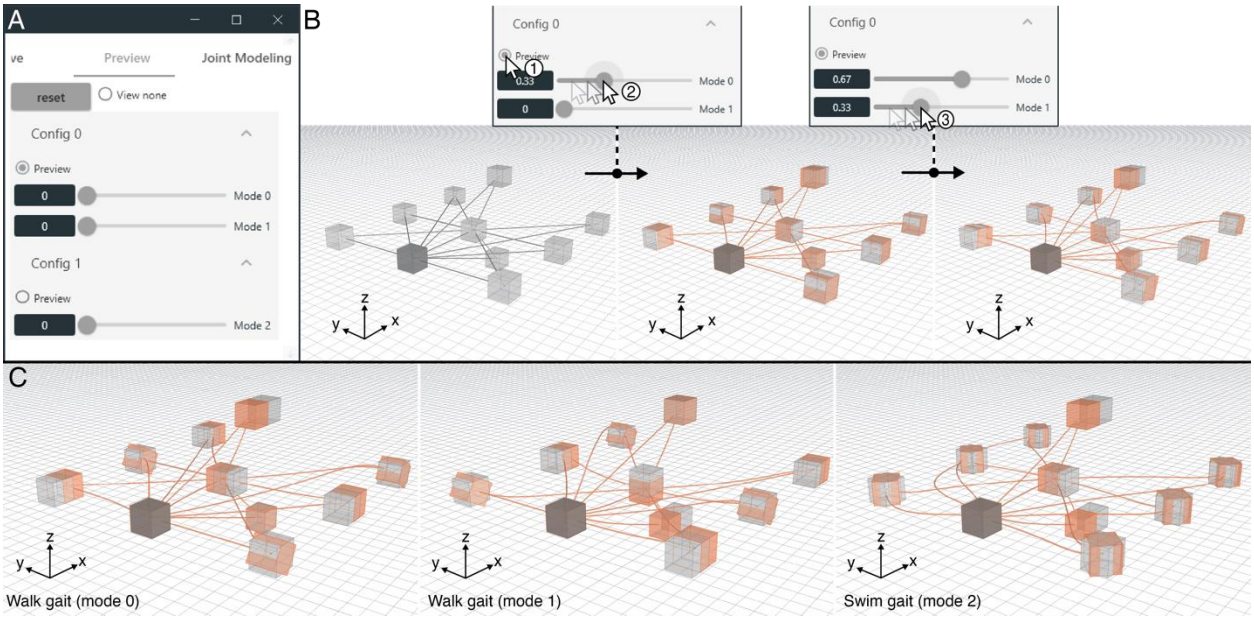

Figure 7-14. Transmission preview. The rigid stages are visualized using a proxy box. However, they could be modeled into any shape in latter steps. (A) The panel for reviewing the displacements of each kinematic mode and configuration. (B) The user previewing the combined actuation of the two gait modes under the walking configuration. (C) The transformation previews of each kinematic mode.

Step 5. Flexural Joint Modeling

Once a design is complete, the tool then helps the user to model the flexural joints. The tool parses the velocity estimations to generate modeling instructions per joint (Figure 7-1). These instructions are organized into panels to help users tackle one joint's modeling at a time (Figure 7-1A). Under each panel, the tool helps the user to model both passive and reconfigurable flexures by providing visual and textual prompts to inform their placement (Figure 7-1B-C). Users may choose to use either flexural rods or blades (i.e., thin sheet flexure) Once all joints are modeled, the tool also suggests the flexure states (enabled or disabled) required to instate each configuration (Figure 7-1D) as well as flexure dimensions. Note that designers may also replace the rigid stages with custom geometries to better iterate and visualize the design (Figure 7-1B). The stages may be modeled into any shape as long as they are sufficiently rigid (i.e., at least one magnitude thicker than the flexures at any cross-section). Yet, the tool does not perform a rigidity check for the user.

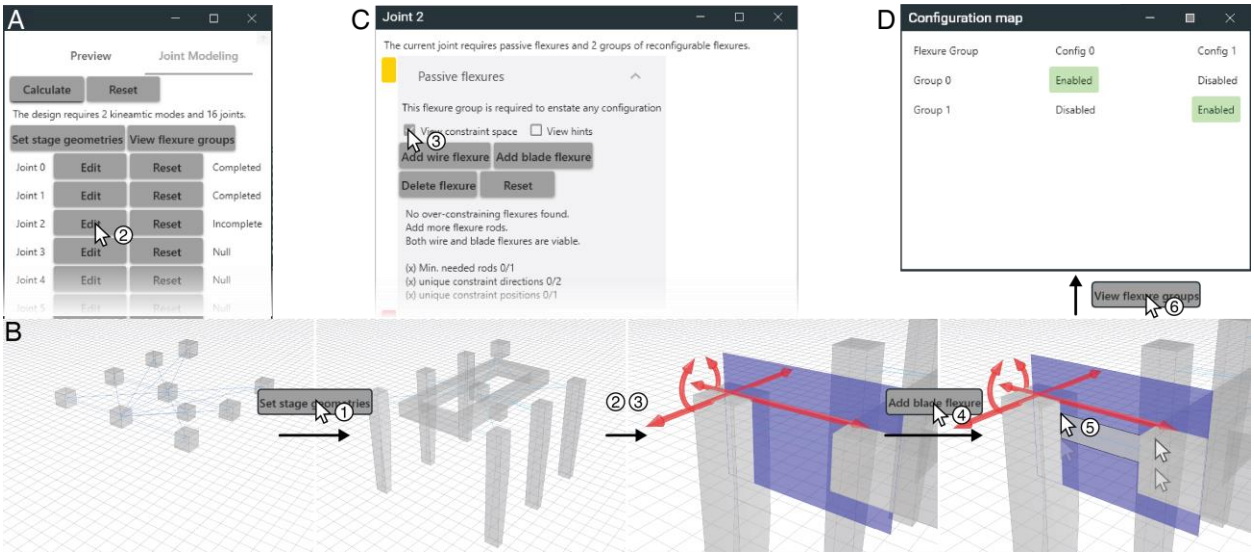

Figure 7-15. Flexure Modeling. (A) The panel summarizes the joints and configurations. (B) In this example workflow, the designer first replaces the rigid stage proxy boxes with coarse models of the robot's limbs and torso, followed by editing the flexural joints. Clicking on the "Edit" button brings up a (C) pop-up window that communicates the number of flexure groups needed for each joint and information about the flexure layout. The designer could then model the flexures (i.e., blades in this case) according to the information. Once completed, the user could click on "View flexure groups" to (D) review the flexure adjustments required for each configuration.

The amphibian robot contains twelve compliant joints in total, but several of these joints are identical due to the repeating structure of a multi-legged robot. In total, the design requires three flexure groups - one passive and two reconfigurable. The passive flexures are shared by both the walking and swimming *configurations*, and the reconfigurable flexures are softened and hardened in alternation to establish each gait's transmission requirement.

7.7. Design Examples

Amphibian Robot

Robotics design has garnered interest in HCI for enabling tangible interactions [87, 129, 187, 284] and creating physical agents that aid users in shaping the situated environment [151–153, 283]. However, existing robotics design tools are often based on linkage or conventional mechanical articulations (e.g., hinges, linear rails), making them less suitable for wet or dusty environments in addition to increased weight. Robotic structures that rely on passive materials are also unimodal; their kinematic functions can only be adjusted by rearranging parts and joints. By contrast, compliant mechanisms are advantageous in these aspects. With the inclusion of property-changing materials, a CM robot may also alter its kinematic functions without manual reassembly.

Here, we use the tool to design a robot that can adjust its gait *modes* to locomote in different environments. The robot consists of six legs driven by two actuators (Figure 7-16A). The gait logics are inscribed in the transmission system, and the robot may switch between swimming and walking *configurations* by softening and hardening different groups of flexures (Figure 7-16B, C). In the walking gait, the two actuators fire in alternation to lift and propel two sets of legs. Alternatively, the actuators contract simultaneously – in expectance of larger resistance from an aqueous environment - to drive legs to rotate and propel water with its fins.

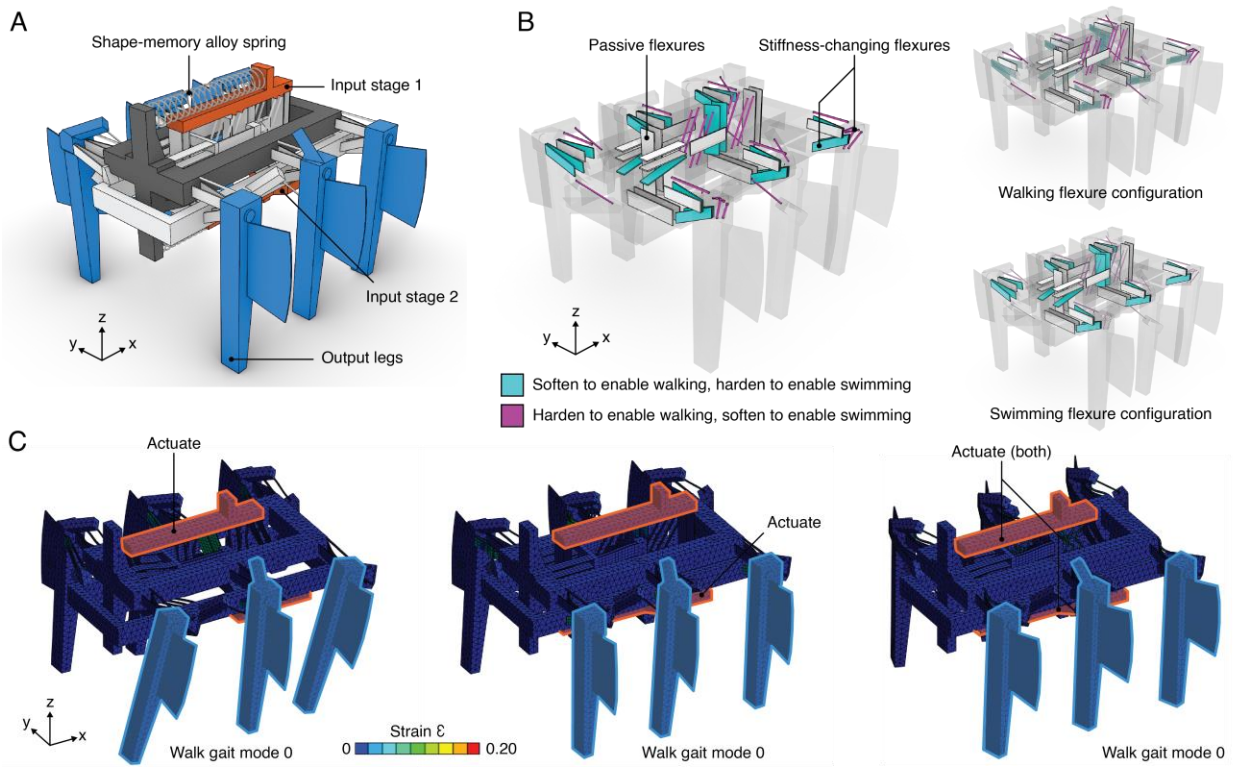

Figure 7-16. Amphibian robot design. (A) The robot's parts and their functions. (B) As designated by the design tool, the flexure groups consist of two reconfigurable sets and one passive set. These flexures may be selectively softened/hardened to enable different gaits. (C) Finite element simulation of the robot under different gaits. The reconfigurable flexures are assumed to have an elastic modulus similar to Epoxy (2.53 GPa when hardened, 20 MPa when softened). We note that the strain is mostly concentrated at softened flexures as they underwent compression and buckling. Only the legs on the near side are highlighted to avoid visual cluttering.

Robotic structure design has traditionally been challenging due to the physics and kinematics involved. Designing a valid transmission between an actuator and an end effector could be challenging for inexperienced engineers, and the difficulty is further worsened by designing for kinematic reconfiguration. In this example, we report that the presented design tool provides a convenient way to create robotic devices. Users are only responsible for specifying the input and output motions, and the tool plans the transmission gear train with optional user input.

Qualitatively speaking, the design tool also helps to convert primitive actuation (e.g., axial contraction) into multiple parts' complex and synchronized movements, enabling designers to harness active materials for more expressive and functional robotic motions.

Smart Door Lock

In this example, we use the design tool to create a door lock wholly made of a compliant mechanism and only unlockable by manipulating the handle in a correct sequence of actions. The door lock consists of an input handle, a lock pin, and a latch (Figure 7-17A). When closed, the latch and lock pin extend into the lock block, mechanically preventing the door from swinging open. The lock pin also hooks onto the lock block with matching geometries to secure the door lock in place. To unlock the door, the input handle should be pushed upward to disengage the lock pin from the hook, followed by (Figure 7-17B) rotating the handle downward to move the lock pin into a clear passageway. The door is then opened by pushing the handle forward to retract the latch and lock pin. A stretchable sensor (e.g., Adafruit Conductive Rubber Cord Stretch Sensor) may be attached between the fixed frame and the latch rigid body. The sensor's resistance changes when the latch is retracted, which may be detected by a microcontroller to trigger further events.

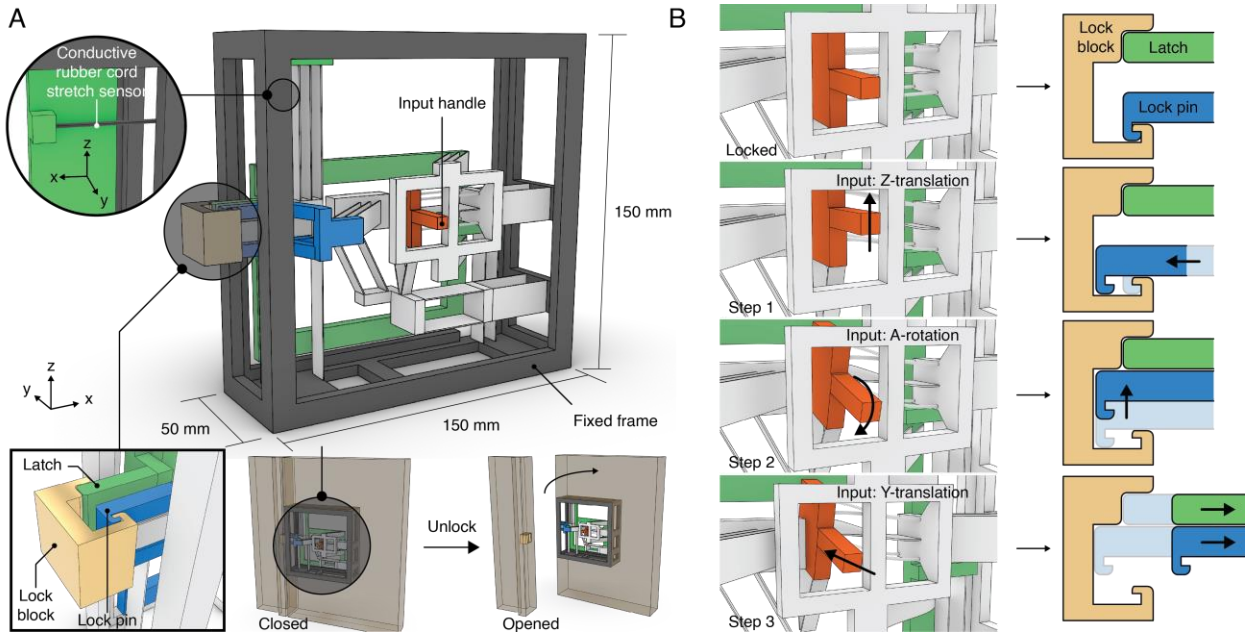

Figure 7-17. Smart door lock design. (A) The door lock is embedded in the door with latch and lock pins extending into a lock block sitting in the door frame. (B) The lock may only be disengaged by moving the handle in three consecutive steps.

The device is designed with an initial model to establish the design domain and bodies of interest (Figure 7-18). We then used the design tool to create an initial topology. Three transmission *modes* were specified to create the design problem, each creating a pin and latch movement, as shown in Figure 7-17B. The design is iterated with automated topology simplification and decoupling issues are fixed by picking from the design tool's suggestions. Next, the flexures are modeled according to the design tool's instructions, and the rigid bodies' shapes are modeled to connect between flexures and provide the intended function. The simulation results showed that the door lock affords the intended functions, and the lock pin and latch movements agreed with the schematics (Figure 7-19).

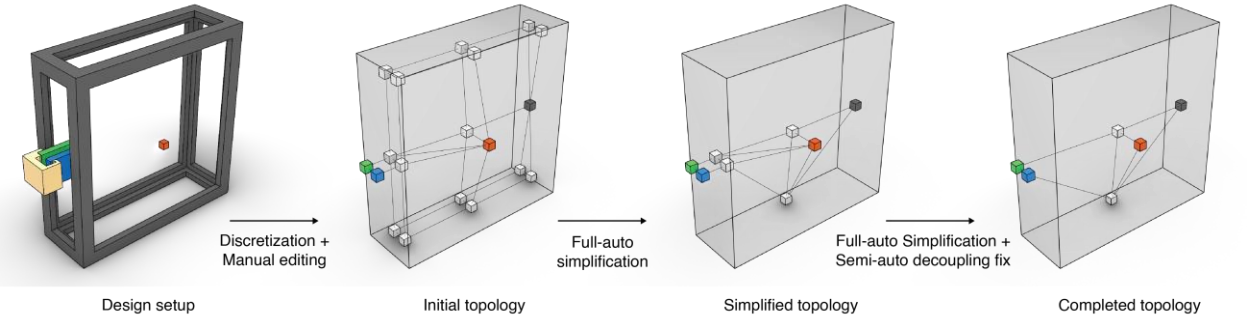

Figure 7-18. Smart door lock design process. The design tool was used to discretize a space inside of the fixed frame to produce a 3 × 2 × 3 rigid body array. The latch and lock pins were manually added to the topology, and the input handles' position and connectivity were also adjusted in this step. The design tool suggested topology simplification at the first two iterations, which the designer accepted, and the tool was allowed to automate the decisions. At the third iteration, the design tool identified a transmission decoupling issue, and the designer resolved the issue by picking from the tool's suggested actions, producing the final design.

Qualitatively speaking, we report that compared to metamaterial mechanisms [109–111], our design tool can help users design devices with more complex behavior. Specifically, the door latch in [109] is limited to planar motions (2D) and a singular transmission mode. In contrast, this example showed that our tool may prescribe motions in 3D (i.e., the X-rotation and Y-translation are out-of-plane displacements) and allow integration of multiple transmissions. This example also shows that the design tool and algorithm may create structures with computational behavior where the output bodies' motions are conditioned on the input's displacement. Incorporating additional bodies that interact with these motions (i.e., the lock block) may further complexify the behaviors and an "encrypted" tangible interaction with the user. In this case, the door may only be unlocked by users who know about the unlocking sequence (Figure 7-20). We speculate that such features may enable tangible security and mechanical computation in future work.

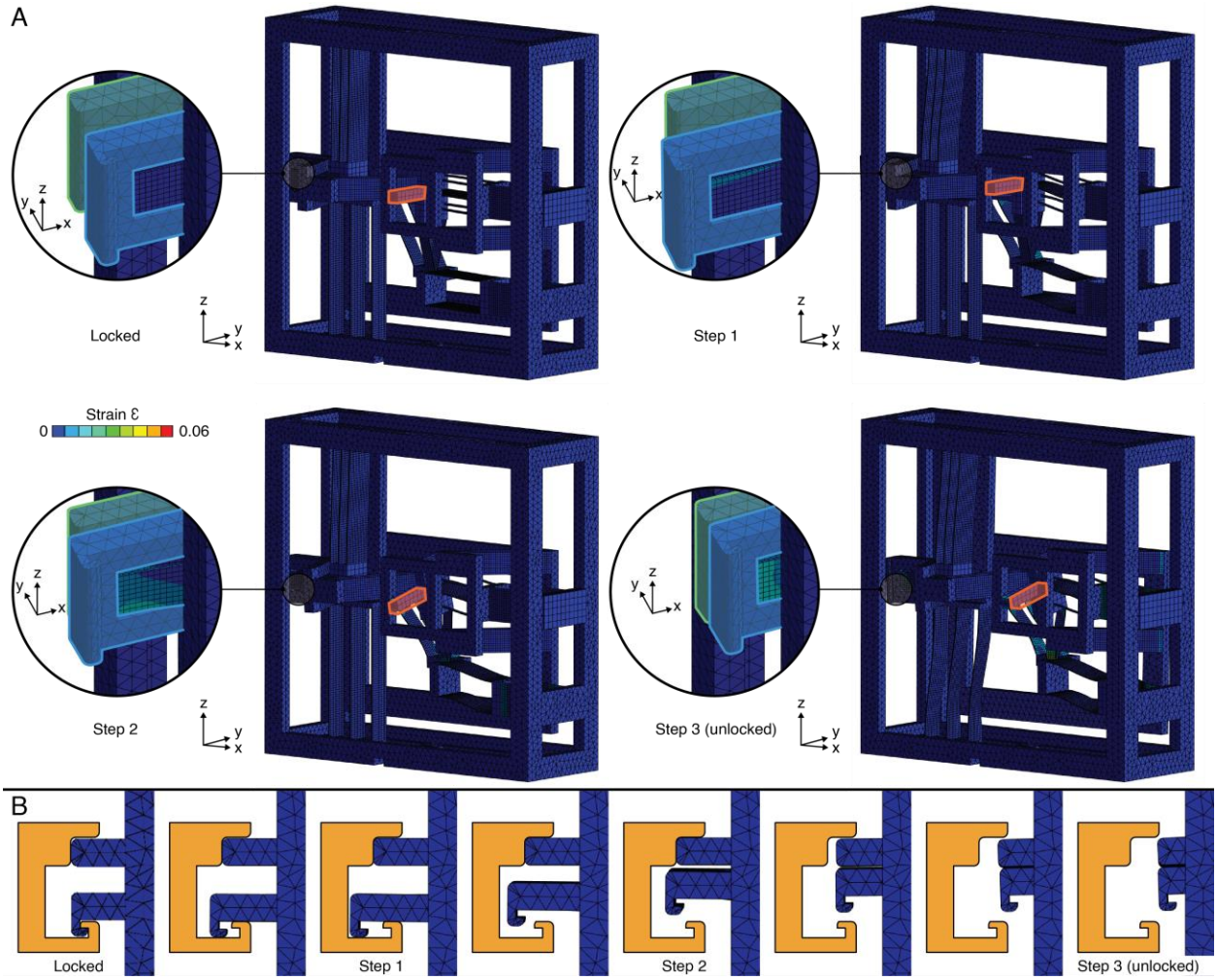

Figure 7-19. Smart door lock finite element simulation result – (A) the device's overall displacements and (B) the latch and lock pin's movement within the lock block. The steps correspond to Figure 7-17B.

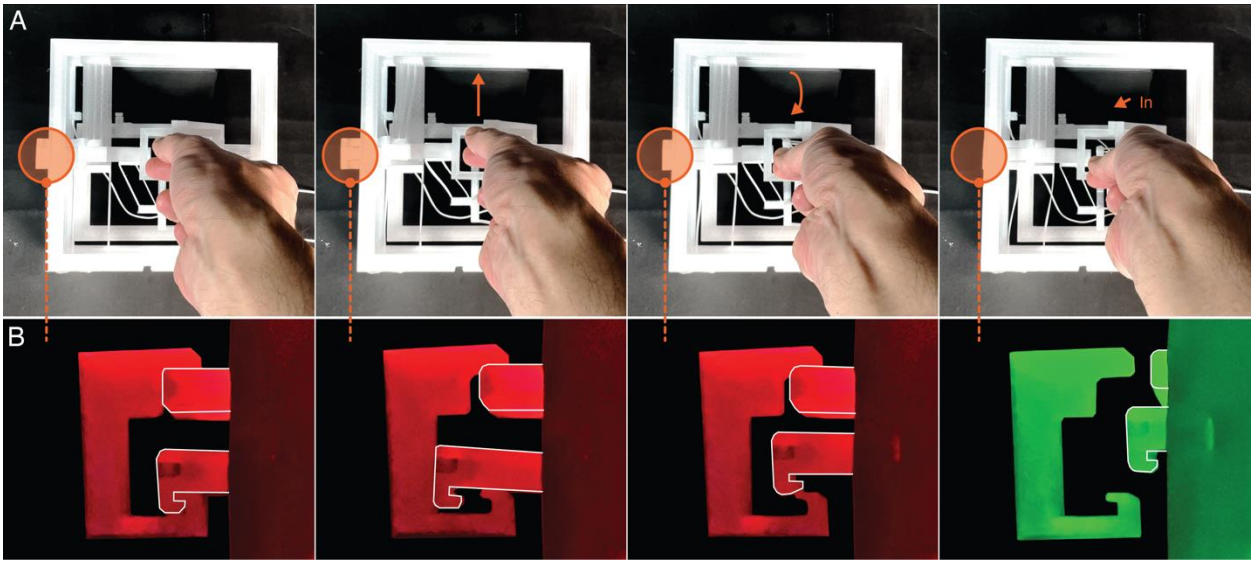

Figure 7-20. The smart door lock unlocking sequence phtographed from a physical prototype: (A) The lock mechanism was viewed at the front, and (B) the latch and lock pin were viewed from the top. In (B), the light illuminating the lock changes when the sensor rubber coord is stretched, causing it to turn from red to green when unlocked.

Modular Physical Interface

Here, we show that the design tool and algorithm may be used to hierarchically produce devices with complex functions. We used the design tool to create four physical interface building blocks, each providing different functions – knob, trigger, and button (Figure 7-21A). The input element is located at the top side of the blocks, and the internal structures transmit the input motion to drive the side plates to translate outward. These building blocks are designed with modular sizes, allowing flexible composition (Figure 7-21B). The knob transmits an input rotation or button press into different side plates' movements, creating two concurrent transmission *modes*. On the other hand, the trigger and button are designed with a singular transmission mode and I/O pair (Figure 7-21C). We also designed an actuator-sensor block that detects motions (as inputs) and renders haptic feedback (as outputs).

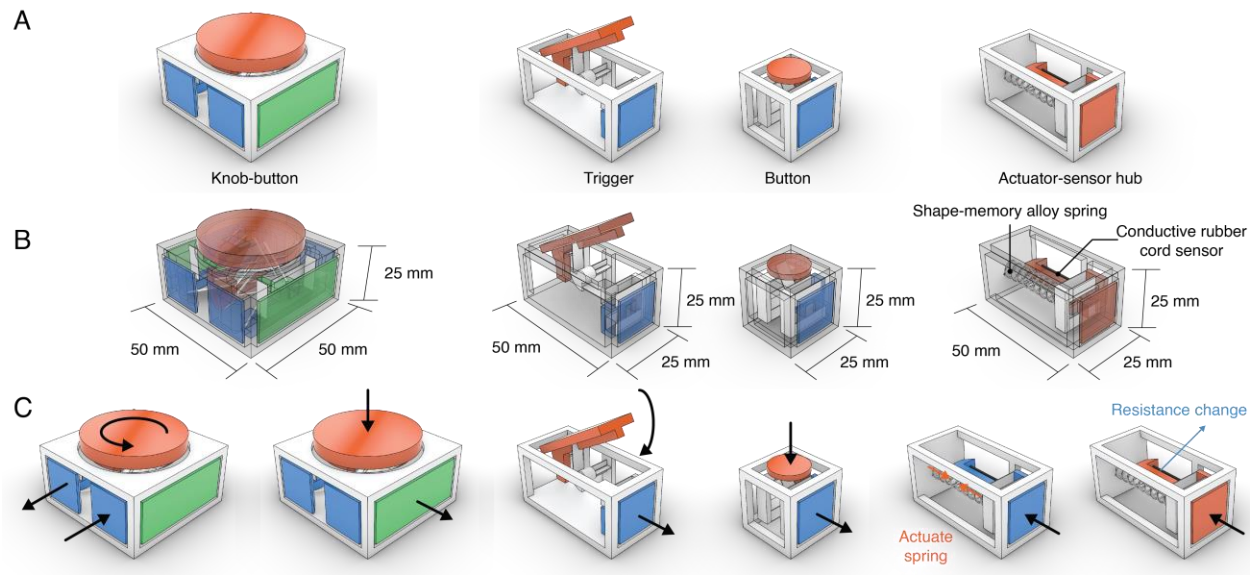

Figure 7-21. Modular interface building block design - (A) different types of building blocks, (B) the devices' dimensions and their internal structure, and (C) their I/O transmission targets.

All physical interfaces are designed with the design tool except for the actuator-sensor hub (Figure 7-22). These building blocks were designed with a modular frame as the fixed end, and the volume inside was designated as the design domain for discretization. Since the transmission design is relatively simple, we allowed the design tool to run in full automation for most of the design process. However, the knob's topology was manually edited to create a direct connection between the input and outputs to further simplify the topology. The simulation results showed that the designed devices can transmit the input end's displacements into lateral translations (Figure 7-23A, Figure 7-24). The output translations were larger than 2 mm, more than 10% of the stretchable sensor's length. On the other hand, the hub only contains two stages connected by one compliant joint; therefore, it does not establish an interconnected topology design problem. The one-joint design scenario is also a canonical CM design problem well-addressed by literature [96, 97, 99, 105].

The interface building blocks may be joined together by, for instance, magnets on the side plates to form a temporary coupling between units. Forces and displacements may then transmit through the connections. For instance, Figure 7-23B illustrates that an actuator-sensor hub may be connected to a trigger to detect pressing events (see also Figure 7-25B, C). The actuator's contraction also generates a force that mitigates the force required to pull the trigger (Figure 7-25A), allowing for dynamically tuning its haptic response. Users may also assemble the building

blocks into different form factors to adapt to the use scenario (Figure 7-23C). This design example anecdotally shows different ways to produce larger, more complex designs other than designing a complex device in one sitting.

While not physically implemented and tested, we speculate that creating physical interfaces with compliant mechanisms and active materials may provide various benefits. For instance, these devices may be quieter in action without electromechanical actuators. The absence of mechanical articulations and motors makes these devices suitable for wet or dusty environments. However, compared to their electromechanical counterparts, active materials also have disadvantages in tangible interaction, such as limited displacement range and lower actuation frequency. Future work may wish to address these challenges to broaden the design space of such material systems and design methods.

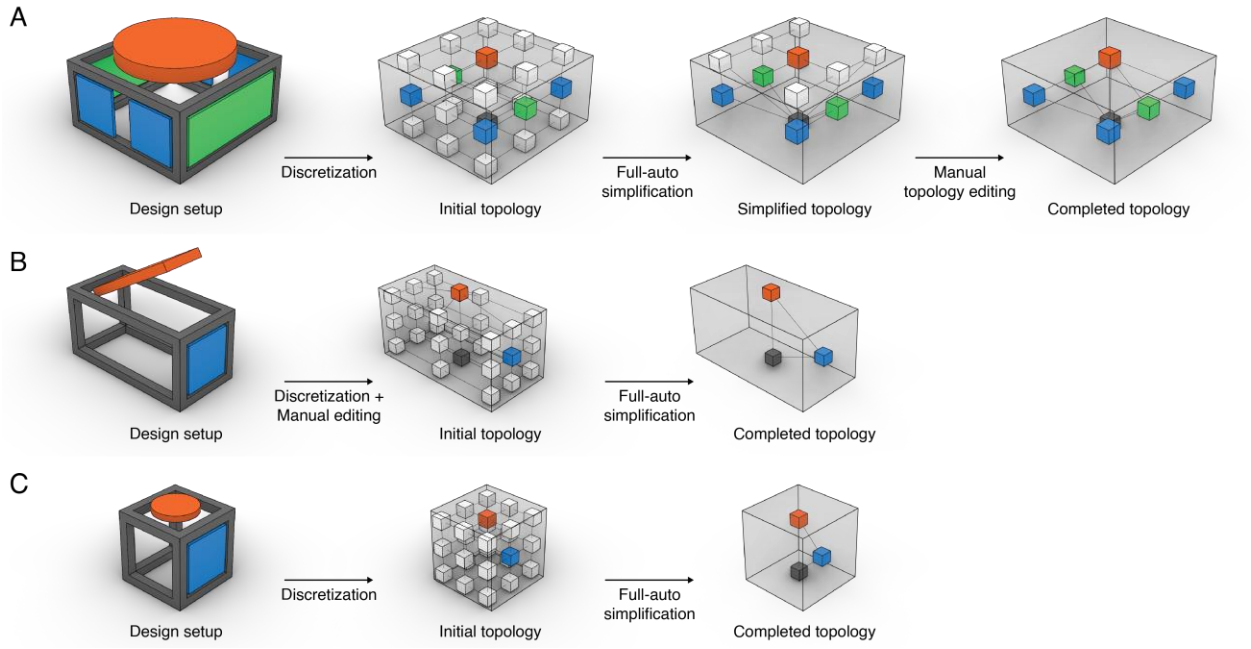

Figure 7-22. The design iterations of (A) the knob, (B) the trigger, and (C) the button.

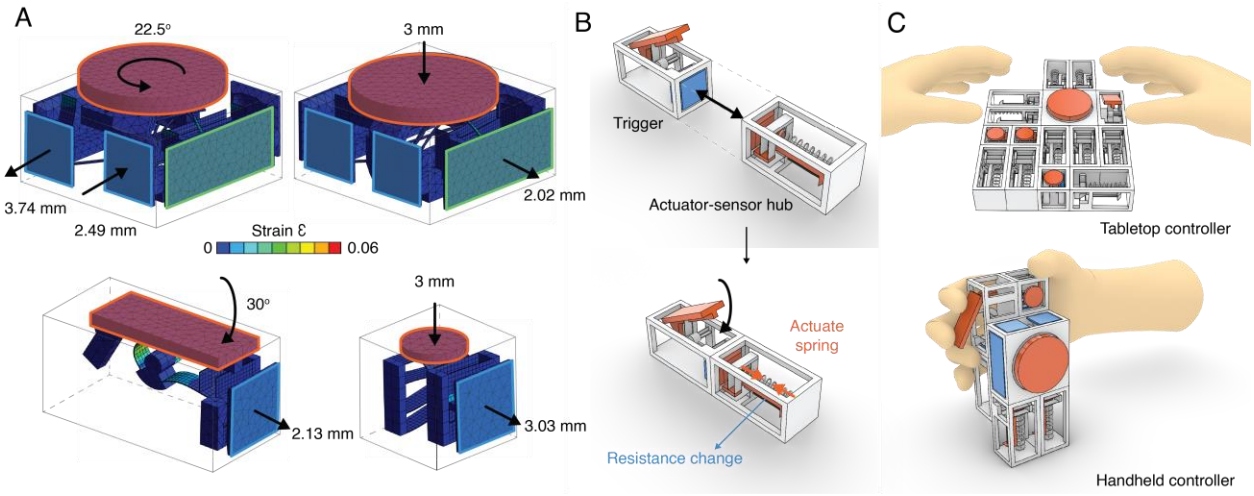

Figure 7-23. Modular physical interface (A) simulation results and assembly examples – combining a trigger and an actuator-sensor hub to create an I/O device and (C) assembling multiple units to create devices with different form factors.

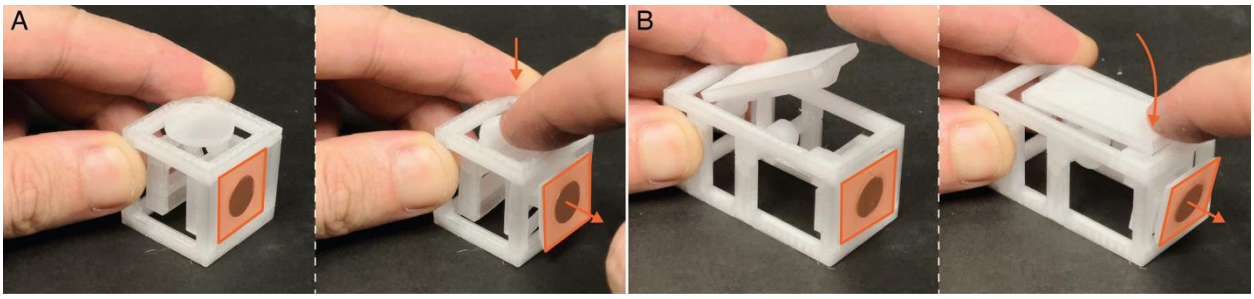

Figure 7-24. Modular interface building blocks transmission – (A) button and (B) trigger input blocks.

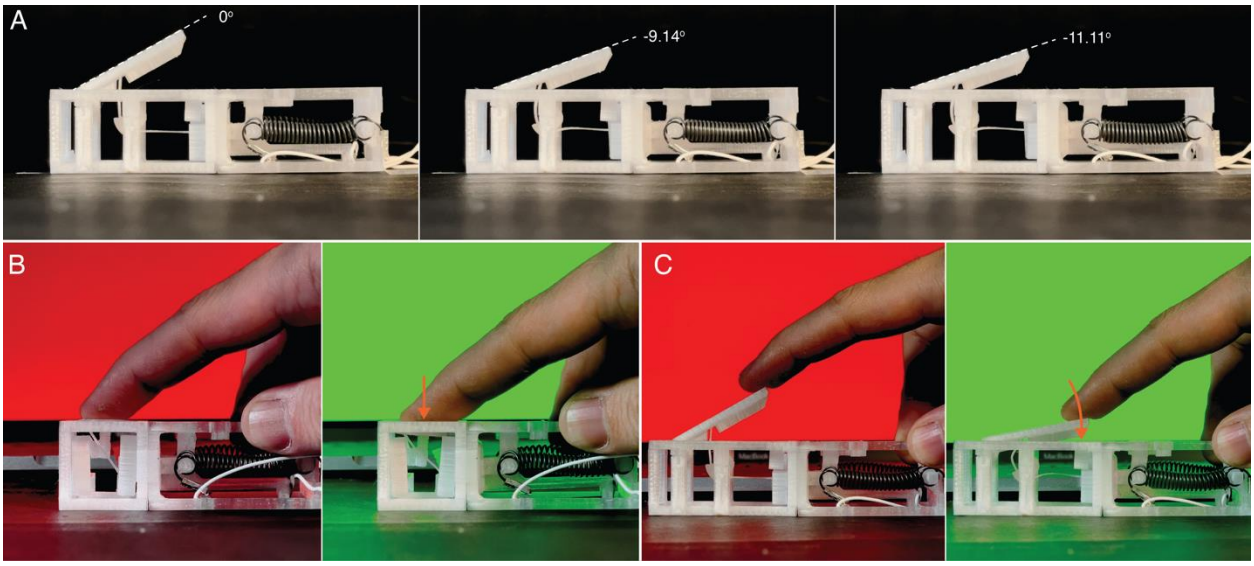

Figure 7-25. Modular interface building blocks combined with actuator-sensor hub - (A) the trigger being driven by the shape-memory alloy spring to provide haptic feedback and (B) the button and (C) trigger press causing the sensor to stretch and trigger the background monitor to change color from red to green.

7.8. Modeling Design Requirements

Numerical Model

Given an interconnected topology and a kinematic *mode* prescription, the displacement of compliant joints and stages could be modeled and solved as a linear system of equations in the form of

$$[A]X = B, [A] = \begin{bmatrix} [A'_1] \\ \vdots \\ [A'_k] \end{bmatrix}, X = (x_1 \oplus \dots \oplus x_e)^T, B = B'_1 \oplus \dots \oplus B'_k$$

eq. 7-1

Where $[A]$ and B are modeled after the transmission *mode's* boundary conditions and the kinematic compatibilities embedded within the graph topology (see the following section), and X describes how each compliant joint should displace in order to satisfy the kinematic *mode*. In particular, $[A]$ and B are attained by concatenating k number of constraints. Each constraint is modeled as a pair $[A']$ and B'. $[A']$ nominates a set of edges and their afforded DOF and accumulates their displacement by multiplication with X to attain the targeted kinematic relation (a twists vector) B':

$$[A']X = B'$$

eq. 7-2

which partially fulfills a fraction of the transmission function. The number of constraints may depend on the topology in question and the number of prescribed motions, and they could be kinematic loop constraints, stage mobility constraints, or transmission constraints. $[A']$ is a $6 \times 6e$ matrix, where the first dimension corresponds to the screw vector length of six, and the second dimension corresponds to each of the six DOF afforded by each joint within the topology (e in total), hence $6e$. Similarly, X has a length of $6e$ and B' has a length of six.

Joints Modeling

The displacement of a compliant mechanism joint could be described using a linear equation:

$$[T]_{6\times6}\, x_{6\times1} = \hat{T}_{6\times1}$$

eq. 7-3

Where $[T]$ is the freedom space of the joint: a full-span square matrix whose columns encode the unit motional freedoms (i.e., three translations and three rotations about the X, Y, and Z axis) passing through the joint center. Here, in a design synthesis problem, since we do not know the DOF needed at a joint to achieve a kinematic function, we start by assuming the joint could afford all six DOF at the same time. When multiplied by x, the joint's displacement velocity along each DOF, the resulting vector $\hat{T}$ then describe a joint's summed displacement.

System Modeling

An interconnected compliant mechanism topology could be modeled by concatenating multiple joints into an extended linear system:

$$[T_{all}]X = [T_1 \quad \dots \quad T_e](x_1 \quad \dots \quad x_e)^T$$

eq. 7-4

The first part concatenates the joints' freedom spaces together and encodes all possible kinematic DOF afforded by the joints within the system, and the second part of the equation concatenates the velocity vectors that correspond to each joint into the extended velocity vector X, which indicates how joints within the topology displace along their possible DOF in synchrony.

Constraint Modeling

$[T_{all}]$ encodes and nominates all joints within a CM topology. However, constraints are often scoped to a few joints and stages within the system (e.g., summing displacement along a path in the graph) Hence, when modeling constraint equations (eq. 7-2), we need to apply a mask and select the joints in question:

$$[A'] = P \odot [T_{all}]$$

eq. 7-5

Specifically, $[A']$ is created by factoring $[T_{all}]$ with a path vector P. The path vector has a dimension equal to the number of edges in the graph and nominates edges, with each entry indicating the traversal direction along the corresponding edge (1: traversal along the edge, -1:traversal opposite to the direction of the edge, 0: ignoring the edge). When traversing along an edge e in the graph, the edge's twist $\widehat{T_e}$ describes the displacement between the two rigid bodies connected by the edge. Note that we use a signed graph to represent the edges and the traversal $\widehat{T_e}$ should also be factored by the sign of traversal direction. Since twist vectors are additive, the displacement of a stage several edges away, with respect to a particular starting stage, could be found by accumulating the displacements as twist vectors along the edges between them.

Once a P is found, $[A']$ could then be created by inserting each joint's freedom space $[T_i]$ (where i is the index of the edge) multiplied by P_i in their corresponding submatrix. This way, the DOF factored by zero will not be accumulated when computing eq. 7-2, and the nominated edges will be factored by their direction.

In the following part, we describe three constraints pivotal to interconnected CM design. Here, we will use the mechanism shown in Figure 7-3A as an example to introduce the modeling of each constraint.

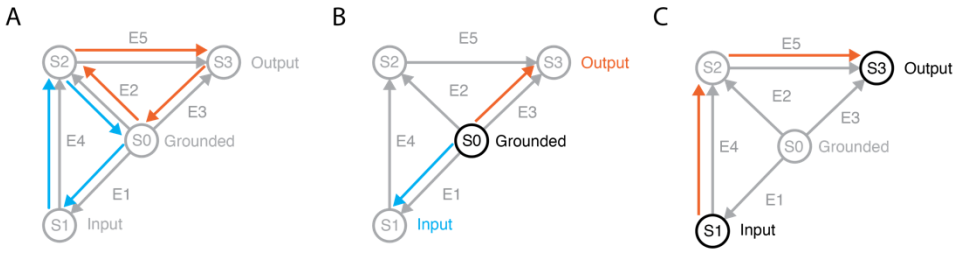

Figure 7-26. Interconnected CM design constraints using the specification and topology in Figure 7-3 as an example: (A) Kinematic loop constraints, (B) stage mobility constraints, and (C) transmission constraints.

Kinematic Loop Constraint

An interconnected CM topology often contains multiple closed loops (Figure 7-1A). These close loops must remain closed with any given transmission or motion generated by the system. The twist vectors at each edge must sum to zero, which indicates that the starting stage does not displace against itself, or the stage would not be rigid. Should the twists along a loop sum to a non-zero

twist, this would indicate that the starting stage would displace against itself, which violates the rigid body assumption.

To model this constraint, we could identify the fewest number of non-redundant kinematic loops [80, 84] in a graph by finding $[Q]$, the left null space of $[C]$. $[Q]$ has a dimension of size $e \times q$, where its first dimension, e, equals the number of edges, and the second dimension, q, is the number of independent, non-redundant kinematic loops in the graph. Each column in $[Q]$ traverses a loop in the graph and shares the same schema with P and should be considered as such. The constraint equations to satisfy a kinematic loop could then be modeled by using each column of $[Q]$ as P in eq. 7-5 and a zero-twist vector as B'.

The simple I/O device (Figure 7-3A) consists of two independent kinematic loops, as can be seen in its graphical representation (Figure 7-26A). The loops are computed to be:

$$[Q]^T = \begin{bmatrix} 1 & -1 & 0 & 1 & 0 \\ 0 & 1 & -1 & 0 & 1 \end{bmatrix}$$

, where the first loop sums among the grounded, input, and intermediate stages, and the second loops between the grounded, intermediate, and output stages.

Stage Mobility Constraint

The stage mobility constraints are used to prescribe a specific displacement to a stage in the interconnected CM. When modeling a transmission system, this constraint is used to prescribe an input or output motion to a stage. Outside of transmission I/O conditions, such constraints could also be used to enforce a rigid body in the mechanism to displace in certain ways.

The path vector P for this constraint should point from the grounded stage to the stage of interest, which could be found by applying Dijkstra's algorithm [58] to the graph's adjacency matrix. The twist accumulated along such traversal should equate to the desired displacement. The constraint equation $[A']$ is then created by eq. 7-5, and B' should describe the stage's targeted displacement as a twist vector.

In the simple I/O device, a twist is prescribed for each of the input and output stages (Figure 7-26B). For the input stage, the path vector $P_{inp} = [1 \quad 0 \quad 0 \quad 0 \quad 0]$ traverses along the first edge from the ground to the input node, and the input displacement - translation along the y-axis - corresponds

to a twist vector $B'_{inp} = \hat{T}_{inp} = [0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0]^T$. Likewise, the path vector $P_{out} = [0 \quad 0 \quad 1 \quad 0 \quad 0]$ points to the output stage from the grounded one along the $[E]_3$, and its twist vector prescription $B'_{out} = \hat{T}_{out} = [0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0]^T$ is a translation along the x-axis.

Transmission Constraint

The transmissive relation is modeled as a cumulative traversal from the input to the output stage and describes the output stage's displacement with respect to the input stage. This constraint could be modeled using the same approach as the stage mobility constraint but using the input stage as the starting node instead. Similarly, B' describes the difference in displacement between the input and output stage and is computed as $\hat{T}_{out} - \hat{T}_{inp}$. If a transmission specifies the relation between multiple inputs and outputs, each input-output pair should be modeled as a singular constraint.

In the simple I/O device, only one transmission constraint is needed as only one transmission pair exists (Figure 7-26C). The path vector points from the input to the output stage: $P = [0 \quad 0 \quad 0 \quad 1 \quad 1]$, and the transmission - as a relative displacement twist vector - between them is computed to be $T_{io} = [0 \quad 0 \quad 0 \quad 1 \quad -1 \quad 0]^T$.

Constructing Linear System

The functional prescription and constraints (i.e., multiple $[A']$ and B' pairs) over the same graph could be concatenated along their first dimension into the linear system $[A]X = B$, and the velocity vector X found by solving the linear system would satisfy all constraints at once. Specifically, each entry of X indicates how fast a joint should move in its corresponding DOF to satisfy all the kinematic loops and a transmission. Therefore, if an entry of x is non-zero, the corresponding joint and its corresponding DOF should be mobile in the interconnected CM design. Conversely, if an entry is zero, then the DOF could be constrained when translated into a CM design.

Note that the linear system could be under, exactly-, or over-constrained. In the first case, x is found as a linear affine subspace embedded in the velocity space, and each point on the subspace is a plausible configuration (i.e., displacements along each DOF at each joint) to achieve the desired transmission. In the exactly constrained case, only one configuration is possible to achieve the desired transmission. Finally, should the system be over-constrained, then no configuration

could possibly afford the desired transmission, and either the initial graph topology or transmission targets should be changed.

The simple I/O device has five joints in total, hence the second dimension of $[A]$ and the dimension of X equates to $6 \times 5 = 30$. The design problem leads to five kinematic constraints, two resulting from the topology (i.e., kinematic loops) and three from the I/O specifications. Consequently, the first dimension of $[A]$ is also $6 \times 5 = 30$. However, one of the I/O constraints is redundant, as the linear combination of the other two readily specifies the full transmission loop (i.e., traversing ground-input-output-ground). Consequently, the linear $[A]x = B$ is under-constrained, and x is found as an affine linear subspace spanned by six independent vectors.

7.9. Solving Design Requirements

Solving the Linear System

Should a solution be available for a linear system, the solution could be described in the form of $X_0 + [X_s]y$, where X_0 is a particular solution to the liner system and a reference point located on the solution subspace, and $[X_s]$ is the solution subspace's span where each column is an independent span vector. $[X_s]y$ linearly combines the span vectors with parameters y to move the particular solution X_0 to produce a different velocity vector, which is also a plausible solution. If the problem is exactly constrained, $[X_s]$ would be empty, and the particular solution is unique and singular. Conceptually, as long as at least one span vector exists, there should be (continuously) infinitely many solutions of x that could be generated in such a way.

However, in the case that the design problem is under-constrained, it becomes pivotal to find a "good" solution to X as opposed to any, random X. While X could be solved using numerical methods, the settings used to locate a particular solution in X could lead to joint DOF designs with different characteristics. A common numerical way to solve this problem is to find a solution with the minimal L2 norm. In this case, the solver would avoid using large numbers to find a solution, which translates to achieving the transmission without using large velocities along a DOF. Structurally speaking, this could be preferred because small displacements lead to small flexure displacement strains. However, an L2-minimized solution would also have more non-zero entries, which necessitates the use of more DOF. In fact, we could often find that an L2-minimized solution

would use all DOF at once. Such a solution is impractical in a realistic setting as it will require complex joint flexure design.

Ideally, we would prefer a sparse solution to X by using L0 regularization. L0-minimization counts and minimizes the number of non-zero entries in X, reducing the number of DOF needed to enable transmission and avoid excessive use of joint freedoms. A sparse solution would indicate that we only need to enable a few DOF in the interconnected CM to provide the desired functions, and more flexures could be added for structural or other reasons. However, L0-minimization is non-convex and not compatible with numerical solvers that assume convexity. Alternatively, we could use L1-minimization (also known as lasso optimization [143] for its sparsity) in defining the convex problem, which compromises between an L0 and an L2 minimizer. I.e., An L1-minimizer reduces the number of non-zero entries but also attempts to reduce the Manhattan length of X. This property often leads to a solution with slightly more non-zero entries than an L0-optimized X and larger magnitudes along individual dimensions than an L2-optimized X, but an L1 minimizer adequately trades off between sparsity and extremity to produce a solution appropriate for CM design.

In summary, we set the L1 minimization term as the objective function and the linear system (eq. 7-1) as a constraint. The formal solver objective is defined as

$$argmin\ \|x\|_1\ s.t.\ [A]x = B$$

eq. 7-6

The first part corresponds to the L1-minimization objective and the second part makes sure the design conforms to the transmission functions and constraints. We implemented the solver in Python using CVXPY and NumPy.

Solving for Mult-IO Transmissions

eq. 7-6 defines the design problem for an interconnected CM device that affords a single kinematic transmission *mode*. This schema could be extended to handle design problems that involve multiple kinematic *modes*. To do this, we first expand the unknown velocity vector X into a stack, $[X]$, of unknown velocities, where each column is a velocity vector corresponding to a transmission *mode*:

$$[X] = [X_1 \quad ... \quad X_m]$$

eq. 7-7

A constraint linear system should then be constructed for each IO *mode* (m *modes* in total) and applied to different columns in $[X]$:

$$min_{[X]}|maxabs([X], dim = 2)|_1, [X] = [X_1 \quad ... \quad X_m]$$

$$s.t.\,[A_1]X_1 = B_1$$

$$\vdots$$

$$s.t.\,[A_m]X_m = B_m$$

eq. 7-8

Note that when designing multiple IO transmission *modes*, in addition to using a few DOF to complete a design, we also want the solver to reuse joint DOF between *modes* as much as possible, as it leads to a simpler flexure design. To encourage this, the L1 minimization objective is instead applied across the columns of $[X]$. We first apply a max function over the absolute values of $[X]$ along each row to produce a vector X' that has the same dimension as X in eq. 7-6. Each cell in X' then corresponds to the highest velocity along each DOF across all *modes*. The L1-norm function is then applied over X' and used as the convex numerical objective. This formulation posits that velocities along each DOF would only be counted once, and reusing a DOF between *modes* will incur no additional cost hence is encouraged.

DOF Reduction Using Joint Re-orientation

In addition to using L1 minimization to reduce the number of DOF required by a solution X, we could also reduce the requirement to one for each *mode* by re-orienting the DOF orthogonal system at each joint. While modeling compliant joint displacements (eq. 7-3), we made an implicit assumption that the DOF axes are aligned with the principal axes when constructing $[T_i]$, and the rotation axes are centered around a joint pivot point. Each cell in X then corresponds to a velocity along these principal-axes-aligned DOF. As a result, a solution may require several of these DOF to produce a non-principal-axes-aligned twist or a rotation distant from the pivot point. This non-

aligned twist vector $\widehat{T_i}$ of joint i required to complete a transmission *mode* could be found by calculating

$$\widehat{T_i} = [T_i]x_i$$

eq. 7-9

where $[T_i]$ is the joint freedom space and x_i is the sub-vector in X that corresponds to the joint i. This way, $\widehat{T_i}$ linearly combines the principal-axes-aligned DOF, which could have arbitrary orientations and rotation pivot points. The number of required DOF is reduced to one per joint per *mode*.

7.10. Joint Velocity Estimation Algorithm

In summary, the algorithm for finding joint displacements to satisfy the transmission functions of a given interconnected CM topology is defined as follows:

Input

The interconnected CM topology is defined by its signed adjacency and incidence matrix $[J]$ and $[C]$, respectively, as well as a pivot point per joint (edge). Multiple desired kinematic transmissions could be prescribed to the design problem, each defined by setting one stage in the topology as the fixed, grounded reference and a set of inputs and output stages. The inputs and outputs are defined by a target stage index and the displacement (as a twist vector) desired for that stage in the transmission. Additional stage mobility requirements (i.e., forced stage movement) could also be provided.

Step 1. Model joint freedom space.

For each joint in the topology, model their available DOF into a freedom space $[T_i]$ using the joint pivot points. A DOF could be excluded from the system by removing its corresponding unit twist vector in $[T_i]$.

Step 2. Modal kinematic constraints.

Construct three types of constraints for the design problem:

Kinematic loop constraints

Given the topology's incidence matrix $[C]$, find the matrix $[Q]$ encoding the independent closed loops as a stack of path vectors. The accumulative displacement by traversing the loops should sum to zero.

Stage mobility constraint

For each input and output stage in each I/O *mode*, construct a path vector pointing to them from the grounded stage using the adjacency matrix $[J]$ and Dijkstra's algorithm. The accumulative displacement by traversing the loops should sum to the velocity at which the stage displaces (as a twist vector).

Transmission constraint

For each input and output pair, construct a path vector pointing from the input to the output using $[J]$ and Dijkstra's algorithm. The displacement accumulated along the path should sum to the output stage's twist minus that of the input stage.

Step 3. Model problem linear system.

For each path vector produced in step 2, multiply the joints' available DOF matrices $[T_i]$ by their corresponding cell and insert them as submatrices to produce constraint equations, creating the matrix $[A']$ in the linear algebraic problem $[A']X = B'$ (eq. 7-2). B' encodes the desired kinematic relation as a sum of displacement twists. All constraint equations modeling an I/O *mode* should then be concatenated to form the complete linear system $[A]x = B$. For designs involving multiple kinematic transmission *modes*, a linear system should be constructed per *mode*, and the unknown velocity X is expanded into $[X]$ as a stack of velocity vectors.

Step 4. Solve the design problem.

The solutions of X can be modeled as an affine linear subspace defined by a particular solution and set of span vectors. A particular solution requiring a small number of joint DOF could be found by minimizing the L1-norm of X with the linear system $[A]X = B$ as a constraint. The span vectors could be found by finding the null space of $[A]$. If the design involves multiple *modes*, minimize instead the L1 norm of the (unsigned) maximum velocity of each DOF across all X vectors, and

the span vectors should be calculated per *mode* using their corresponding $[A]$. If no solution to x can be found, then it indicates that the kinematic relations are impossible to realize given the provided topology, and either the input topology or the design objectives should be modified.

Step 5. Solution refinement.

The solution found in the previous step could be modified by a linear combination of their span vectors. The combination could be arbitrary and would always satisfy the specifications. Alternatively, the DOF requirements of each joint could be further simplified by multiplying $[T_i]$ with their corresponding velocity in x_i in X to produce a singular twist vector. The required DOF for that joint is then reduced to one and may no longer be axis-aligned.

Step 6. Identify the required DOF.

The mobility requirements of each joint could be determined by their corresponding cells in x_i. If a cell is non-zero, then the joint's corresponding DOF should be enabled. If the cell is instead zero, then the DOF could be omitted. A design involving multiple simultaneous transmissions could be solved using the same method, and a DOF should be allowed if any of its corresponding velocity is non-zero in $[X]$.

For designs involving reconfigurable smart materials, the reconfigurability could be identified by checking their values between different *modes* and *configurations*. If a DOF's corresponding velocity is all zero in one *configuration* and non-zero in the other(s) and vice versa, then the DOF should be designed with reconfigurability to satisfy different configuration's requirements.

Output

The output of the algorithm is the mobility requirement of each DOF at each joint in the topology. The DOF deemed necessary by the algorithm creates a freedom space per joint (i.e., spanned by the DOF found in Step 5 and 6), which could then be converted into modeling instructions using the FACT method [97, 98] or ReCompFig's single-joint modeling tool [335].

7.11. Design Modification Heuristics

In the iterative algorithm, the order of applying these rules is also deliberately structured. We prioritize topology simplification rules over decoupling fix rules because topology simplification is best suited for removing redundant components in the topology, therefore reduces the computation time of velocity estimation and decimates the design problem complexity at the early stage. The following sections provide a description of each heuristic rule in the order of application priority.

Topology Simplification: Merging Synchronized Stages

Based on the velocity estimations, stages displacing at the same speed (i.e., twist) can be merged into a single rigid body to simplify the topology (Figure 7-27). These synchronized stages have zero displacements between one another, resembling rigid connections. Consequently, these stages could be combined into a rigid body without compromising the transmission functions.

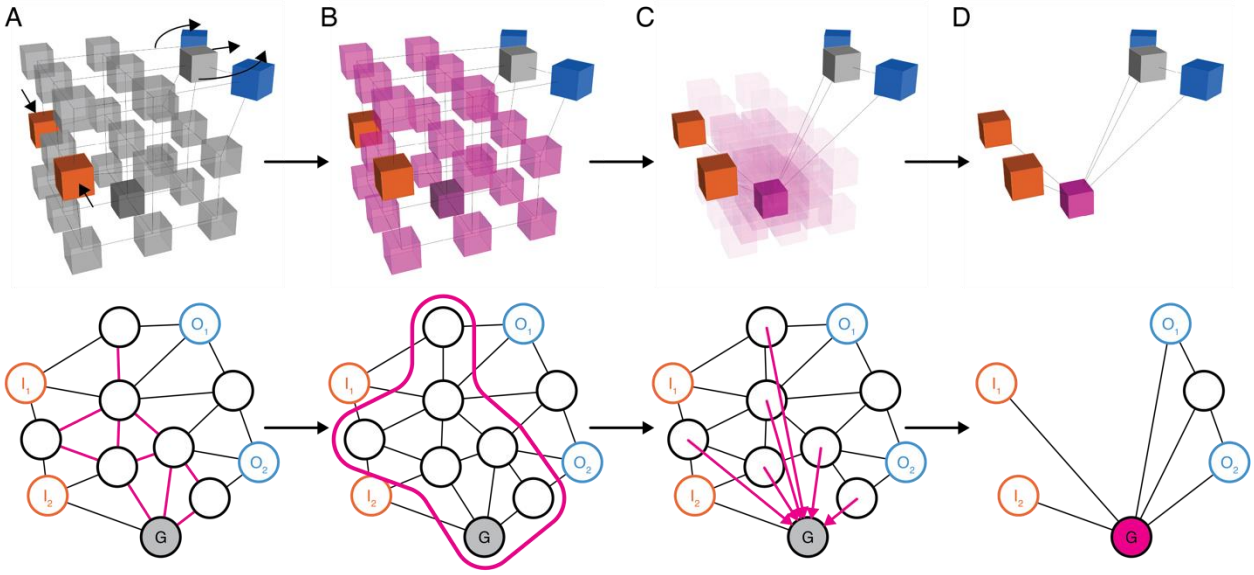

Figure 7-27.A visual example of Topology Simplification: Merging Synchronized Stages. The top row shows the topology, and the bottom row shows the abstract graphical representation. The example design contains two inputs and outputs. (A) The velocity estimation over the topology informed stage displacements. One intermediate stage had complementary motions with the I/O stages, while the rest remained still. Edges with zero displacement are highlighted in the graph. (B) Stages with zero displacements are grouped together, including the grounded stage. (C) Grouped stages are merged into a single body (merging into the grounded stage), producing (D) the updated topology and graph.

To *check* for these merging opportunities, we examine the velocity estimation and identify compliant joints that have zero displacement velocity across all *modes* and *configurations* (Figure

7-27A). Such joints require no DOF to achieve any of the transmissive functions, and a *modification* could be made to the graph to combine the two nodes connected by the joint and simplify the topology. A topology and velocity estimation may result in multiple joints that have zero displacements (Figure 7-27B, C), and the rigid bodies connected by these joints could be merged into a single body (Figure 7-27D).

It is worth noting that merging stages may also cause transmission gear trains to become invalid. Specifically, a transmission gear train is only valid when a graph cycle exists (i.e., a loop within the graph without visiting any node more than once) between the input, output, and fix stages. However, merging stages may cause part of the gear train to collapse and invalidate the transmission (Figure 7-28), therefore requiring attention after applying the heuristics.

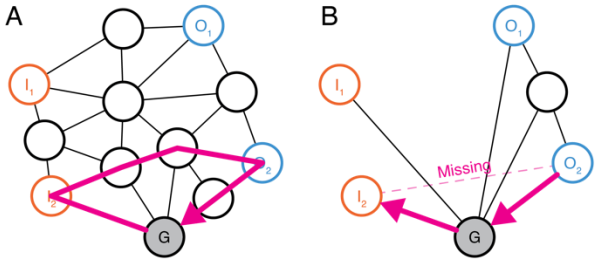

Figure 7-28. A visual example of transmission gear train collapse, continuing the example shown in Figure 7-27. (A) The topology had a valid gear train between I_2 and O_2 prior to merging stages. The loop cycles through the grounded, input, and output stages and returns to the same node. (B) After merging stages, the gear train collapses, and the transmission cycle misses an arc connecting I_2 and O_2.

We used a gear train preservation algorithm (Figure 7-29) over the merged graph to ensure a cycle between the I/O and fixed stages. The algorithm assumes that all stages are connected by existing or virtual edges (Figure 7-29A), and the gear train could be found – or reconstructed – by finding a minimal-cost cycle that passed through the fixed, input, and output stages in order and arrived at the fixed stage again (Figure 7-29B-E). The costs signify the availability of an edge between the two nodes and could be expressed with a symmetric matrix $[M']$:

$$[M'] = \begin{bmatrix} M'_{11} & \cdots & M'_{1n'} \\ \vdots & \ddots & \vdots \\ M'_{n'1} & \cdots & M'_{n'n'} \end{bmatrix}, M'_{ij} = \begin{cases} 0 \; if \; i = j \\ \varepsilon \; if \; [A']_{ij} \neq 0 \\ Dist([A])_{ij} \; if [A']_{ij} = 0 \end{cases}$$

eq. 7-10

Where the cost (M'_{ij}) of traversing between nodes i and j are determined based on the adjacency matrices before ($[A]$) and after ($[A']$) stage merge. Specifically, traversing an edge that persisted through the merge incurs a minimal cost of $\varepsilon = 1^{-6}$. Conversely, if an edge collapsed during the merge, the cost to traverse (i.e., reconnect) the edge is based on the arc distance (i.e., number of edges) between the two nodes in $[A]$. This way, the pathing algorithm will attempt to find a cycle using readily available edges in $[A']$ as much as possible. However, if a path cannot be found between two waypoint nodes, as few as possible short edges will be added to establish the gear train. This algorithm is applied for each I/O pair of each *mode* to make sure all transmissions are valid. Reconnecting an edge will change its traversal cost M'_{ij} from $Dist([A])_{ij}$ to ε to promote subsequent pathfinding to reuse the newly constructed edge.

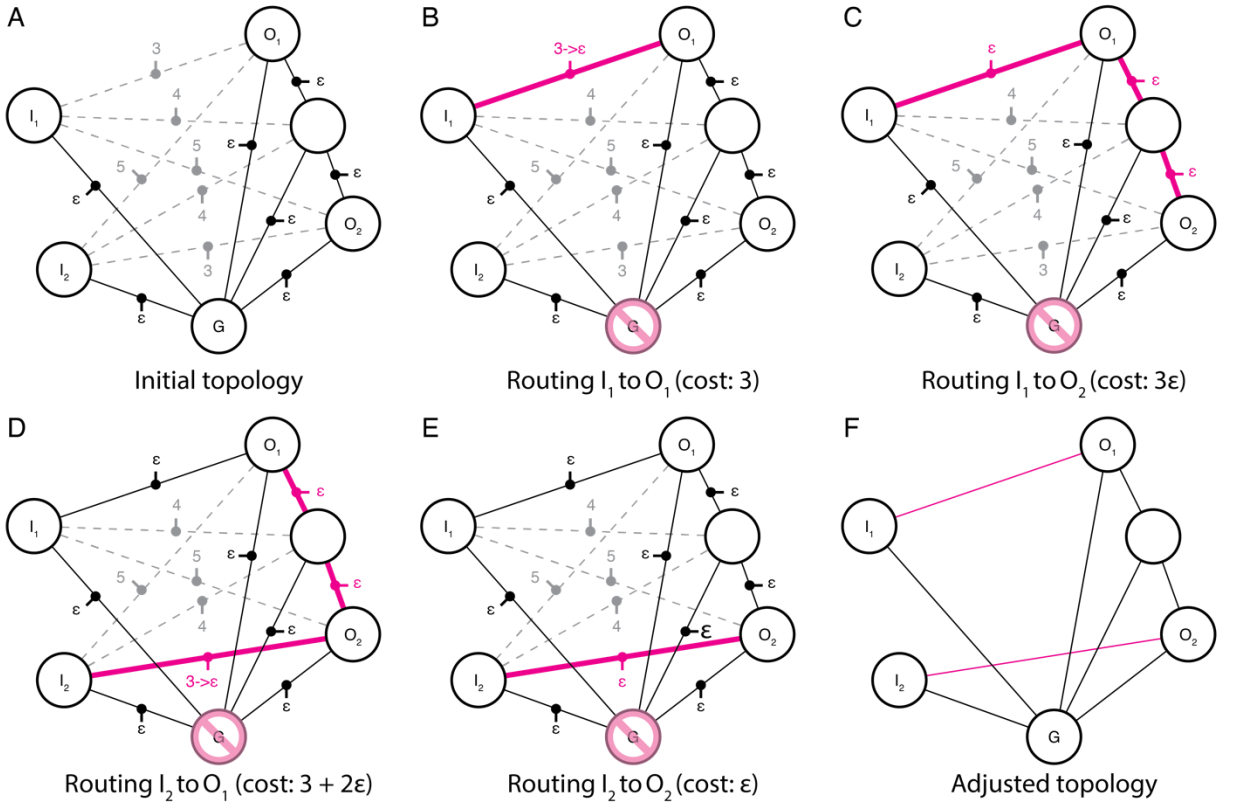

Figure 7-29. A visual example of transmission gear train preservation algorithm, continuing the example shown in Figure 7-28. Note that each I/O stage could be pathed to the grounded stage, so only the gear train between inputs and outputs is examined here. (A) The topology after merging stages and each edge's calculated traversal cost. (B) A direct connection between I_1 and O_1 was unavailable, so new edges must be added. An edge was selected with the lowest, changing its cost from 3 to ε in subsequent steps. Note that the grounded stage is fixed and cannot displace to transmit displacements; hence, it must be ignored in pathfinding. (C) A direct connection between I_1 and O_2 was available with the updated graph. Hence, no edge was added. These steps were repeated for connections (D) I_2 and O_1 and (E) I_2 and O_2, arriving at (F) the corrected topology.

Topology Simplification: Collapsing Linear Arcs

A linear arc is a subset of serially connected edges within a graph that is connected to the rest of the graph at only the end and starting node (Figure 7-30). In graph-based kinematics analysis, these arcs have identical design implications as a single joint: the DOF/DOC of a serially connected arc could be equivalently established by a single compliant joint [99]. Linear arcs could be identified by *checking* for two-degree (i.e., number of incident edges) nodes within the graph, and the graph could be simplified with a *modification* that replaces the arc with a bypassing edge between the start and end node.

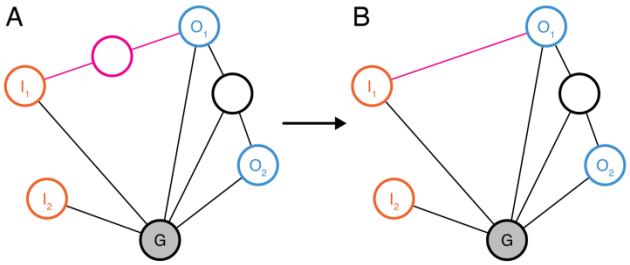

Figure 7-30. A visual example of Topology Simplification: Collapsing Linear arcs. (A) A linear arc could be found between I_1 and O_1 containing one node and two edges. Note that while I_1 is also a part of the linear arc, it is also specified as an input node and should be exempted from topology simplification. (B) The arc could be replaced by a direct connection between the starting and ending nodes, I_1 and O_1.

Topology Simplification: Removing Saturated Joints

An edge within an interconnected CM topology may be removed if all its six DOF are used under a kinematic *configuration*. In this case, the edge is saturated by DOF and must afford zero DOC between the connected stages, leading to no legal flexure placement. Hence, the joint could be removed to simplify a design (Figure 7-31). To locate these opportunities, we could *check* the rank of each joint's twist vectors under a *configuration*. Specifically, the rank of twists is the number of kinematic DOF needed at the same time, and if a joint requires rank-six twists under a mode, all six DOF are used, and the joint is saturated. The edge could then be removed in the subsequent modification.

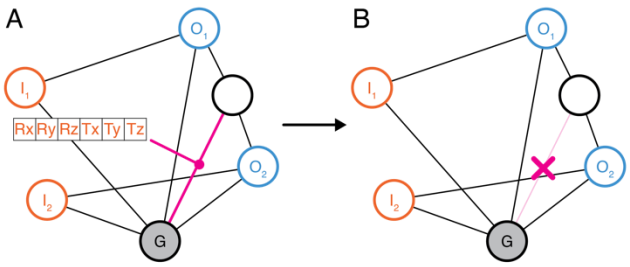

Figure 7-31. A visual example of Topology Simplification: Removing Saturated Edges. (A) An edge with using all six DOF under a configuration should be (B) removed to simplify the graph.

Decoupling Fix: Intra-modal

In a transmission *mode*, the input and output nodes' motions are decoupled when the input can displace independently without incurring displacement of the output node. Consequently, a force or displacement applied to the input stage could lead to no response on the output side (Figure 7-32). This is an inherent issue when using L-1 normalization during velocity estimation. The solver finds a sparse solution to the transmission problem using a few DOF. For any stage within the gear train that is not the input or output node, their displacement is unconstrained by any prescription, making it possible for the solver to assign a zero displacement. The stage would then remain idle in the gear train and prevent displacements from transmitting through (Figure 7-32A). Alternatively, a joint could also be assigned with the same DOF that transmits through it, causing the input displacement to be absorbed by the joint's compliant motion (Figure 7-32B, C). To avoid these issues, the displacements transmitted along the gear train must be preconditioned to avoid decoupling.

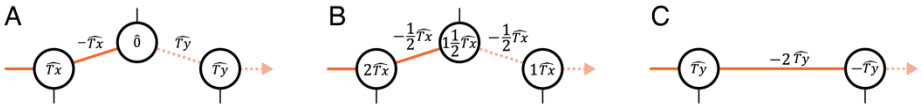

Figure 7-32. Transmission decoupling examples, assuming all displacements as translations along the principal axes. The values over each node and edge show their displacement along an axis with a scalar velocity. (A) A gear train (orange arrow indicating flow from input to output) could be decoupled by an idling stage where displacements cannot pass through. (B) Part of the gear train could also be assigned with the same DOF at different velocities, and the edges connecting them must comply with motions in the same DOF. However, the edge could absorb all displacements along the same DOF, leading to no out-flowing displacement on the other side. In the image, the edge connecting the first and middle stages would absorb all displacements from the input side, causing the middle stage to idle and produce a problem identical to (A). (C) Decoupling could also occur when two edges are assigned with the same DOF but different signs, creating a "reversal" situation. The edge would also absorb all inflowing displacements.

The decoupling *check* traces the displacements along a gear train to find edges that use the same DOF as the motion flowing through it. Computationally, this check could be performed by evaluating the following expression:

$$\left\|null(\hat{T}_e)\hat{T}_{n1}\right\| = 0 \; or \; \left\|null(\hat{T}_e)\hat{T}_{n2}\right\| = 0$$

eq. 7-11

where $\hat{T}_e$ is the edge's displacement twist, and $\hat{T}_{n1}$ and $\hat{T}_{n2}$ are the incident nodes' displacements. The nullity of $\hat{T}_e$ finds the DOF unused by the joint (with a size of 5×6), and the vector computed by $null(\hat{T}_e)\hat{T}_n$ decomposes the in-flowing twists into components along each unused DOF. If an in-flowing twist contains no components along the unused DOF (i.e., $\left\|null(\hat{T}_e)\hat{T}_{n1}\right\| = 0$), the displacement will be absorbed by the compliant joint and cause decoupling. Conversely, if the in-flowing twist contains components along any unused DOF, the component will not be attenuated by the joint's compliance and will be transmitted through the edge.

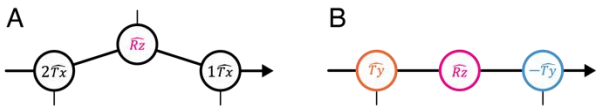

Figure 7-33. Decoupling fix examples. (A) Node biasing could be used to correct idling nodes or prevent a series of nodes from sharing the same DOF. (B) Edge splitting could be used when an edge has I/O nodes on both sides.

Once a decoupling edge is identified, a *modification* – or resolution - could then be formulated by biasing the input or output node's displacement (only one is needed) with an additional component to make sure the displacement will not be completely absorbed by the incident joints (Figure 7-33A). Note that the bias cannot be applied to an I/O or fixed stage as it would cause conflicting constraints (i.e., two different displacements applied to a node at the same time). If the two nodes are both I/O or fixed nodes, the *modification* should instead insert a control node to the edge followed by biasing the control node's displacement (Figure 7-33B). In either situation, such biasing could be numerically applied in four steps:

Step 1: Identify unused DOF.

A stage's biasing component should be unused by its incident joints. The unused DOF could be found by computing:

$$[T_{unused}] = null([T_{used}])^T$$

eq. 7-12

Where $[T_{used}]$ is a collection of (column) twist vectors used by the node's incident joints within the gear train, and the collection of unused DOF $[T_{unused}]$ is found by the used twist vectors' null space.

Step 2: Select unused DOF

Each column of $[T_{unused}]$ represents a unit twist vector $\hat{T}_{bias}$ that could be used to bias the node's displacement. All these twists could not be absorbed by incident joints, and we should pick from these options to formulate a fix. We note that there is no definitive way to select a DOF, and the decision could be affected by varying design considerations (e.g., prioritizing rotations over translations or vice versa, prioritizing DOF along a particular axis). Still, in our implementation, we set our selection to use translations over rotations by default.

Step 3: Determine the magnitude of the bias.

The biasing displacement $\hat{T}_{bias}$ is unitary and could be factored by a magnitude scalar α, and it is valid as long as the magnitude is non-zero (i.e., $\alpha \neq 0$). Still, larger α will cause joints to deform extravagantly, and small α will cause displacements to become mechanically disadvantageous (i.e., driving large subsequent motions with a small displacement). Thus, we formulate the magnitude by the average of its neighboring nodes' displacements to ensure a proportionate motion:

$$\alpha = average(\left\|\hat{T}_n\right\|_{screw} \; \forall \; adjacent \; nodes)$$

eq. 7-13

the screw norm ($\left\|\hat{T}_n\right\|_{screw}$) measures the adjacent nodes' displacement twist magnitudes, which leads to both a rotational and translational quantity per twist. These quantities are then averaged among all adjacent nodes and applied to $\hat{T}_{bias}$ using the corresponding value (i.e., rotational magnitude for a rotational $\hat{T}_{bias}$ and vice versa).

Step 4: Augment the constraint equations.

The biasing fix is applied to consequent velocity estimations by augmenting the *mode's* constraint equation $[A]x = B$. New constraint equations $[A']X = B'$ (eq. 7-2) could be constructed by tracing

a path vector P_{bias} that points from the fixed stage to the stage for biasing, inserting P_{bias} into eq. 7-5 to derive $[A']$, and finding $B' = \alpha * \hat{T}_{bias}$. Subsequent velocity estimations would then force the biased stage to displace with the specified twist and resolve transmission decoupling.

Decoupling. Fix: Inter-modal

In addition to the intra-*modal* scenario illustrated above, transmission decoupling could also occur between concurrent *modes* under a *configuration*. In this case, a joint should provide more than a single DOF to enable these transmissions to take place at the same time, and within a gear train, a *mode's* in-flowing displacement is absorbed by the DOF added to enable *another* mode. To *check* for this issue, we could modify eq. 7-11 to consider cross-modal DOF/DOC interactions. Specifically, $\hat{T}_e$ should be replaced by $[T_e]$, a collection of DOF (as column twists vectors) used by simultaneous *modes*. The expression then evaluates if a *mode's* in-flowing displacement is completely absorbed by the concurrent DOF. Similarly, when formulating a *modification* to fix the decoupling issue, the collection of incident joint twist vectors $[T_{used}]$ in eq. 7-12 is also expanded to cover all twist vectors across simultaneous *modes* to find a biasing twist that is unused under that *configuration*. Nonetheless, the biasing magnitude α should still be computed per *mode*, and the augmenting constraint equations should be applied to the corresponding linear systems.

We note that the changes made to eq. 7-11 and eq. 7-12 is a generalization of the intra-*modal* case. Applying the inter-*modal* version of *check* and *modification* could also address intra-*modal* decoupling. Therefore, when using the iterative algorithm, only the inter-*modal* decoupling heuristic is needed.

7.12. Numerical Validation

Test setup

We used finite element analysis (FEA) to validate the effectiveness of the algorithm and design tool. Compared to physical experiments (e.g., loading with Instron machines), FEA allows us to set up different load conditions and measure multiple output displacements at the same time, providing a convenient and repeatable way to examine transmissive CM devices consisting of multiple I/Os. We analyze the designs using Ansys's static structural analysis function with

isotropic material settings (Ultimaker Polycarbonate, elastic modulus: 2579 MPa, Poisson ratio: 0.3). In the experiments, a displacement load is applied to the input stage with corresponding magnitude, and we measure the output stages' displacements against design specifications. Two devices were designed using the tool (Figure 7-34). These devices' rigid stages are modeled with simple geometries (i.e., rectangular boxes and cylinders) to maintain generalizability. The flexures are modeled according to the design tool's suggestions using flexural blades and rods.

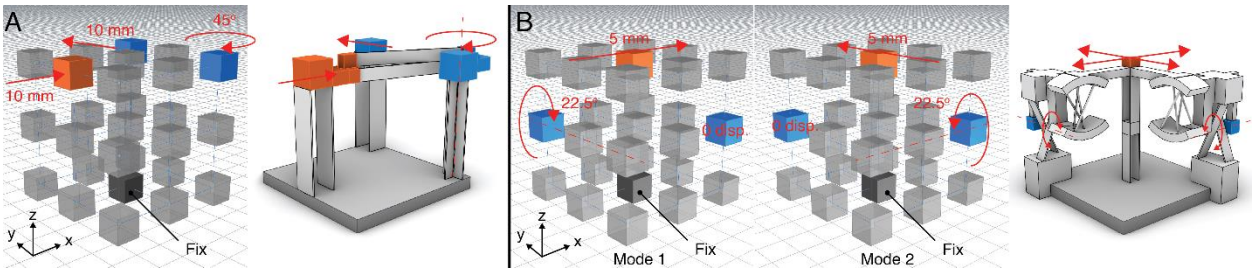

Figure 7-34. Devices for validation. Images show design specs (left) and the modeled test samples (right). (A) A simple device consisting of one kinematic input and two outputs. (B). A dual-transmission device consisting of one kinematic input and two potential outputs. The input's XY-translations drive separate outputs.

Transmission and DOF Validation

The first device was used to check if the generated design affords the specified transmission functions (Figure 7-34A). The device converts the input stage's X-translation into Y-translation and Z-rotation on two stages on the distal end (Figure 7-35A, B). Figure 7-35C showed a good agreement between the stages' mobilities and the transmission targets. Both stages responded along the targeted DOF in the presence of an input motion. The targeted DOF had at least a magnitude higher displacement than the rest. However, due to compliant mechanism's inherent nonlinear structural behaviors, the output motions were not constantly proportional to the input. This observation is typical for compliant mechanisms with large deflections (i.e., >10% of flexure length; our examples have 20%) [105, 300]. On the other hand, the output stages have minimal displacements along the other DOF. This result suggests that the designed device was able to prevent undesired transmissions (i.e., motions not included in the specification).

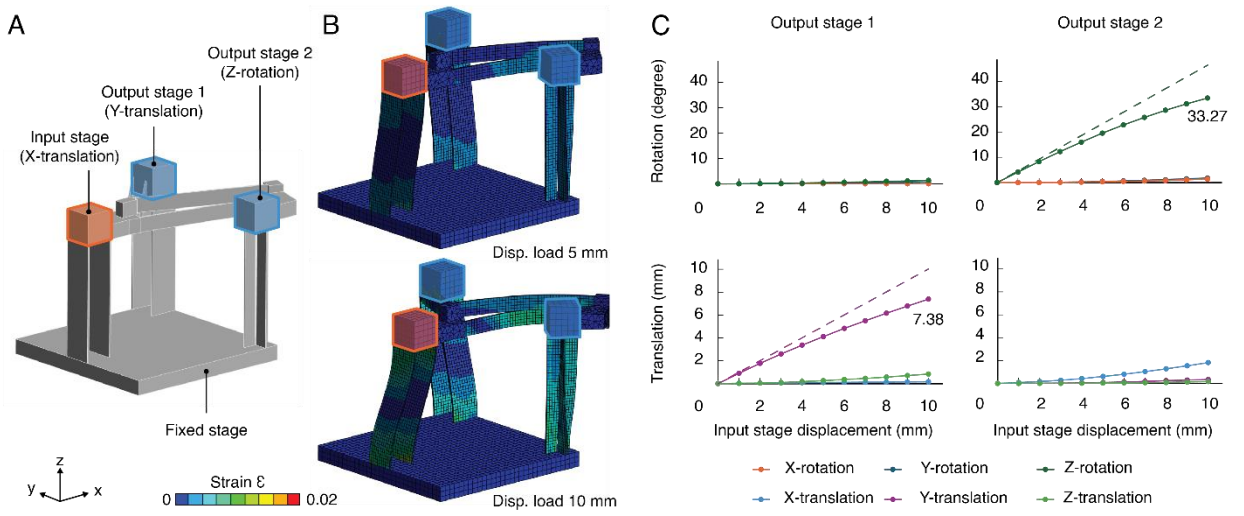

Figure 7-35. Simple IO device simulation result - (A) test setup, (B) visual results, and (C) measured displacements along each DOF. Dashed lines in (C) show the expected displacement curve. Solid lines show simulation results.

Multiple Transmission Mode Validation

The second device was used to check if the generated design could afford multiple simultaneous transmission modes (Figure 7-34B). This device maps two independent (X & Y) translations on the input stage to corresponding rotations along the other axis, creating two transmission gear trains (Figure 7-36A). I.e., the input stage's X-translation will trigger output stage 2 to rotate about the Y-axis, and the input stage's Y-translation will instead trigger output stage 1 to rotate about the X-axis. The transmission modes may be independently or combinatorically activated depending on the input stage's displacement DOF (Figure 7-36B). Simulation results show that both transmission modes are decoupled from each other. When one transmission mode is activated, the other output stage remains nearly still (<0.87°, 5.1% of the mobile output stage's displacement). Yet, when both modes are activated at the same time (i.e., the input stage moves diagonally along the X-Y plane), both output stages become mobile and displaced with a comparable magnitude. This result indicates that the designed device could afford multiple independent transmissions simultaneously.

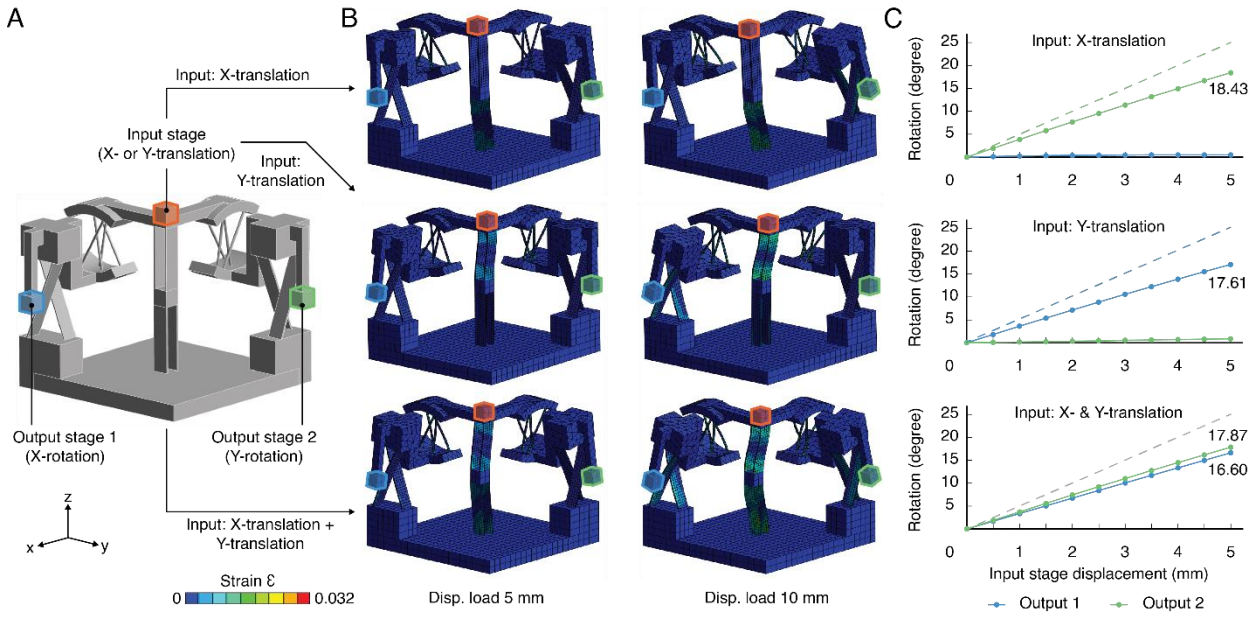

Figure 7-36. Double IO device simulation result - (A) test setup, (B) visual results, and (C) measured output stages displacement. Each row shows a different input condition; from top to bottom: input stage moves along X-translation, Y-translation, and both translations at the same time. Dashed lines in (C) show the expected displacement curve. Solid lines show simulation results.

7.13. Discussion and Limitation

Topological Complexity

The topological complexity needed to complete a design may depend on the transmission specifications: the number of *configurations* and *modes* needed in one device and what combinations of displacements are prescribed to the I/O stages. Intuitively, the more *modes* and varying displacements are prescribed simultaneously; the more compliant joints are needed to create valid transmission gear trains between I/O stages. To the best of our knowledge, it might be difficult – if not impossible - to predict what topology is needed to complete a design task. Still, as a rule of thumb, the number of edges between any input and output node should be proportional to the number of concurrent *modes*. An intermediate stage (i.e., a stage between I/O) can arithmetically compound the displacements it receives, allowing modal actuation to remix and pass through. Hence, the number of edges between any I/O pair should be at least or equal to the maximum number of concurrent modes under any configuration to allow signal combination. I.e., one edge for unimodal designs, two edges (one intermediate node) between I/O for two-modal designs, etc.

Nonetheless, adding additional intermediate stages may be useful in certain design scenarios, such as routing gear trains through a turn. An intermediate stage placed at the corner may be used to redirect the displacement direction (e.g., an x-translation could be transformed into a y-translation by using a z-rotation, Figure 7-37). This technique could be useful in design scenarios containing impassable spaces between the I/O stages.

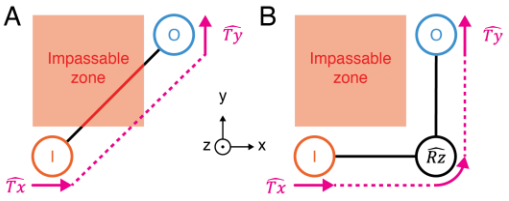

Figure 7-37. Using intermediate nodes to circumvent impassable spaces. (A) The edge between the input and output stages trespasses the impassable zone (which may be designated by the design problem), causing the design to be invalid. However, (B) the impassable zone could be circumvented by rerouting the gear train via an intermediate stage to redirect the displacements.

Scalability

Interconnected compliant mechanism designs may also scale to dimensions other than the mesoscale (i.e., centimeter scale) examples shown in this work. Compliant mechanisms are widely used in milli- or micrometer scales to produce microelectromechanical systems (MEMS). Larger scale structures and systems, such as passenger vehicles, also consist of functional CM subsystems in a meter scale (e.g., airplane wings, automobile leaf springs). Still, we note that there are different challenges in scaling interconnected CMs. Fabrication becomes a challenge when scaling down interconnected CMs. This issue further worsens with the integration of active materials, which demands a multi-material fabrication method. In this case, either part assembly or advanced monolithic manufacturing methods (e.g., embedded printing [8]) may be required.

On the other hand, scaling up interconnected CMs may face challenges in kinetics design. Based on the Euler-Bernoulli beam theory, flexural stiffness is a quartic function of its dimension. As a result, compliant mechanisms in the meter scale may be thousands of times stiffer than their mesoscale counterparts. Thus, designing devices in this scale may require more attention to their kinetic limits, potentially switching to more suitable material like metals.

Kinetics design

The algorithms presented in this work are focused on kinematics design, relating only to a device's displacement behavior. Still, certain load-sensitive application scenarios, such as wearable haptics or the integration of small force actuators (e.g., thermoplastic stress [342]), may require the designer to consider the interconnected CM's stiffness against a design criterion. We acknowledge that this is a current limitation, and future works may consider using Turrkan et al.'s screw algebraic kinetic analysis method [300] to aid in flexural system design. I.e., in addition to the topology and flexural joint DOF requirements handled by the presented work's algorithm, the flexural joints modeling step could be combined with a kinetic simulator to generate a flexure for targeted device stiffness automatically.

Transmission Systems

In this work, the interconnected compliant mechanisms design only considers conventional flexures that behave like Euler beams. The flexures and joints are also permanently connected into a continuum; while displacements can add up and multiply in an arithmetic manner, such a mechanism cannot produce digital logic gate behaviors like those demonstrated by Ion et al. [111], which requires signal (displacement) thresholding and dynamic decoupling between I/O nodes. E.g., In an AND gate, when one input signal is 1 and the other is 0, the output should be 0. A combination like this is inherently I/O-decoupled as one input has motion, but the output does not. As a result, this structure cannot be created by this work's algorithms and structural assumptions. However, future work may consider using contact-aided compliant mechanisms [173, 207, 299] to allow signal thresholding, which only allows displacements and forces to pass through when flexures deform past a certain threshold and touch other rigid bodies.

Computational Interactivity

The computational design tool provides forward, inverse, and suggestive modes to help users find a design solution. Designers could choose between these modes at different stages of the design process. Anecdotally, we found that inverse design functions effectively reduce the design problem at the early design stage through topology simplification. The resulting topology often has a magnitude fewer stages and joints than the unprocessed design, making it easier for the designer to manipulate and reason. However, the resulting topology may also require manual editing to

clean up the generated topology, as seen in the amphibian robot example (Figure 7-13B-C). The simplified topology may contain irregular stage placements (i.e., a merged stage's position is the spatial centroid of its parents, which could be somewhat random), which could be adjusted to reflect form factors that the tool is unaware of, such as symmetry. In this case, the designer's intervention is required to bypass the tool's rigidness, reflecting the toolmaking motifs proposed by this thesis.

On the other hand, providing all design modes for fixing node biasing makes the tool more flexible in solving a design problem. We note that there is no definitive way of deciding on a node biasing twist. For instance, both rotation and translation biases are effective at fixing the issue, but there may be a tradeoff between flexural complexity and mechanical advantage between the two. The magnitude and direction of the biasing twist are also flexible if they are non-zero. In this situation, designers may select the most appropriate method to proceed with the modification. If the design designer is unconcerned with the gear train's displacements, they may allow the design tool to automate the decision. However, if the designer wishes for more control, they could either – in ascending order of user control - alter the heuristic's decision priorities to generate a different *modification*, pick from the tool's calculated options, or manually apply their own *modification* - in ascending order of direct user control. Admittedly, the latter two options require more user input, but they also allow the designer to reflect and act upon opportunities or factors that arise during the design process.

The design tool can generate alternative design solutions by applying different modifications at each design step. Users may then generate alternative design solutions by testing out different (sequences) of modifications to iterate a design. At each step, the design tool's interface also shows the options the user may take to iterate the design, such as suggesting biasing twists. This mechanism of generating design variations differs from conventional methods that navigate a fixed-dimension parametric space [179, 259, 260]; such a process is Markovian, and variations are situated in a decision tree. However, providing only undo and redo buttons may not be the most effective design history control mechanism to navigate opportunities and branches. For instance, the designer may arrive at a key version that presents multiple opportunities to generate design variations, but it could be tedious to manually revert changes and bookkeep notable variations. Conversely, if the designer could flag a design version as a key point, the design tool may provide,

e.g., graphical interfaces or visualizations to scaffold the exploration. Design tools may even automate explorations from these key nodes (in the decision tree) and report key findings to the user for further decisions or reflections, making the interaction more proactive and provocative. Still, these topics fall out of the current work's scope, but future work may wish to explore these topics to better facilitate computational design thinking.

7.14. Future Work

Besides the design of interconnected compliant mechanisms, their fabrication challenge also bottlenecks their broader impact in the engineering, maker, and HCI fields. The flexures are often slender with extreme aspect ratios that make them prone to deformations during fabrication, leading to imprecision and quality issues in manufacturing processes. While support-free fabrication methods like powder-sintering and embedded printing could offset this challenge, these methods require expensive hardware or proprietary machine setups, making them less appropriable to hobbyist makers and the non-expert public. Alternatively, the manufacturing challenge could also be addressed by computational toolmaking. For instance, flexure layouts could be optimized for planar fabrication [332] or additionally supported by cut-away flexures to avoid vibration and slacking in an additive printing process. Fabricating devices through transformative active materials (e.g., using self-folding structures [205] or 4D printing [313]) may also allow designers to print a device in a more fabricable form factor that transforms to take the functional shape via post-fabrication self-assembly.

Interconnected compliant mechanisms also have the potential to create mechanical computing devices that do not rely on electrical circuits to carry logic and function. Integrating active materials further augments their ability to detect and respond to ambient interactions or environmental factors or even adjust their functions without a digital processor for mediation. This design paradigm has pros and cons that set it apart from conventional electronics IoT. While they may be limited in computation speed and versatility, they could be less expensive and more resilient in hazardous (e.g., outdoor, humid) contexts. In HCI, these devices' potential applications may include

- distributed sensors (energy harvesting and sensing elements in the environment),

- wearable technologies (lightweight and reconfigurable based on use context), and
- input devices (force feedback and functional reconfiguration).

However, these visions may require further research on material selections, prototyping, and functional evaluation to establish viability. Nonetheless, we are confident that the presented algorithm and design tool enable the community to tackle such problems.

7.15. Conclusion

In this work, we proposed an algorithm for designing interconnected compliant mechanisms. These devices are composed of deformable joints capable of transmitting forces and displacements; such assembly can carry out arithmetic computations that determine the device's behavior based on the input condition. The algorithm combines numerical methods, screw algebra, and graph theory to help designers to prescribe device behaviors in the form of input/output transmissions, providing a way to prescribe physical events. Such an approach also allows users to incorporate active materials – materials that sense, actuate, and reconfigure – to make devices more versatile and function-rich. In addition to the algorithm, this work also contributes a computational design tool that helps users find design solutions interactively and procedurally with different levels of control. Our evaluations show a good agreement between transmission design objectives and the resulting device. Several application examples also highlight the enabled design opportunities, including amphibian robots with embedded gait logic and wearable devices that augment disability in unconventional contexts. We believe the proposed algorithm and design tool will empower HCI researchers to navigate new design spaces and provide enriched interactivity through physical interfaces. Envisioning future implications, we also anticipate that our contribution will pave way for embodied computation and interaction to seamlessly integrate into everyday life.

7.16. Computational Toolmaking Remark

We implement the algorithm into a computational tool to help users model and iterate designs in a singular environment. The tool provides different interactivities during the design process – from full automation to manual editing, as well as suggestive design modes that collaborate with designers to modify the design. The tool's diverse design modes provide different benefits to the

user, from the convenience provided by automation to informed direct manipulation in manual or suggestive editing. Designers may freely switch between these modes to adapt to their design task and workflow or respond to emergent design opportunities or considerations that arise from the iterative process. The design tool's versatile workflow and user involvement exemplify the flexible workflow envisioned in the design toolmaking motifs. Still, while the design tool provides basic functions to explore design alternatives, it also highlights the need for more effective or intelligent mechanisms for navigating design history and versions.

Chapter 8. Conclusion

As active materials design emerged as a new physical creative practice, the need for a computer-aided design infrastructure also surfaced. Yet, active materials are novel, exotic, and dynamic, creating design challenges that set them apart from conventional static media. This work addresses these challenges and formulates toolmaking principles that could make active material design tools more versatile, assistive, and appropriable by designers. This thesis advocates a shift from making "tools that solve design problems" to "tools that help designers find solutions." These tools deemphasize automation and focus on collaborating with the user to solve design tasks. By actively involving the user in the design process, the tools also provide opportunities for the user to familiarize themselves with the media (i.e., active materials) and reflect on the design's progression and goals. Incorporating the designer's intelligence into the design process also helps bypass a CAD tool's rigidness and algorithmism, allowing designers and CAD tools to "do what they do best" and arrive at a more satisficing solution.

This thesis explores toolmaking methods and techniques that respond to "helping designers find solutions." To achieve this, each study is contextualized in an active material system to prototype a CAD tool according to the toolmaking principles. Each material system exemplifies a family of active materials design problems (i.e., parametric and combinatorial). The prototyped CAD tools employed different mechanisms to assist the users, including using machine learning to develop fast and accurate simulators to support various workflows, expert systems that help designers to model for targeted functions progressively, and heuristic algorithms and engines that help designs plan out design strategies. Suggestive design tools emerge as an effective method to support users in solving design tasks; they guide the users to iterate designs toward a satisficing solution but allow the users to co-steer the course of design for factors not apprehended by the tool.

This thesis sheds light on toolmaking principles and implementations to augment designers' ability to work with active materials. Both computational agents and human designers are indispensable in designing for complex real-world problems. The challenges of active material design further underscore the value of the partnership between human and CAD tools in working with novel media and applying them to address realistic, contextualized design tasks.

8.1. Contribution

This thesis's primary conceptual contribution is a set of toolmaking motifs for active material CAD tools. The motifs advocate for CAD tools to augment designers' search for design solutions instead of automating the generation of a solution. These tools allow designers and computational design agents to co-steer the course of design. In an active material design process, the tools provide guidance to iterate the design toward algorithmic goals. Meanwhile, designers make informed design changes while simultaneously incorporating non-algorithmic values into a computational design process. Designers may also use these tools to generate and explore alternative design solutions, potentially reflecting and pivoting the design problem through trial-and-error and comparing between satisficing options.

In each chapter, this work employed the CAD toolmaking motifs to develop CAD tools for active material design problems of different natures. The making of these tools is, in themselves, a technical contribution to the HCI and engineering community. These contributions include the pioneering using machine learning to speed up parametric active material simulation and provide support in versatile modalities of interactions. Subsequent projects also developed expert systems that help designers tackle reconfigurable compliant mechanisms and compliant mechanism assemblies integrated with active material components. The algorithms and computational tools developed for these studies also expand the design knowledge in respective HCI and engineering fields.

Finally, this thesis presents research artifacts, including design tools and application demonstrations. The design tools are readily available online at https://github.com/morphing-matter-lab with the hope of democratizing active materials research and design to a broader audience. The application demos exemplify the design opportunities enabled by providing computational aids in specific active material design systems. The thesis also evaluates the design tool's effectiveness and the application demo's effectiveness in their intended use context.

8.2. Limitation

This thesis primarily focuses on the implementation and technical evaluation of computational toolmaking for active materials design. The evaluations are mostly centered on validating the

design tools' effectiveness in addressing algorithmic design goals. However, the usability of these tools and their implications for human-design tool collaboration are yet to be investigated in future research. SimuLearn's follow-up user study anecdotally supports the idea that a real-time responsive dynamics engine could help designers develop an understanding of the media and design rules. However, the study also warned that tools exercising strong agency and falling short at communication might cause designers to feel uneasy about collaborating with them or avoid using inverse or hybrid design functions altogether, especially when the tool appears to know more about the media than the user. While the subsequent CAD tools provided more communication with users – both visual and textual, it is still undetermined if the feedback was effective at communicating the state of design and elaborating on its suggestions, and further studies are required to explore this aspect of human-CAD tool interaction.

Examining the design tools' effectiveness in supporting designers by inspecting the product artifacts may also limit what we could learn about the human-tool collaborative design process. This thesis mainly uses suggestive interfaces to support designers in active materials design. However, the designers' intentions (e.g., divergence and convergence) may also change during the design process. It is unclear if a suggestive mode is sufficient to provide the proper support throughout the process or if it was only helpful in certain scenarios. For instance, a suggestive mode requires the designer to express both the design goal and the design itself, and this act of communication may be unnecessary when the designer has a clear understanding of the satisficing design and its criteria. In this case, either a forward or inverse design tool could be more convenient to use since they only require the designer to express either the goals or the design itself. Several CAD tools presented in this thesis also afford both forward, inverse, and suggestive design modes in the same system. Yet, this thesis did not study how users leverage or compose these modes in the design process. Investigating the transitions between design modes may help toolmakers understand how designers use their tools to support different design phases.

The design tools presented in this work are targeted at designers with some knowledge of the media. In particular, the tools assume that the designers have at least a conceptual understanding of what the material systems could achieve but lack the tacit knowledge to manipulate them. Thus, the tools do not provide an introduction to the materials (e.g., a welcome page). The designers are expected to use the tools to develop a more comprehensive understanding of the material, not to

initiate that understanding. Future research may consider incorporating an introductory curriculum (i.e., a demonstration of material capabilities and parameters) to help completely novice users kick-start learning to work with active materials.

On the other hand, knowledge may also be a factor that affects designer-CAD tool interaction in active materials design, but this thesis did not study its effect nor respond to the differences. A designer who is less familiar with a tool and media may prioritize establishing an understanding with them, and the design tool may better support this need through communications and simulations. Design tools may also decrease the magnitude of design changes between iterations or reduce the complexity of design suggestions to make them easier for novice designers to understand. Conversely, an experienced designer may be more proficient at framing design problems, projecting the qualities of satisficing designs, and predicting the CAD tool's output. In this case, the design tools may increase the magnitude of design changes between iterations to speed up the generation of a solution.

Finally, the computational tools implemented in the thesis are scoped only to help users solve design tasks. Yet, there are other aspects of active materials design that CAD tools could also augment designers to tackle, such as ideation from examples. In SimuLearn's follow-up user study [78], a participant commented in the post-study review that other than the design iteration supports, they also anticipate design tools could provide examples from prior works to help inspire design concepts. The idea of an active material database has also been proposed by literature to aid designers in finding active material design concepts [233]. However, this thesis reckons that while active materials design is garnering increasing interest across design and engineering fields, it has not yet reached a critical mass of design examples to establish a design library.

8.3. Future Work

In addition to visual and textual feedback, multimodal communication, including verbal, gestural, and graphical communications, may further boost collaboration between design tools and CAD users. Compared to keying in their commands, verbal conversation to interact with a design tool may allow designers to express their intentions and needs more naturally. For users with limited

motor proficiency, a verbal interface may also empower them to work with CAD tools without struggling with the keyboard and mouse.

By talking with the user, design tools may also detect designers' intentions and mental processes. For instance, when a user uses phrases that imply uncertainty (e.g., "maybe," "what if," "perhaps"), it might indicate that the designer is trying out an idea. The design tool could then switch to a forward design mode to help users implement the idea, ask them to elicit their goal, and switch to a suggestive or inverse design workflow. However, such conversations require the tool to communicate with the user through natural language. In this case, large language models (LLMs) may lend themselves as a handy mediator between human users and CAD tools in verbal communications.

Gesture is another form of communication designers often employ to explain their intentions, especially when co-working with another human designer. This has also been observed in SimuLearn's follow-up study [78] when we asked participants to work in pairs. Designers often use mouse movements (i.e., circling, pointing) to complement their speech and clarify which part of the design they are addressing. This information is often neglected by design tools and may present an opportunity to better understand the designer's current intentions and goals. Other than mouse gestures, hand gestures may also be an effective way of communicating active material behaviors. For instance, a user can communicate a morphing beam's transformation by curling their fingers. Such tangible interaction may allow designers to reason with spatiotemporal behaviors by tapping into their motor skills [201].

Other than solving design problems, defining algorithmic design objectives may also be a challenging aspect of computational design. Design tools are rigid; they cannot comprehensively cover all objective functions that designers may need when faced with diverse real-world problems. Yet, when designing for real-world problems with active materials, providing more contextualization may be critical in assessing the design's performance and informing further changes. For instance, in SimuLearn's user study [78], designers anticipated optimizing a bottle holder's friction, but such a function falls out of the tool's scope. As a result, design tools could only provide limited – if any - support when designers attempt to attain these qualities in their design.

To overcome this bottleneck, this thesis speculates that large language models may be able to help designers hack into the tools and incorporate new objective functions and optimization mechanisms. As a proof of concept, LLMs have shown promise in synthesizing code from natural language descriptions [38, 57]. Literature [39] also demonstrated using LLMs to generate codes for physical simulation and reason about physical problems based on simulation output. A question then arises: Can we structure active material CAD tools to be modular and use LLMs as a toolmaking agent to customize functions for contextualized problems? If a design tool could understand context-specific objectives and computationally model them for the user, it may allow active material CAD tools to become more versatile and context-aware.

There are also other aspects of computational active material design that this thesis did not cover but may be essential in supporting designers' computational thinking, such as design history navigation and prompting for reflection. The user's design history control (e.g., redo and undo) may provide a way of estimating the key points in the design. For instance, if a designer frequently reverts a work-in-progress design back to a certain version, it may imply that the version is a key point in the solution-finding process. A design tool may then use the key version as a basis to explore design variations and report noteworthy findings to the user to inspire their decisions. Similarly, design tools may ask generative design questions like "do you think this (i.e., the current design) is what you want?"

Lastly, I speculate that combining reinforcement learning and active materials design may lead to a new toolmaking paradigm in this field. Reinforcement learning (RL) is conventionally used in robotics to develop a robot control policy in a simulated environment. At training, the agent is rewarded each time it takes an action that brings the robot closer to the goal, or it is penalized otherwise. An optimal policy is developed by maximizing the agent's expected reward (i.e., taking the best action toward the objective given the current state of the robot). When applied to a design problem, the actions are replaced by legal design changes, and the robot is replaced by the design model. An agent is then trained to complete a family of design tasks by repeated trial and error. Such a strategy has been explored in various design problems, like approximative modeling [156], generating performative drone designs [354, 355], maximizing chemical plant venue [275], producing viable PCB layouts [189], and mechanical design [30, 66, 83, 147].

When applied to active materials design, RL agents may allow us to create design tools that develop their own heuristics to modify a design, thereby reducing the need for expert design rules in toolmaking. Given a computationally modeled active material, a design tool could be acquired by training an RL agent to complete a class of design tasks. The trained model could then select or suggest appropriate actions that would improve the design toward the given objective. In active material CAD toolmaking, this concept implies that as long as a material model or simulator is available, a CAD tool could be developed to help users find design solutions. Still, RL reward shaping could be difficult if the design objective (hence reward function) is discontinuous, which is common in active materials design (e.g., compliant mechanisms' kinematics). The policy of a trained RL agent could also be difficult to interpret. It selects an action but does not explain the rationale behind that decision. Thus, RL's suitability for making collaborative CAD tools remains to be explored in future research. Nonetheless, an explainable RL design agent may shed light on a new era of physical creative processes. The tools may be able to generate design and engineering knowledge for emergent, exotic media.

The dynamics between computational tools and users shifted when developing CAD tools for novel and exotic media. When faced with an unfamiliar design problem, a computational agent may be more capable and knowledgeable (i.e., predicting and optimizing the media's behavior) than its user. Consequently, the user relies more heavily on the agents to handle design tasks. At the same time, computational tools are also becoming more powerful as technologies advance. Speculating into the future, the disparity between a CAD tool and its designer's knowledge may continue to grow, and a computational toolmaking practice for "human-aided design (HAD)" [320] may surface. In this paradigm, the computational agent replaces the designer in coordinating the design collaboration and workflow, and the human designer may play a more assistive role in finding a solution by, e.g., making small, piecemeal adjustments to help meet design requirements. In this case, new research topics may arise, including managing responsibilities, communicating design initiative transactions, and negotiating conflicts. With such a tool, design as an intellectual activity may become more appropriable to common folks without professional training, empowering individuals to innovate. Design tools may even provide a spectrum of interactivities between CAD and HAD to provide diverse ways and workflows of solving design problems, making them more adaptive toward the user's needs and skill levels.

8.4. Closing

The vision of this thesis is to establish CAD toolmaking principles that address active materials' unique challenges and make it easier for designers to harness them. These tools should also empower designers to apply active materials in developing functional products that respond to real-world problems. Yet, designing for the real world with emerging media challenges both the designer's capabilities and the tools that support them. Designers may not know what the materials' capabilities are, where they could be applied, and how to manipulate them given goals and constraints. Compared to instituted creative practices, media, and established design problems, creating with active materials is often experimental and open-ended. Therefore, design tools must employ a different strategy in augmenting their user.

The projects presented in this thesis provide proofs-of-concept for how CAD tools may be designed to better support a computational designer in working with active materials. These tools used different mechanisms to provide interactive workflows in which the human designer and the computational agent work together to find a satisficing design solution. The designers specify numerical and algorithmic design objectives for the tools. The tools then provide guidance for solving the algorithmic aspect of the design problems, leading to steadily improving design performance.

On the other hand, human users make design changes with the tool's counsel while also considering the non-algorithmic factors. By doing so, the designer and the tool may complement each other's strengths and shortcomings to acquire a more satisficing solution that either of them could achieve alone. Users may also branch out design variations under the tool's guidance to focus their exploration within a viable design space. The designers are also provided with the means to co-steer the course of design iterations and incorporate emergent factors (i.e., design constraints and objectives absent in the design specification conveyed to the tool) that arise from the design process. These features allow designers to apply active materials in real-world, contextualized design problems more agilely, where the design goals and qualities of a satisficing solution may be unclear to start with and must be elicited through the design process.

Design tools may also provide functions and interactions that help users better reason about active materials. The challenge of designing with active materials came from its novelty and the

opaqueness of the design tools. The former could be addressed by incorporating simulations that help users preview and explore material affordances. Users may also experiment and develop design strategies by adjusting design parameters and examining their impact on the devices' performance. In SimuLearn, we found that rapid and accurate simulators effectively support designers in developing a mental model of the material through trial and error.

Conversely, the opaqueness of design tools may be explicated by communicating the tool's actions to the user. The tool's inner workings – finding and presenting design updates - should be structured and prioritized for understandability instead of efficiency. Design tools that are aggressive at modifying the design may cause users to become overwhelmed and frustrated when working with an unfamiliar tool and media, leading to adverse effects in supporting the designer. In comparison, while design tools that exercise design in a slower-paced and piecemeal manner could be slower at arriving at a design solution, their actions and logic may be more intelligible to the user and make it easier to collaborate with such tools.

As an emerging physical creative practice, active materials design is still in its formative stage. To help designers better navigate these unfamiliar design landscapes, computational tools must go *Beyond Automation* and become more collaborative with their users to unlock the media's potential. I posit that *CAD tools that help designers find solutions* will better support designers in agilely managing design processes and attaining satisficing outcomes.

Computational toolmakers stand on the verge of scientific frontiers. While they may not be the forerunners of scientific discovery, they are the architects of modern design and manufacturing infrastructure. Using active materials design as an example, this thesis sheds light on toolmaking challenges and principles for emergent media. Gazing forward, I hope that with increased research in human-computer collaboration, future computational tools may become more integral and augmentative to the design and engineering creative practice.

Chapter 9. Appendix

This section contains:

Appendix 1. Rationalizing Compliant Mechanisms

Kinematics in Three-Dimensional Space

In this work, we primarily evaluate and design compliant mechanisms through their motional degree of freedom (DOF) and constraint (DOC). In 3D space, a rigid body's configuration can be defined by six numbers: its position and rotation along the three principal axes (Figure 2A), and a mechanical system is considered to have a kinematic DOF when it is free to translate along or rotate about an axis. Similarly, a system is considered to have a degree of constraint when it is unable to generate motion along or about an axis, and a device's DOF and DOC are complementary: DOF + DOC = 6 motional freedoms in 3D space.

Screw Algebra Representation of Compliant Mechanisms

This section summarizes the algebraic representation and modeling required to understand this document. The mathematical foundation was first introduced by Johnathan et al. [97, 98, 278]. We refer readers to the literature for further details. In the following part, we will use only flexural rods as an example of flexure and skip the other more complex flexure types (blade and notch) to help introduce the concept. The following table provides the notation used throughout this section:

Table 9-1. List of notations used throughout this work.

Scalars	
p	Twist vector pitch
q	Wrench vector pitch
ω	Twist vector angular velocity
υ	Pure translational twist vector velocity
f	Wrench vector force magnitude
n	A compliant mechanism's degree of freedom
m	A compliant mechanism's degree of constraint
D	Rod and motion direction indicator
K	Stiffness value
r	Flexural rod radius
l	Flexural rod length
θ, φ	Angular parameters
Vectors	
$\hat{T}$	6×1 Twist screw vector
$\widehat{W}$	6×1 Wrench screw vector
$\hat{n}$	3×1 Axis direction unit vector
$\hat{c}$	3×1 Twist screw vector reference point on axis coordinates
$\hat{r}$	3×1 Wrench screw vector reference point on axis coordinates
$\hat{0}$	Zero vector
$\widehat{J_f}$	6×1 Joint instantaneous freedom velocity
$\widehat{J_c}$	6×1 Joint instantaneous constraint force
$\hat{x}$	Joint freedom space velocity parameter vector
$\hat{y}$	Joint constraint space force parameter vector
$\|\hat{a}\|_2$	L2 norm of $\hat{a}$

Linear systems	
$[T]$	Freedom subspace
$[W]$	Constraint subspace
$[K]$	Stiffness matrix
$[0]$	Zero matrix
[Ad]	Adjoint transformation matrix
$\mathcal{N}([A])$	The nullity (kernel) of linear space $[A]$
$Rank([A])$	The rank of a linear space $[A]$
Boolean operations	
$\cup$	Union of two sets or linear spaces
$\cap$	Intersection of sets or linear spaces
$\setminus$	Difference between sets or linear spaces
$\bigsqcup$	Consecutive Union (of sets or linear spaces) over a
$\bigcap$	Consecutive Intersection (of sets or linear spaces) over a
Sets	
$\boldsymbol{A}$	(Bold and italic text) A set
$\subset$	Strict subset of
$\subseteq$	Subset of
Logic	
$\leftrightarrow$	Linear system equivalence.
$\Leftrightarrow$	Conditional equivalence
$\rightarrow$	If condition (i.e., left-hand side if the right-hand side is true)

Representing motional freedoms and constraints as screw vectors

Compliant mechanisms' kinematics design through the screw theory has been demonstrated by the literature [96–100, 102, 264]. In the 3D space, the instantaneous motion $\hat{T}$ of a body can be described using a 6×1 screw vector:

$$\hat{T} = \omega[\hat{n} \quad (\hat{c} \times \hat{n}) + p\hat{n}]^T$$

eq. 9-1

Where $\hat{T}$ is also called a twist vector, ω is the motion's angular velocity, and p is the pitch of the motion (i.e., translation along the screw axis per revolution). $\hat{n}$ and $\hat{c}$ are 3×1, vectors that describe the screw axis and a reference point on the axis, respectively. Specifically, $\hat{n}$ should be a unit vector pointing along the motional axis, and $\hat{c}$ is a vector that points from the spatial origin to any point along the axis. For pure rotational motions, p is zero; for pure translational motions, the motion has an infinite pitch, leading to the following form after normalization (i.e., divide by infinity[97]):

$$\hat{T} = \upsilon[\hat{0} \quad \hat{n}]^T$$

eq. 9-2

Where υ is the translational velocity and $\hat{0}$ is a 3×1 zero vector.

Similar to motional freedoms, constraints imposed by flexure elements can also be described using screw vectors. For a rod flexure (i.e., a linear element with circular cross-section), its one degree of constraint load $\widehat{W}$ can be described as

$$\widehat{W} = \mathrm{f}[\hat{n} \quad (\hat{r} \times \hat{n}) + q\hat{n}]^T$$

eq. 9-3

Where $\widehat{W}$ is also called a wrench vector, f is the constraining force magnitude, and the reference point $\hat{c}$ and screw pitch p in (eq. 9-1) are replaced with new notations $\hat{r}$ and q, respectively. Figure 9-1 provides a summary of these notations. In this work, we also refer to the first half of a screw vector as the directional component and the latter as the positional component.

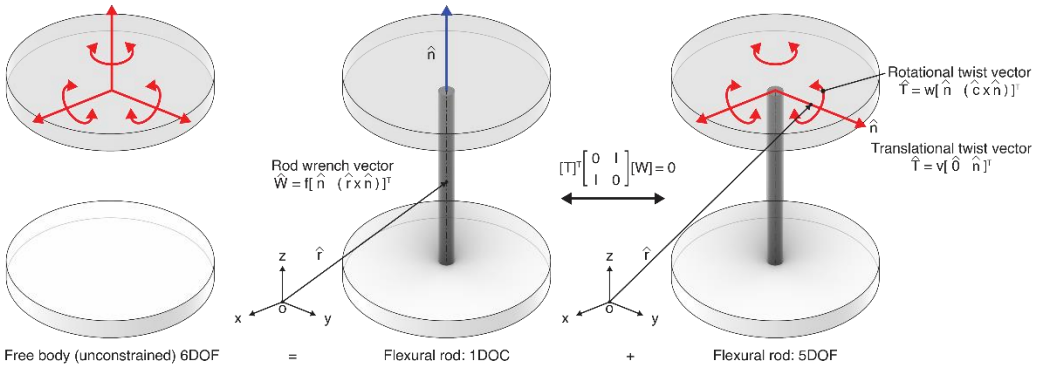

Figure 9-1. Screw algebraic representation of a single flexural rod.

Linear space representation of compliant mechanism joint freedoms and constraints

A rigid body in 3D space may have up to six independent degrees of freedom (i.e., translation and rotation along the principal axes) at the same time. For a compliant mechanical system consisting of two rigid stages connected by parallel flexures, the free end may have $n \in [0, 6]$ DOF with respect to the fixed end, depending on the flexure layout. In particular, the free end can move in any combination of the enabled DOF, and the collection of permissible motions can be represented as a $6 \times n$ linear subspace $[T]$ spanned by the independent DOF:

$$\widehat{J_f} = [T]\hat{x}, \hat{x} = [x_1 \ \dots \ x_n]^T$$

eq. 9-4

Where $\widehat{J_f}$ is the joint's permissible motions, $[T]$ is the freedom space afforded by the joint, and $\hat{x}$ is a $n \times 1$ vector that linearly combines the vectors in $[T]$ to produce $\widehat{J_f}$. Specifically, $[T]$ is a matrix consisting of n independent unit twist vectors, each describing an independent DOF afforded by the joint, and $\hat{x}$ described the velocity of $\widehat{J_f}$ with respect to each DOF in $[T]$.

When designing compliant mechanisms consisting of parallel flexures, given a set of twist vectors representing the desired motional freedoms between the stages, the targeted freedom space can be computed by stacking the desired twist vectors into a $n' \times 6$ ($n' \geq n$) matrix $[T']$, finding its row echelon form, eliminating zero rows, and transposing the matrix. The row echelon form could be found by addition and subtraction of rows to create a matrix where:

1. All non-zero rows are above all-zero rows.

2. Each non-zero row's leading coefficient is 1.
3. The leading 1 in each row is the right of the 1 in the previous row.

Note that the twist vectors in $[T']$ should describe the free stage's motion with respect to the fixed stage. The rank of the resulting matrix equates to the number of independent DOF. Alternatively, the freedom space can also be found by computing the kernel (i.e., a collection of independent vectors in the same dimension that are orthogonal to the input vectors) of its kernel, i.e.,

$$[T] = \mathrm{N}\big(\mathrm{N}([T'])\big)$$

eq. 9-5

The compliant flexures between two rigid stages can also be represented as a constraint space $[W]$ that is derived using the same method as $[T]$, but the twist vectors are replaced by wrench vectors representing the flexural elements. The constraint space's rank m equates to the compliant mechanism's independent degrees of constraints. Thus, a joint's permissible constraint forces $\widehat{f_c}$ can be represented with

$$\widehat{f_c} = [W]\hat{y},\ \hat{y} = [y_1\ \ldots\ y_m]^T$$

eq. 9-6

Where $\hat{y}$ represents the constraining force magnitude as a $m \times 1$ vector.

There exists a mapping between a compliant mechanism's freedom and constraint spaces $[T]$ and $[W]$:

$$[T]^T \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} [W] = [0]$$

eq. 9-7

where the freedom and constraint spaces are correlated by a swap operator [159] consisting of I and 0 as 3×3 identity and zero submatrices, respectively. Thus, given any of $[T]$ or $[W]$, the other can be found by solving eq. 9-7. It is worth noting that following Maxwell's equation, a compliant mechanism's number of DOF and DOC sums to six (Figure 9-1), thus:

$$6 = n + m$$

eq. 9-8

Note that it is assumed that the stages can assume any shape as long as they are sufficiently rigid and do not provide additional DOF to the compliant mechanism. Under this assumption, the kinematics of a compliant mechanism is solely determined by the flexure layout, and the stages have their shapes adapted for any targeted function.

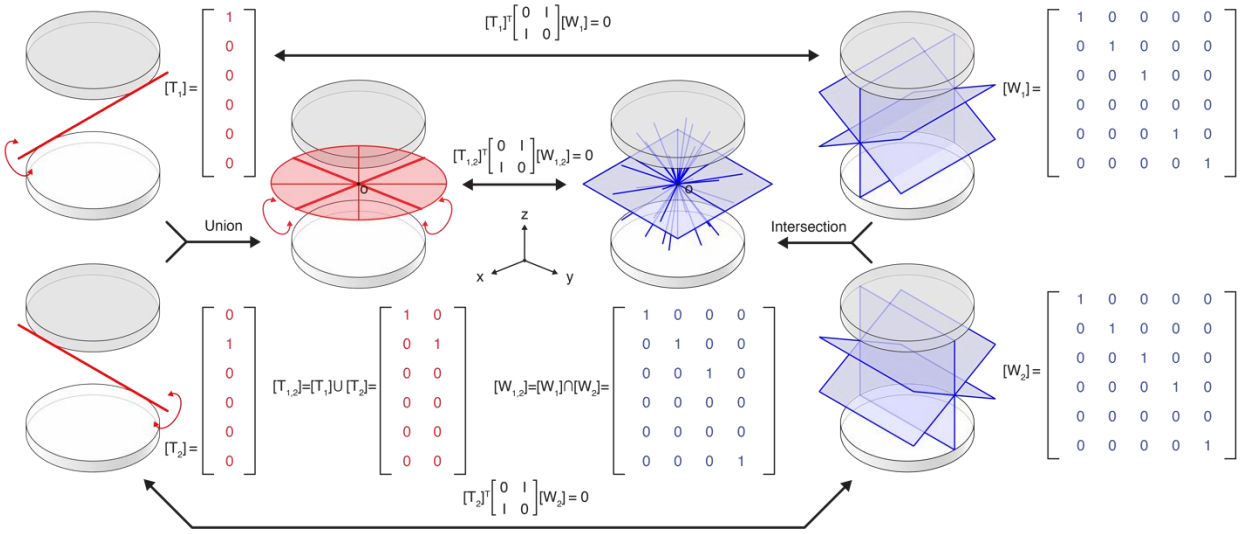

Figure 9-2. Using screw linear subspaces to represent a compliant joint kinematics and flexure design. The screw subspace union and intersection are also shown in this image, using the wrist joint's kinematic freedoms as an example.

Visualization of freedom and constraint spaces

A freedom space describes the direction and position of a joint's permissible motions. Similarly, the constraint space describes the direction and position of constraint lines (i.e., the longitudinal axis marking the center line of a flexural rod). While screw vectors are inherently six-dimensional, they can be broken down into their constituent components for visualization in the 3D space. In particular, the axis direction ($\hat{n}$) and reference point ($\hat{c}$, $\hat{r}$) components can be used to find the placement and orientation of the motional or constraint axis. For twist vectors, if the L2 norm of the directional component equals zero (i.e., $\|\hat{n}\|_2 = 0$), then the motion is a pure translation, and the motional axis can lie anywhere in space. Conversely, if $\|\hat{n}\|_2 \neq 0$, the motion is a pure rotation if $p = 0$ and a screw motion otherwise. A placement is only valid for wrench vectors when $\|\hat{n}\|_2 \neq 0$, since a zero-vector directional component implies a flexural rod without direction.

On the other hand, freedom and constraint spaces $[T]$ and $[W]$ can also be interpreted using an identical method. In this case, the spaces can be regarded as implicit geometries with the velocity vector $\hat{x}$ in eq. 9-4 or force magnitude vector $\hat{y}$ in eq. 9-6 as the parameters. The independent

vectors in $[T]$ and $[W]$ can be linearly combined to produce motional or constraint subspaces (i.e., point, line, plane, or volume), which in turn describes the shape spanned by permissible motional axes for $[T]$ and flexure placements for $[W]$ in the 3D space. Hopkins et al [97]. have provided a comprehensive library of different $[T]$ and $[W]$ combinations. We refer readers to the literature for more details.

Constraint space validity

A constraint space is only valid when it has at least one non-zero directional component in its constituent vectors, or else the constraint space cannot produce a valid placement for flexural rods (i.e., axis-less rods).

Constraint space completion

When designing a compliant mechanism through the freedom and constraint topology[97, 98] (FACT) method, the flexures within the system should be placed to exactly constrain the device's DOF. That is, the space $\left[W_{flx}\right]$ spanned by the flexures' wrench vectors should match that required by the targeted constraint space $[W_{tar}]$ without additional independent wrench vectors in $\left[W_{flx}\right]$. In this case, the following relation should be true:

$$\mathcal{N}([W_{tar}])\left[W_{flx}\right] = [0]$$

eq. 9-9

If eq. 9-9 evaluates to false, the design is over-constrained and may cause the design to have fewer or mismatching degrees of freedom than intended. On the other hand, under-constraining design can also be identified by checking the rank of $\left[W_{flx}\right]$ against $[W_{tar}]$. For an exactly constrained design, the following relation should be true:

$$Rank([W_{tar}]) - Rank\left(\left[W_{flx}\right]\right) = 0$$

eq. 9-10

Alternatively, eq. 9-10 can be replaced by the following condition:

$$\mathcal{N}\left(\left[W_{flx}\right]\right)[W_{tar}] = [0]$$

eq. 9-11

eq. 9-9 and eq. 9-11 are both true if and only if $\left[W_{flx}\right]$ and $[W_{tar}]$ have the same ranks and are equivalent (i.e., spanning the same linear subspace). The mechanism is underconstrained if $\left[W_{flx}\right]$ has a lower rank than $[W_{tar}]$ and overconstrained if $\left[W_{flx}\right]$ has a higher rank than $[W_{tar}]$. If a design is under-constrained, more flexural rods should be added until both eq. 9-9 and eq. 9-11 evaluate to true; if the design is instead over-constrained, the over-constraining flexure(s) should be removed. Over-constraining rods can be identified by checking eq. 9-9, but with individual flexures' wrench vectors replacing $\left[W_{flx}\right]$. The minimum number of flexural rods required to complete a constraint space is the same as the constraint space's rank. However, more rods can be added to achieve targeted device performance (e.g., structural or stiffness demands) so long as they are added within the constraint space. We use the notation $[W_a] \leftrightarrow [W_b]$ when $[W_a]$ is completed by $[W_b]$.

In this work, we manually and iteratively added flexure rods and checked eq. 9-9 and eq. 9-11 to make sure a target constraint space $[W_{tar}]$ is exactly constrained and satisfied. When generating a flexure layout $\left[W_{flx}\right]$, we linearly combine the basis vectors in $[W_{tar}]$ to create flexure wrench screws, which are then decomposed into flexure placement information (i.e., the wrench axis as the orientation and a reference point along the axis's extended line). During the process, if the $\left[W_{flx}\right]$ was underconstrained with respect to $[W_{tar}]$, we identify missing parts (i.e., span vectors) and use them in generating the next flexure placement. Conversely, if a flexure was Over-constraining, we modified its placement to remove the Over-constraining parts of its wrench vector.

Freedom and Constraint Space Boolean Operation.

Three operations are frequently used when designing a reconfigurable kinematic device - subspace union, intersection, and relative complement (also termed difference). The union (Figure 9-2) of two freedom spaces $[T_a]$ and $[T_b]$ is defined as

$$[T_a] \cup [T_b] = \mathcal{N}\left(\mathcal{N}([T_{ab}])\right)$$

eq. 9-12

The two consecutive kernel operators are used to make sure the union subspace is spanned by linearly independent vectors, and $[T_{ab}]$ is the concatenation of the two linear systems along the

first dimension. A union operator is useful in finding the summed freedom or constraint space between multiple kinematic modes. On the other hand, the intersection operator (Figure 9-2) allows us to find the freedom or constraint subspace shared by different kinematic modes:

$$[T_a] \cap [T_b] = \mathcal{N}(\mathcal{N}([T_a])^T \cup \mathcal{N}([T_b])^T)^T$$

eq. 9-13

Finally, the difference operator (Figure 9-3) allows us to find the subspace used by one space but not the other(s). The difference between the two freedom subspaces $[T_a]$ and $[T_b]$ (from $[T_b]$ to $[T_a]$) is defined as

$$[T_a] \backslash [T_b] = \mathcal{N}(\mathcal{N}([T_a])^T \cup [T_b])^T$$

eq. 9-14

Additionally, when comparing two subspaces, one space may be fully spanned by the other. In that case, we define

$$[T_a] \subseteq [T_b] \Leftrightarrow \mathcal{N}([T_a])[T_b] = [0]$$

eq. 9-15

When $[T_a]$ is included by $[T_b]$ (i.e., $[T_a]$ is a subset of $[T_b]$). The same notation is also used for screw vectors. Like freedom spaces, constraint spaces can also be computed with the same schema to find their intersections, unions, and differences, but $[T]$ is replaced by $[W]$.

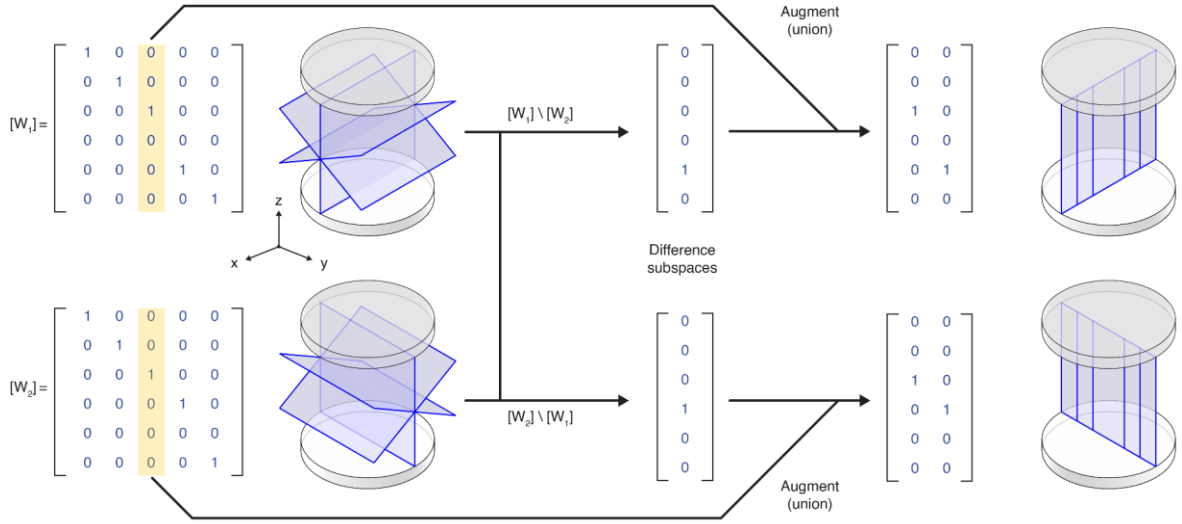

Figure 9-3. Constraint subspace Boolean differences. This example shows the derivation of the wrist joint (Figure 6-2) constraint subspaces. The operation produced unviable constraint subspaces, which were then supplemented to make them viable.

Appendix 2. Supplementary Notes for Compliant Meta-structure Reconfigurable at Six Degrees of Freedoms

Fabrication and Control

Fabrication of Stiffness-Changing Flexural Rods.

The stiffness-changing flexural rods are prepared by casting in customized jigs (Figure 9-4). The jig is machined from a 6061 aluminum stock (McMaster-Carr) and has hemicylindrical grooves, alignment features, and securing screws and nuts for lining up heating wires (34-gauge 316L stainless steel wire, Master Wire Supply) at the center of the rods. To cast the rods, silicone tubes (2 mm ID * 3 mm OD or 1.5 mm ID * 3 mm OD, uxcell) are cut into desired lengths, placed on the grooves, and taped (Kapton masking tape, McMaster-Carr) to the jig to allow for threading heating wires through their center. The alignment features on both ends of the grooves are used to ensure the heating wires are positioned at the center of the silicone tubes. Once the heating wires are tightened and straightened, the screws and nuts on the far ends are used to secure them throughout the fabrication process.

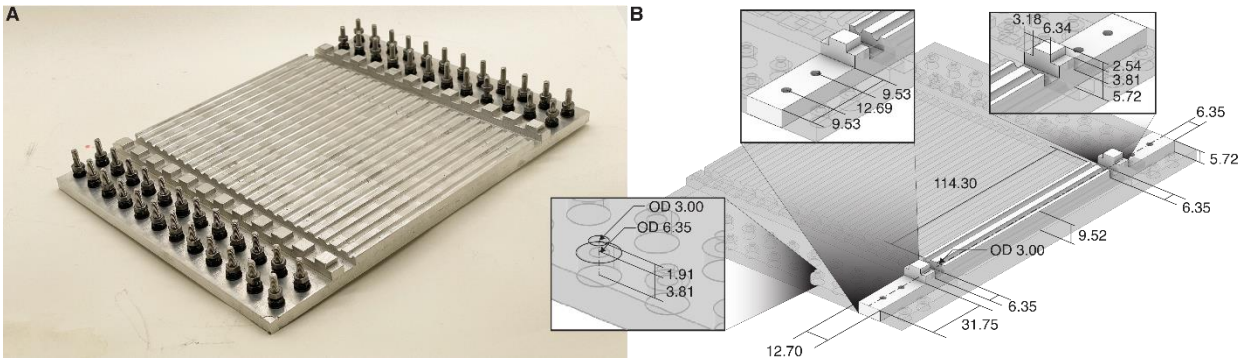

Figure 9-4. Stiffness-changing flexural rod casting jig design. (A) The machined jig prior to casting flexural rods. (B) Schematic diagram of a single unit on the jig. Unit, mm.

To prepare the casting resin solution, Bisphenol A Epichlorohydrin-based epoxy resin (Hexion EPON resin 828, monomer) and the curing agent (Epikure 3380, cross-linker) are combined at a weight ratio of 10:4, respectively. The resin is mixed and degassed for three minutes each using a

planetary centrifugal mixer (Thinky AR-100) to derive a uniform, bubbleless solution. The mixed resin is then loaded into a syringe with an 18- or 22-gauge dispensing tip and injected into the silicone tubes on the aluminum jig (Figure 9-5). Once cast, the resin is left to gel at room temperature (25°C) for twenty-four hours, followed by thermal curing at 100°C in an oven (725F, Thermo Fisher Scientific) for five hours to crosslink fully. Cured resin rods, along with the jig, are then removed from the oven and left to cool down to room temperature. The rods sheathed in silicone tubes are then released from the aluminum jig by removing the tape and loosening the nuts that secure the heating wires. Finally, the rods are unsheathed by slicing and peeling off the silicone tubes.

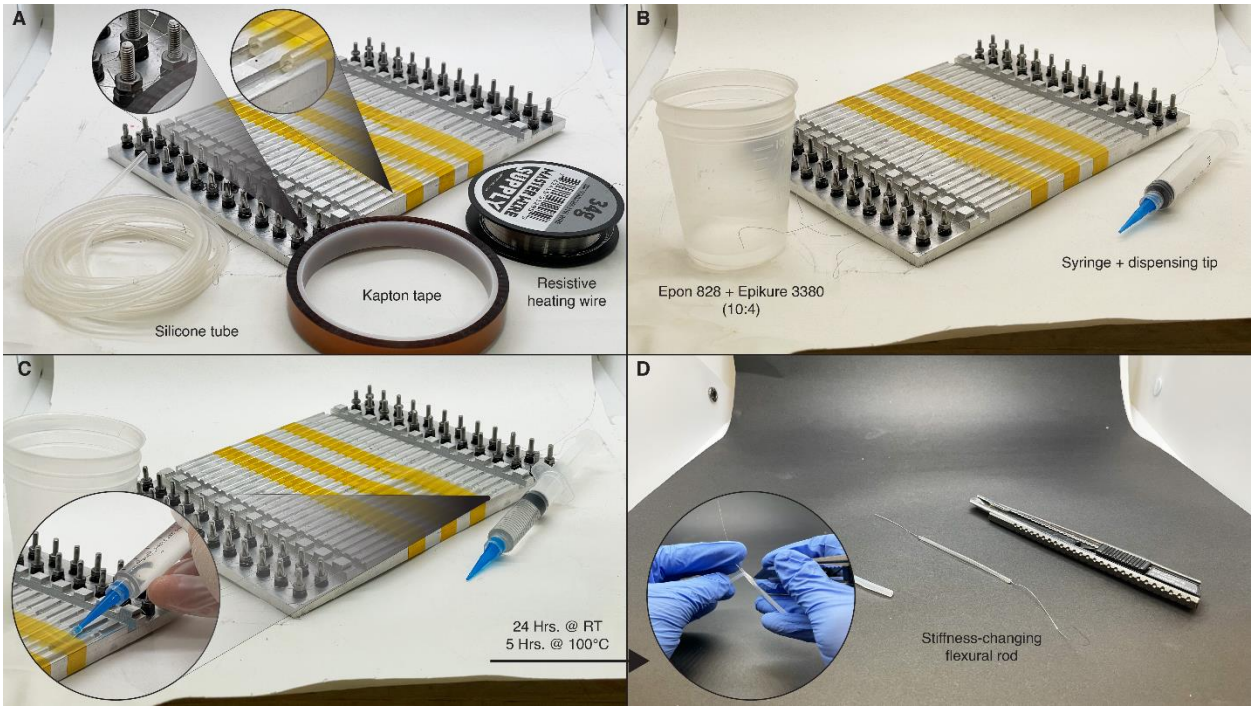

Figure 9-5. The fabrication process of stiffness-changing flexural rods. (A) Adding tubes and securing resistive heating wires to the jig. Inset images show the wire and tube alignment and the screws securing the heating wires on both ends. (B) Mixing epoxy solution for injection into the molds. (C) Injecting epoxy into the tubes. Inset image, injecting the solution from the end of tubes using a 5 ml syringe and a 22-gauge industrial dispensing tip. (D) Releasing the stiffness-changing rods from the tubes. The inset image shows a knife slicing the tube open, which was then peeled off from the rod.

Fabrication of Rigid Stages, Testing Jigs, and Passive Flexures.

The rigid stages and testing jigs used and demonstrated in this work are designed in Rhinoceros 3D version 7 and exported as STL files for fabrication. The parts are made with either a fuse-deposition modeling (FDM) (Ultimaker S5) or a stereolithography (SLS) 3D printer (Formlabs 3B).

White polylactic acid (PLA) filaments and 0.4 mm extruder nozzles are purchased from Ultimaker and are used to fabricate the 6-DOF devices' rigid stages and the fixture for mechanical tests. In the printing processor (Ultimaker Cura), the parts are sliced with default settings at a layered height of 0.1 mm and a 60% gyroid infill for increased mechanical strength. Both adhesion plates and structural supports are enabled to ensure print quality. The passive flexures used in the wearable device are also printed with PLA but with a 0.25 mm nozzle for higher resolution. The flexures are sliced at a 0.1 mm layer height with a 100% infill rate to produce solid objects.

All other parts are made with the SLS printer using the Formlabs White resin V4 with default settings with a layer height of 0.1 mm. SLS printed parts are washed with isopropyl alcohol to remove resin residue and flood-UV cured at 60°C for 60 minutes using Formlabs' FormCure post-curing machine.

Assembling Stages with Flexural Rods.

The rigid stages are designed with circular through holes for housing the rods (Figure 9-6). The rods are inserted into their designated holes to assemble them to the stages, followed by applying cyanoacrylate adhesives (Scotch-Weld, 3M) at the interface. The adhesive is then left to dry at room temperature for twenty-four hours. Finally, the heating wires are connected to conductive wires by ferrules (Copper rivets 0.4 mm, Voltera) 3 mm away from the end of the rods where the heating wires are exposed.

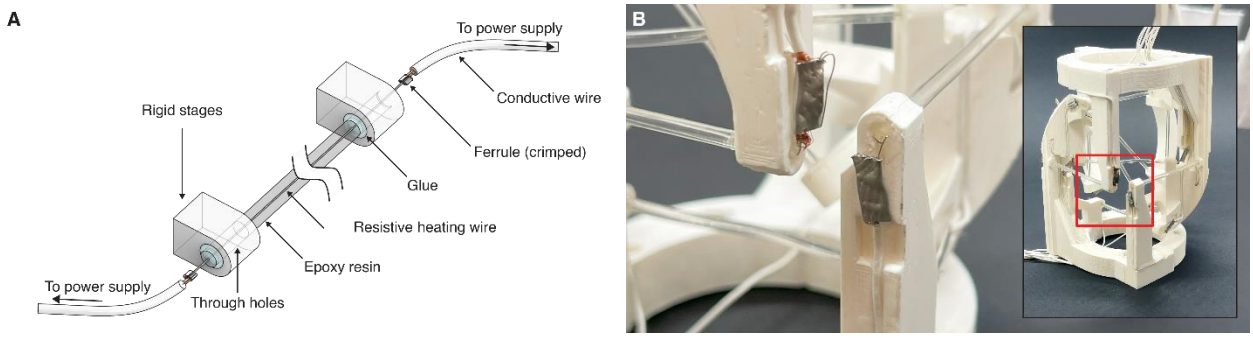

Figure 9-6. Structure assembly and connection. (A) A schematic diagram of the connection between the flexures and rigid stages, as well as electrical connections. (B) A picture of the assembled 6-DOF device showing the connections.

Heating Control

To heat a stiffness-changing flexural rod above its glass transition temperature, we connect the two ends of the conductive wires to a power supply (SPS 3010, Kungber) and provide a current of

0.4 A to the resistive heating wire (rated resistance: 3.68 x 10^{-2} Ω/mm), which corresponds to 5.88 x 10^{-3} watts/mm. The voltage is adjusted depending on the length of a rod and scales linearly with its length. It takes 30 seconds to heat a rod from ambient condition (25°C) to above their glass-transition temperature (T_g, 54°C), yet in our experiments (see following sections), we allow the rods to heat for three minutes to reach quasi-thermal equilibrium. On the other hand, to cool down a rod below its T_g, we cut off the current and allow it to passively dissipate heat in ambient conditions for three minutes without extraneous loads. Any external load may cause the rod to retain in a deformed state as it cools down, which may unintentionally alter the device's kinematics freedoms.

Multiple rods can be simultaneously heated by serially connecting their heating wires end-to-end (Figure 9-7). We manually connect and disconnect the wires to alter the circuit and switch rods between their cold and hot states to reconfigure kinematic modes. While not implemented in this work, future iterations may consider using relays and transistors to afford digital control and reconfiguration of the devices.

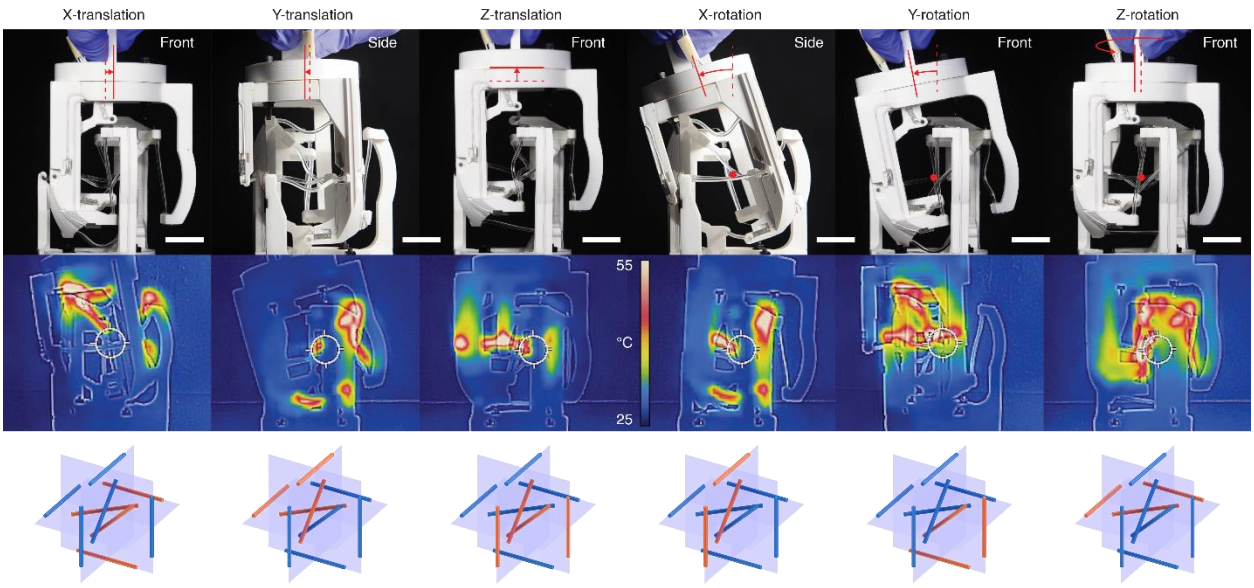

Figure 9-7. Thermal images of the 6-DOF device under different mode configurations. Top row, picture of the device; middle row, thermal images; bottom row, rod configuration (matching Figure 6-3C). Blue rods are configured cold, while rods in orange/red are heated.

Notes on Material Selection and Safety

In this work, we use epoxy to make stiffness-changing flexures for its availability, customizability, and ease of control. In particular, epoxy is a common material for making stiffness-changing

components in robotic systems [31]. Their glass transition temperature could be tuned by altering the ratio between the crosslinker and monomer during its synthesis [357]. The material could also be conveniently made into different shapes [357] by casting, printing, or laser cutting, thus providing high customizability.

The monomer-to-crosslinker ratio was selected to produce a suitable glass transition temperature. In particular, a higher monomer crosslinker fraction will produce flexures with a higher T_g as the polymer chains require more energy to recoil. In contrast, a lower T_g can be obtained by reducing the crosslinker fraction. A T_g of 54℃ was chosen because it is relatively close to the body temperature but sufficiently high to be insensitive to ambient heat fluctuations (e.g., body heat, warm water, in a wearable context). Moreover, a crosslinker that leads to a lower T_g would also compromise the flexure's mechanical strength due to reduced crosslinking density [357]. Regarding safety, we note that the literature [328] had reported that the skin's exposure tolerance to heat is a function of the temperature, and the safe time against a 54℃ heat source is 14.04 seconds without incurring strong thermal injury [328]. Still, extended exposure may lead to first or second-degree burns, and insulation or protection is required [115]. In this work, we ensure safety by placing the flexure rods away from the skin to avoid collision and covering the skin with fabrics.

Material Characterization and Mechanical Experiment

Dynamic Mechanical Analysis (DMA)

The glass transition temperature (Tg) of the sample (cross-linked epoxy resin with a ratio of 10:4) was determined with a Dynamic Mechanical Analyser (DMA, RSA–G2 Deta; TA Instruments). Samples were heated from room temperature to 80 ℃ and cooled to 30℃. The sample dimension used for the DMA test was 12.42 x 8.63 x 0.85 mm^3. The observed glass transition temperature (Tg) from the test was 54.0 ℃ (Figure 9-8), where tan δ peaked at 0.571.

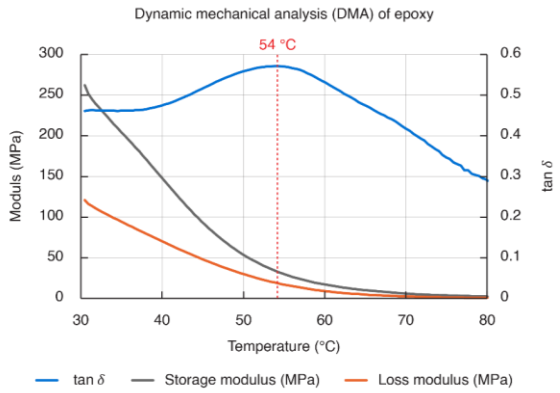

Figure 9-8. DMA test result for the epoxy sample.

Material Characterization

Dog bone (ASTM D412) samples were cast with a mixing ratio of 10:4 (monomer: crosslinker) and characterized using mechanical tensile tests under room temperature (RT) and 54℃ (Figure 9-9). The dogbone samples were heated using hot air from both sides, and their temperature was monitored using a FLIR camera. The strain rate was 10 mm/min. The observed young modulus at room temperature and 54℃ were 1135.4 ± 180.7 and 20.86 ± 7.81 MPa, respectively. The strain rate at failure was 11.84 ± 3.94 and 31.75 ± 1.72 % for the RT and 54℃ samples, respectively. The Young's modulus and elongation at break changed 54.4 times and 0.37 times for the RT and 54℃ samples, respectively.

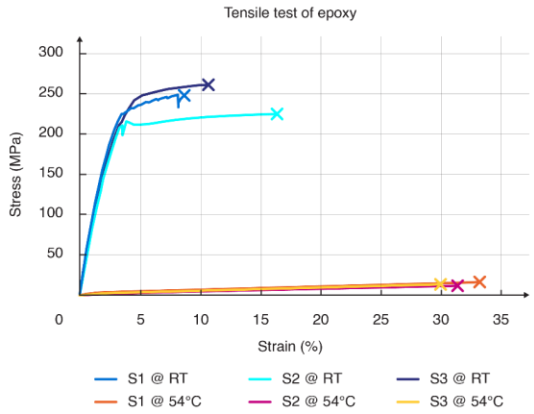

Figure 9-9. Mechanical properties of Epoxy under room temperature and 54°C.

Flexural Rod Characterization

The flexural rod with heating wire (50 mm length) was subjected to compression at 1% strain (0.5 mm displacement) at a strain rate of 5 mm/min at RT and 54 °C. Both flexure rods with OD 1.5 and 2.0 mm were tested and modeled separately (Figure 9-10). The equivalent modulus of elasticity was calculated at 0.1% strain by dividing the load by strain. For rods of OD 2 mm, the equivalent modulus was observed to be 2958.37 ± 50.19 MPa and 152.02 ± 15.74 MPa for the cold and heated states, respectively. For rods of OD 1.5 mm, the equivalent modulus was observed to be 2668.87 ± 1370.85 MPa and 480.85 ± 179.203 MPa for the cold and heated states, respectively.

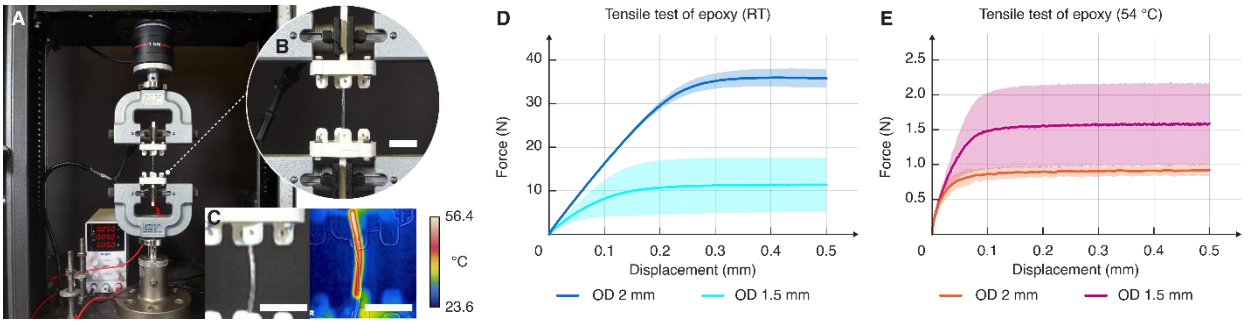

Figure 9-10. Flexural rod characterization test setup and results. (A) The compression test setup. (B) A zoomed-in image of the sample flexural rod before the test. Scale bar, 20 mm. (C) The heated flexural rod at the maximum load viewed from the camera. The image on the right shows the corresponding thermal image. Scale bar, 20 mm. (D) The cold flexural rod load curves. Data are means ± s.d. n = 3 samples. (E) The heated flexural rod load curves. Data are means ± s.d. n = 3 samples.

Mechanical Tests Protocol

The flexural rods and devices were mechanically tested using a Universal Testing Machine (UTM, Instron 5969) equipped with a 1kN Load cell. All tests were repeated for multiple cycles, and the second load cycles were used in the reported plots unless otherwise specified. The 50 mm long

stiffness-changing flexural rods were subjected to a compressive strain of 1% (0.5 mm, displacement) at 5 mm/min for five cycles at cold and hot states.

Three 6-DOF devices were fabricated and tested for translations along and rotations (bending) about the X, Y, and Z axis in the locked and unlocked states (Figure 9-11). The loads were applied at 10 mm/min for five cycles. For translational motions, the device (Figure 9-11A-B) was subjected to a displacement of 1 mm (1% of the device's total length of 100 mm). Since it was difficult to apply the load as a pure toque for XY rotations, we mounted the devices as cantilever beams and subjected lateral loads (2 mm, 2% with respect to device length) to their free end to report their deflection (Figure 9-11C). A displacement load of 2 mm (2% with respect to device length) accounted for the compounded displacement (i.e., the free end is allowed to translate and rotate simultaneously). For Z rotations, the load was applied through a loading arm 50 mm away from the rotation axis as a pure torque (Figure 9-11D). The displacement load corresponded to 2.29° at a load rate of 11.46°/min.

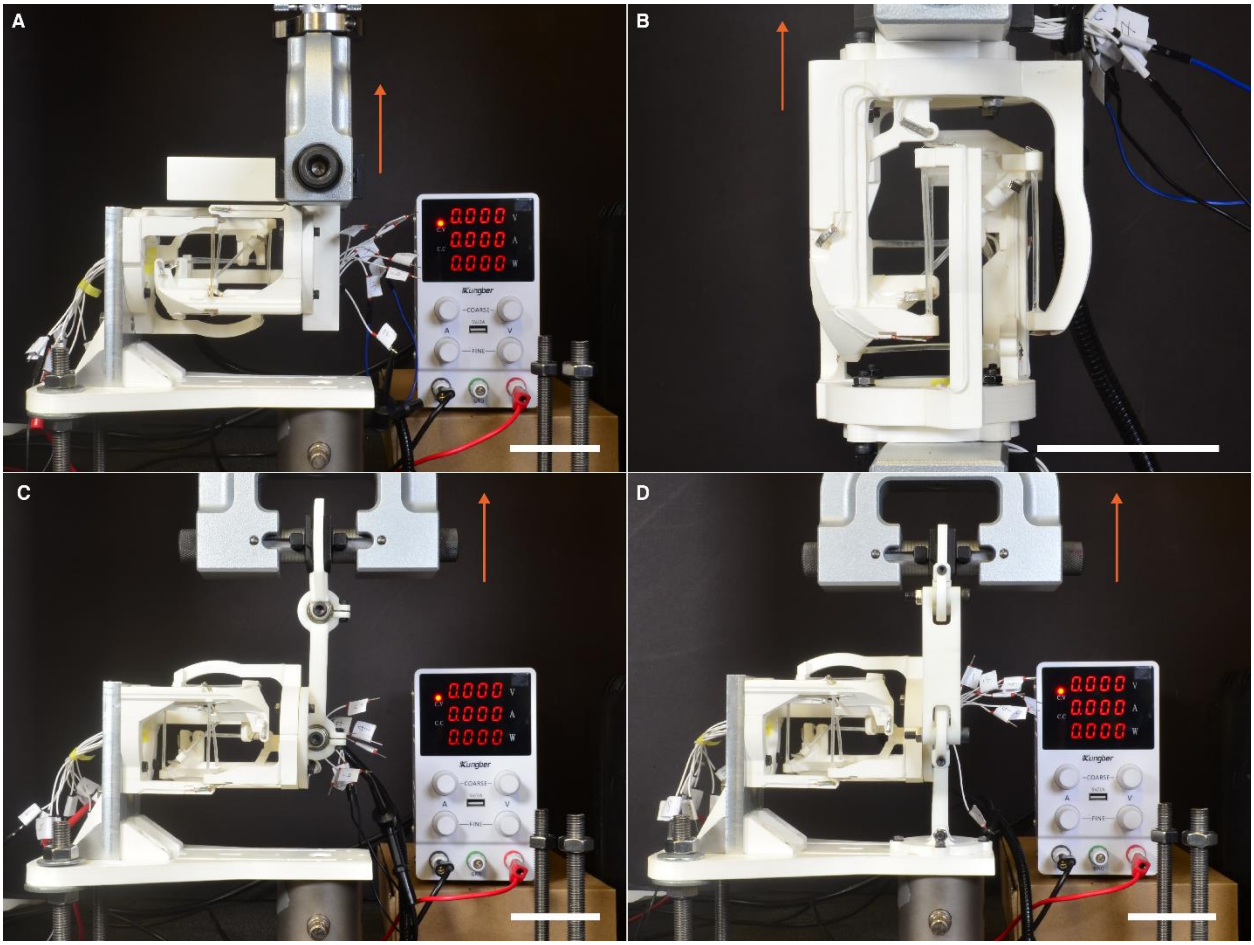

Figure 9-11. Mechanical test setup of the 6-DOF device. (A) The test setup for the x- and y-translation-enabled modes. (B) The test setup for the z-translation-enabled mode. (C) The test setup for the x- and y-rotation-enabled modes. (D) The test setup for the z-rotation-enabled mode. Scale bar, 50 mm.

A wrist device was fabricated and tested three times under each mode along the two DOF (Figure 9-12). The loads were applied through a loading arc of radius 114.3 mm at a rate of 10 mm/min, corresponding to 5.01°/min. The displacement loads were determined based on the tested rotation axis and the device configuration mode. A displacement load of 20 mm (10.03°) was applied to the unlocked states. Conversely, the displacement load was lowered to 5 mm (2.50°) under the locked states to avoid damaging the device.

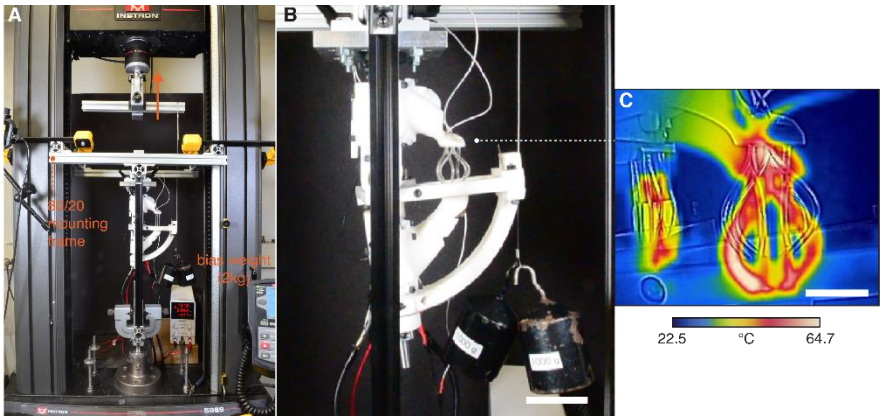

Figure 9-12. Mechanical testing of the wrist joint isolated from the arm-wearable device. (A) The setup overview. (B) The device at 20 mm displacement. Scale bar, 50 mm. (C) A thermal image of the device at 20 mm displacement. Scale bar, 20 mm.

Three thimble devices were tested for three cycles under each reconfiguration listed in Figure 6-5C. The loads were applied (Figure 9-13) at 10 mm/min. The displacement loads varied between rod configuration modes. For configuration [0, 0, 0, 0, 0] where all rods remained stiff, a displacement load of 0.6 mm was used, which corresponded to a 3% system strain with respect to the bounding volume of the flexures (20 mm^3). Alternatively, the partially unlocked configuration [0, 0, 0, 0, 1] was subjected to a load of 3 mm (15% strain), whereas the rest of the modes were loaded with the designed maximum displacement of 5 mm (25% strain).

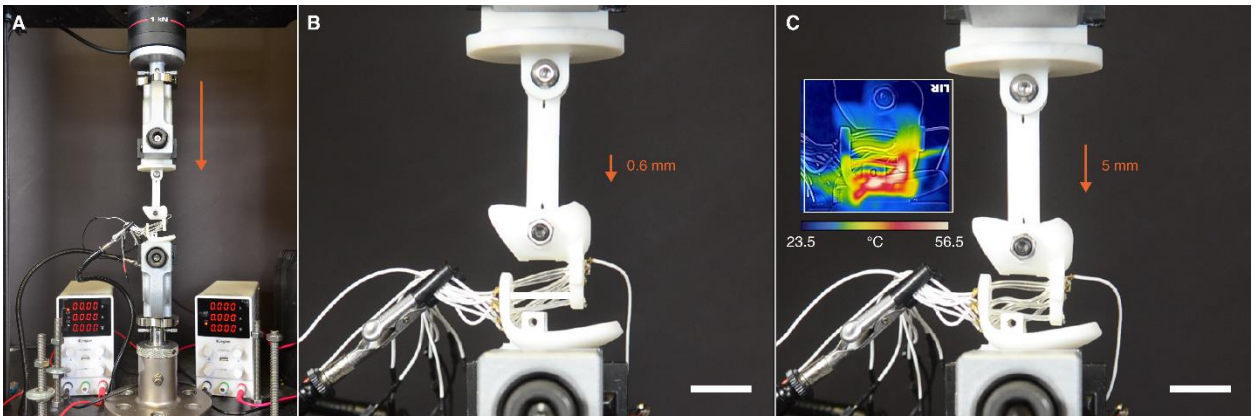

Figure 9-13. Mechanical test setup of the haptic thimble device. (A) The setup overview. (B) The device in the [0, 0, 0, 0, 0] configuration with a compressive displacement of 0.6 mm. (C) The device in the [1, 1, 1, 1, 1] configuration with a compressive displacement of 5 mm. Scale bar, 20 mm.

Mechanical Test Jig Design

The 6-DOF device was tested using customized jigs attached to the Instron machine for applying loads along the appropriate axes. The jigs contained two parts: a fixture that attached the device's

fixed end to the Instron machine's fixed base (Figure 9-14A) and a loading mechanism that applied loads. The jigs were 3D printed using either PLA or UV curable resin except for the mounting plate made with machined 6061 aluminum to minimize deflections when loading the devices (Figure 9-14B). To test translations along the X and Y-axis, a rectangular plate was bolted to the device's free end and clamped to the load cell (Figure 9-14D). Rotations about the X and Y-axis were tested similarly, but rotational bearings and arms were added to the loading mechanism to accommodate the rotation (Figure 9-14E). Translation along the Z-axis was tested by attaching and clamping on 3D-printed plates bolted to both ends of the device (Figure 9-14C). Finally, rotation about the Z-axis was tested with the pure rotational moment. The device's free end was connected to the base fixture via a rotational bearing coaxial to the Z-axis. This design constrained all motions except for rotation resulting from the torque applied through a loading arm 50 mm away from the axis (Figure 9-14F). Similar to the X and Y-axis rotations, additional bearings and arms were added to the loading mechanism to accommodate the device's rotation.

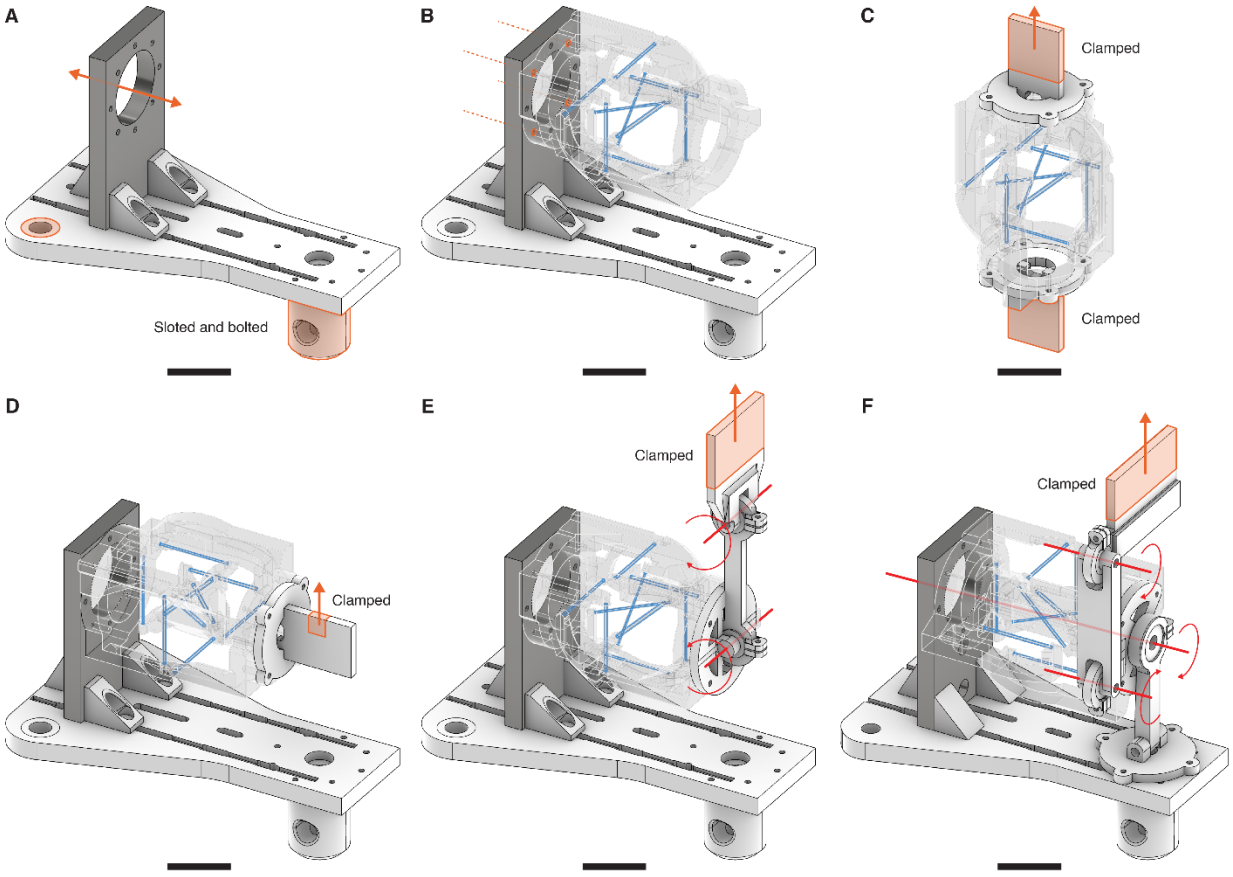

Figure 9-14. The 6-DOF device test jig design. (A) The mounting plate and fixtures used to secure the device to the base of the intron machine. The mounting plate's position can be adjusted along the annotated axis. (B) The device is bolted to the

mounting plate for tests. (C) The z-translation test jig design. (D) The x-translation test jig design. Y-translation was tested using the same design, and the device was reoriented. (E) The x-rotation test jig design. Y-rotation was tested using the same design, and the device was reoriented. (F) The z-rotation test jig design. Scale bar, 50 mm.

The wrist device was mounted on an articulated testing jig to simulate the human skeleton (Figure 9-15A). A U joint connected two stainless steel rods to simulate the two rotational DOF at the human wrist joint. The two stages were mounted on stainless steel rods with interfacing adaptors made with the Formlabs UV-curable resin (Figure 9-15B). The articulated device and jig were then mounted to a machined aluminum stock and frame with flanged mounts and bolts (Figure 9-15C). Secondly, the aluminum frame was clamped to the Instron machine to minimize shakes and undesired displacements through the test (Figure 9-15A).

The free end had been added with loading arcs centered on the flexion and deviation rotation axes for applying displacement loads (Figure 9-15D). The free end was connected to the load cell with a stainless steel cable, and a bias weight of 2 kg was added to the arc in the opposite direction to ensure the cable remained straightened throughout the test. To test the device along a rotational DOF, an additional bearing was added in parallel to the U-joint to constrain the device to displace about the axis of interest while eliminating unwanted displacements (Figure 9-16, Figure 9-17).

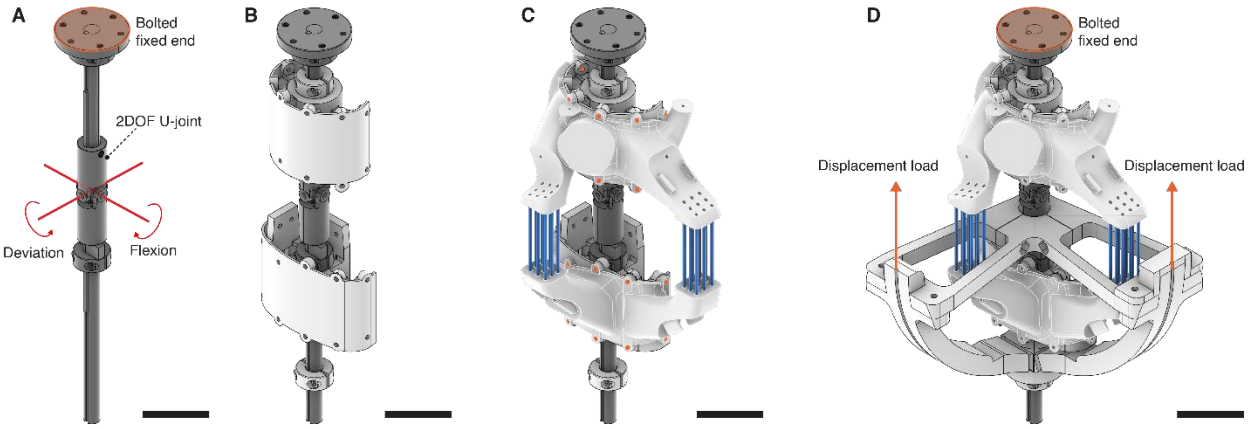

Figure 9-15. The isolated wrist joint test jig design. (A) the articulation assembled using metal parts. (B) The 3D printed adapter parts to interface the jig with the device. (C) The jig with the device mounted and secured through the bolt holes marked in orange. (D) The loading arcs used to apply weight. Scale bar, 50 mm.

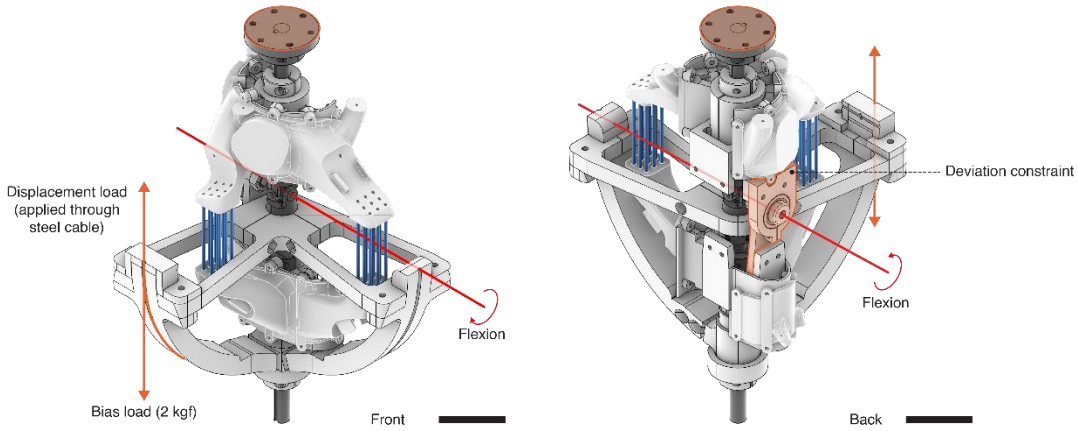

Figure 9-16. Wrist joint test jig configuration for the flexion DOF. Scale bar, 50 mm.

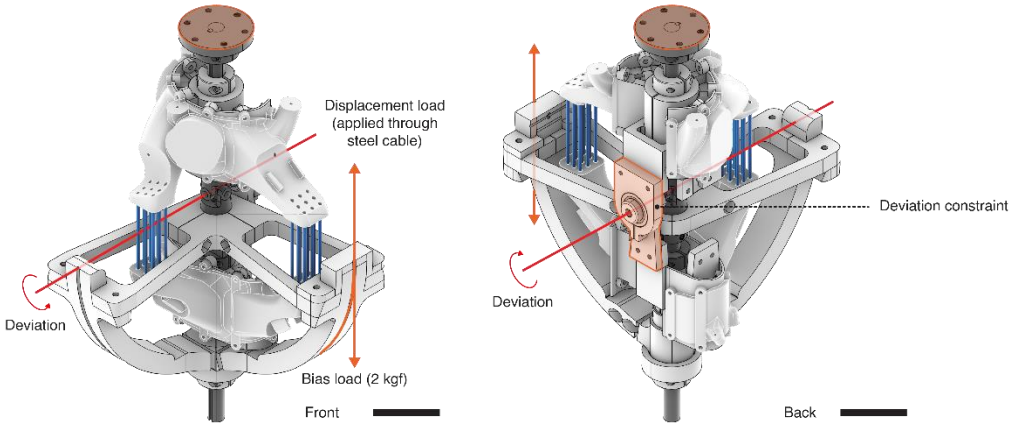

Figure 9-17. Wrist joint test jig configuration for the deviation DOF. Scale bar, 50 mm.

The thimble devices were modeled and fabricated with additional mechanical features on the two stages for mechanical tests (Figure 9-18). The bottom stage was printed with a rectangular plate for clamping by the Instron machine on the fixed end. The displacement loads were applied to the free end at the fingertip contact area to simulate the designed use case. Two bearings and a linkage were added between the free end and the clamp to accommodate rotations during the load cycles.

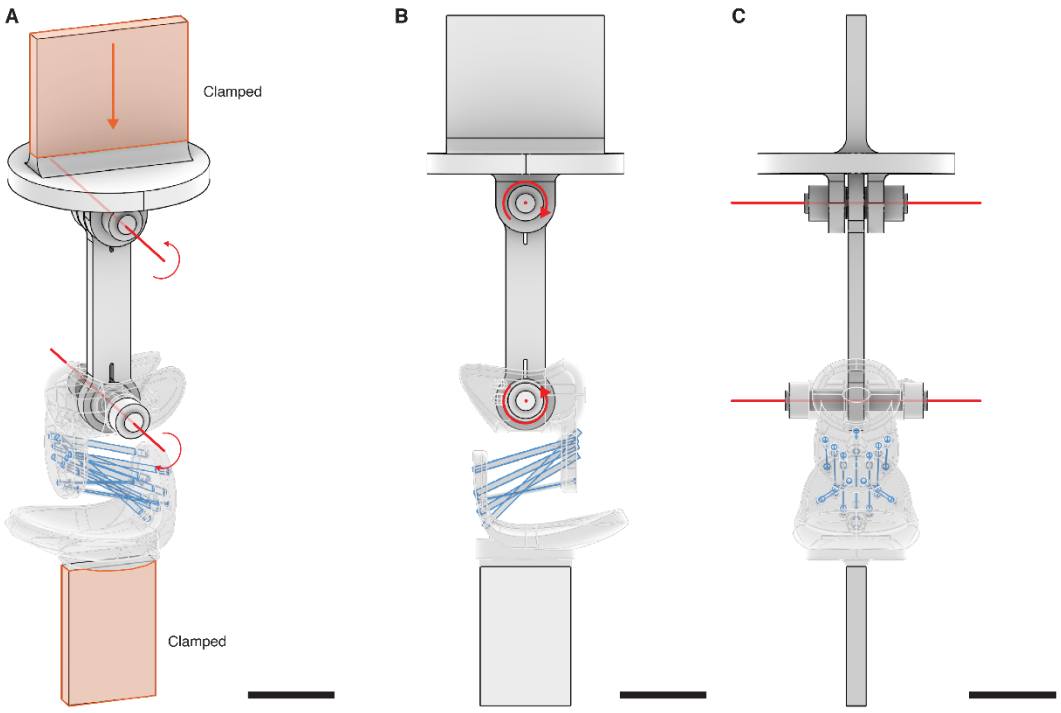

Figure 9-18. The haptic thimble test jig design. (A) The test jig design and loading mechanism articulation to accommodate device displacements in the other directions. (B) The device and jig, viewed at the front. (C) The device and jig, viewed at the side. Scale bar, 20 mm.

Device Repeatability

We conducted an additional repeatability test on the thimble device (Figure 9-19), revealing that the device consistently performed over 100 loading-unloading cycles. We performed the cyclical testing for three modes: fully locked [0, 0, 0, 0, 0], partially unlocked [0, 1, 0, 0, 1], and fully unlocked [1, 1, 1, 1, 1]. For the fully locked state, the observed force remained constant at 30 N over 100 cycles (50 Nmm^{-1} in stiffness). Similarly, the forces remained constant at 1.1 N over 100 cycles when fully unlocked. The forces for the partially locked mode remained constant (20 N) for around 80 cycles and further decreased to 16 N for the remainder. We speculate that a potential reason for the decrease in force/stiffness is due to heat accumulation and diffusion from hot to cold flexures over time.

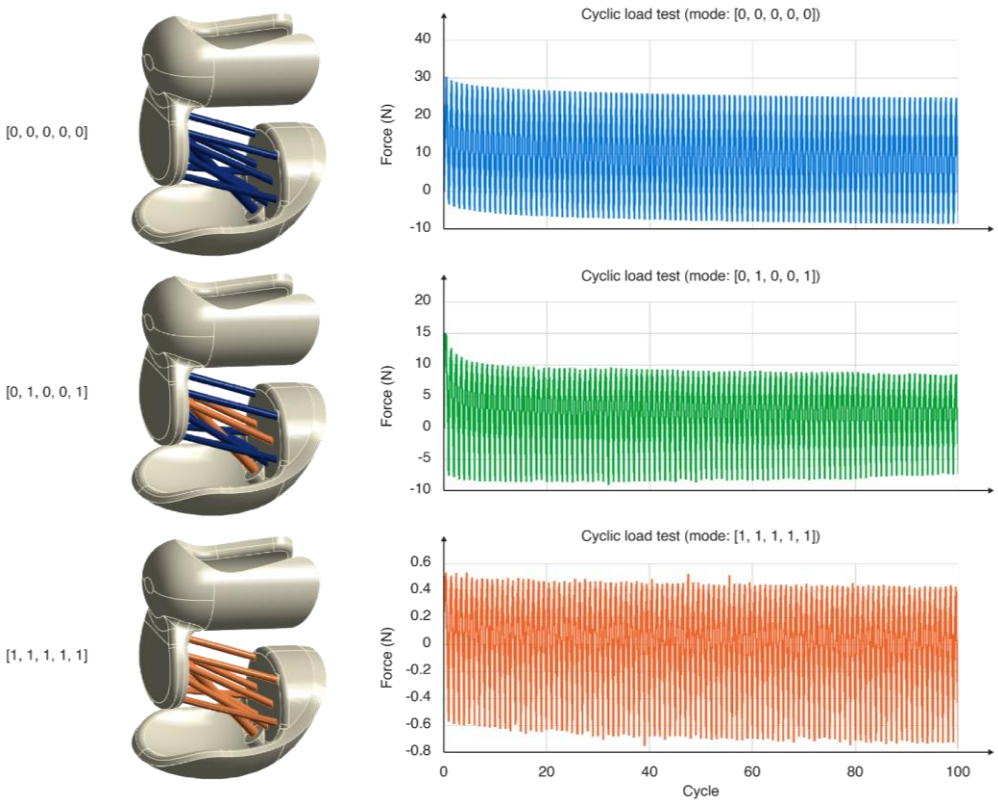

Figure 9-19. The haptic thimble repeatability test over 100 loading cycles.

Literature Benchmark

The literature comparison and benchmark figure (Figure 6-1B) was composed by analyzing the data and devices presented in prior research pertaining to stiffness-changing devices. For each work, the max programmable DOF was determined based on the most complex (i.e., possessing the most reconfigurable DOF) device that had been demonstrated. However, if a generalized design pipeline was presented (e.g., ReCompFig [335]), we then reported the highest number of achievable DOF using the pipeline. The stiffnesses under the softened and hardened modes were calculated at 2% displacement load with respect to device height. We used source data from mechanical load tests when it was available. However, in the case that the source data was not attainable, we used image processing and analysis software (ImageJ) to read out the values from load curves presented in the data figures. Additionally, if a load curve was presented in different units (e.g., Stress over displacement or load over strain), we converted the measurements by factoring in the device dimensions to generate a systematic, effective stiffness readout in Nmm^{-1}.

On the other hand, the stiffness requirements for kinesthetic wearable device design could vary depending on the targeted joint, which has varying isometric strengths and just noticeable displacements. We use several extreme cases to illustrate the range of interest in a kinesthetic design context. The lower bound of 0.013 Nmm^{-1} was determined based on the stiffness JND of the index finger, which was one of the most sensitive receptors on the body and frequently used in haptic explorations [360]. On the other hand, the upper bound of 59.342 Nmm^{-1} was calculated by superimposing a wearable device design context to the data provided by Gupta et al. [89] and Ramos et al. [236]. The elbow joint has an isometric strength[89] of 72.5Nm and a rotational proprioceptive JND [236] of 7 degrees. Assuming a device is placed 100 mm away from the elbow joint rotation axis and should resist the isometric strength below or at the JND displacement, then the device's stiffness should be $\frac{72.5\times1000}{\left(7\times\frac{\pi}{180}\right)\times100^2} = 59.342$ Nmm^{-1}.

Table 9-2. Literature benchmark

Literature	Max programmable DOF	1% displacement stiffness (softened, Nmm^{-1})	1% displacement stiffness (hardened, Nmm^{-1})
Yang et al. (ReCompFig) [335]	5	0.970	10.204
Zappetti et al. [345]	2	0.577	9.813
Mueller et al. [197]	1	2.356	235.573
Liu et al. [160]	1	1.526	58.470
Mekaouche et al. [183]	1	0.196	0.247
Chen et al. [40]	1	0.020	0.383
Mintchev et al. [188]	3	0.103	2.410
Kuppens et al. [140]	1	0.305	9.631
Gao et al. [74]	1	0.182	5.099
Shimohara et al. [266]	2	0.561	5.689
Us (haptic thimble)	1	2.270	52.815
Us (6-DOF-X_T)	6	8.176	20.161
Us (6-DOF-Y_T)	6	6.856	27.807
Us (6-DOF-Z_T)	6	16.692	73.785
Us (6-DOF-X_B)	6	4.250	13.125
Us (6-DOF-Y_B)	6	2.445	7.613

Camera and Thermal Images

The images and videos were recorded with Sony (A7 III) and Canon (5D Mark II) DSLR cameras. Thermal images were recorded using a thermal imaging camera (HTi, HT-19). All images and videos were used without post-processing except for adjusting brightness and contrast for readability. Videos were composed using Adobe Premiere.

Bibliography

[1] Ahmed, S., Irshad, L. and Demirel, H.O. 2021. Prototyping Human-Centered Products in the Age of Industry 4.0. *Journal of Mechanical Design*. 143, 7 (Jul. 2021). DOI:https://doi.org/10.1115/1.4050736.

[2] Albaugh, L., Hudson, S. and Yao, L. 2019. Digital Fabrication of Soft Actuated Objects by Machine Knitting. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2019), 1–13.

[3] Alberti, L.B. 1485. De re aedificatoria. (1485).

[4] Alexander, C. 1965. Notes on the Synthesis of Form. (1965).

[5] Alexander, J., Roudaut, A., Steimle, J., Hornb\ae{}k, K., Bruns Alonso, M., Follmer, S. and Merritt, T. 2018. Grand Challenges in Shape-Changing Interface Research. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018), 1–14.

[6] An, B., Tao, Y., Gu, J., Cheng, T., Chen, X. "Anthony," Zhang, X., Zhao, W., Do, Y., Takahashi, S., Wu, H.-Y., Zhang, T. and Yao, L. 2018. Thermorph. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2018), 1–12.

[7] Archer, N.C. and Hopkins, J.B. 2023. Large-Range Rotation-to-Translation Compliant Transmission Mechanism. *Journal of Mechanical Design*. 145, 12 (Dec. 2023). DOI:https://doi.org/10.1115/1.4063160.

[8] Arun, N.D., Yang, H., Yao, L. and Feinberg, A.W. 2023. Nonplanar 3D Printing of Epoxy Using Freeform Reversible Embedding. *Advanced Materials Technologies*. 8, 7 (Apr. 2023), 2201542. DOI:https://doi.org/https://doi.org/10.1002/admt.202201542.

[9] Austern, G., Capeluto, I.G. and Grobman, Y.J. 2018. Rationalization methods in computer aided fabrication: A critical review. *Automation in Construction*. 90, (Jun. 2018), 281–293. DOI:https://doi.org/10.1016/j.autcon.2017.12.027.

[10] Autodesk 2024. Revit. Autodesk.

[11] de Avila Belbute-Peres, F., Smith, K., Allen, K., Tenenbaum, J. and Kolter, J.Z. 2018. End-to-End Differentiable Physics for Learning and Control. *Advances in Neural Information Processing Systems* (2018).

[12] Ayala Monje, C. and Ayala Garcia, C. 2022. Generative Design: Co-Creation Process Between Designer and Computational Thinking. (Jul. 2022), 111–125.

[13] Bader, C., Kolb, D., Weaver, J.C. and Oxman, N. 2016. Data-Driven Material Modeling with Functional Advection for 3D Printing of Materially Heterogeneous Objects. *3D Printing and Additive Manufacturing*. 3, 2 (Jun. 2016), 71–79. DOI:https://doi.org/10.1089/3dp.2016.0026.

[14] Banathy, B.H. 1996. *Designing Social Systems in a Changing World*. Springer US.

[15] Barbič, J., Sin, F. and Grinspun, E. 2012. Interactive editing of deformable simulations. *ACM Transactions on Graphics*. 31, 4 (Aug. 2012), 1–8. DOI:https://doi.org/10.1145/2185520.2185566.

[16] Battaglia, P., Pascanu, R., Lai, M., Rezende, D. and Kavukcuoglu, K. 2016. Interaction networks for learning about objects, relations and physics. *Advances in Neural Information Processing Systems* (2016), 4502–4510.

[17] Battaglia, P.W., Pascanu, R., Lai, M., Rezende, D. and Kavukcuoglu, K. 2016. Interaction Networks for Learning about Objects, Relations and Physics. (Dec. 2016).

[18] Bergamini, A., Christen, R., Maag, B. and Motavalli, M. 2006. A sandwich beam with electrostatically tunable bending stiffness. *Smart Materials and Structures*. 15, 3 (2006), 678. DOI:https://doi.org/10.1088/0964-1726/15/3/002.

[19] Bergou, M., Audoly, B., Vouga, E., Wardetzky, M. and Grinspun, E. 2010. Discrete viscous threads. *ACM Transactions on Graphics*. 29, 4 (Jul. 2010), 1–10. DOI:https://doi.org/10.1145/1778765.1778853.

[20] Bernal, M., Haymaker, J.R. and Eastman, C. 2015. On the role of computational support for designers in action. *Design Studies*. 41, (2015), 163–182. DOI:https://doi.org/https://doi.org/10.1016/j.destud.2015.08.001.

[21] Bharanidaran, R. and Srikanth, S.A. 2016. A new method for designing a compliant mechanism based displacement amplifier. *IOP Conference Series: Materials Science and Engineering*. 149, (Sep. 2016), 012129. DOI:https://doi.org/10.1088/1757-899X/149/1/012129.

[22] Bilal, O.R., Costanza, V., Israr, A., Palermo, A., Celli, P., Lau, F. and Daraio, C. 2020. A Flexible Spiraling-Metasurface as a Versatile Haptic Interface. *Advanced Materials Technologies*. 5, 8 (Aug. 2020), 2000181. DOI:https://doi.org/https://doi.org/10.1002/admt.202000181.

[23] Birla, R. 2014. *Introduction to Tissue Engineering*. Wiley.

[24] Bodaghi, M., Damanpack, A.R. and Liao, W.H. 2016. Self-expanding/shrinking structures by 4D printing. *Smart Materials and Structures*. 25, 10 (Oct. 2016), 105034. DOI:https://doi.org/10.1088/0964-1726/25/10/105034.

[25] Bodaghi, M., Damanpack, A.R. and Liao, W.H. 2018. Triple shape memory polymers by 4D printing. *Smart Materials and Structures*. 27, 6 (Jun. 2018), 065010. DOI:https://doi.org/10.1088/1361-665X/aabc2a.

[26] Bodaghi, M., Noroozi, R., Zolfagharian, A., Fotouhi, M. and Norouzi, S. 2019. 4D Printing Self-Morphing Structures. *Materials*. 12, 8 (Apr. 2019), 1353. DOI:https://doi.org/10.3390/ma12081353.

[27] Boden, M. 2003. *The Creative Mind: Myths and Mechanisms*.

[28] Boland, C.S., Khan, U., Ryan, G., Barwich, S., Charifou, R., Harvey, A., Backes, C., Li, Z., Ferreira, M.S., Möbius, M.E., Young, R.J. and Coleman, J.N. 2016. Sensitive electromechanical sensors using viscoelastic graphene-polymer nanocomposites. *Science*. 354, 6317 (2016), 1257–1260. DOI:https://doi.org/10.1126/science.aag2879.

[29] Bosse, A. 1643. *La Pratique du Trait a Prevues de Mr. Desargues, Lyonnois, pour la Coupe des Pierres en l'Architecture par A. Bosse*.

[30] Brown, N.K., Garland, A.P., Fadel, G.M. and Li, G. 2022. "Deep reinforcement learning for engineering design through topology optimization of elementally discretized design domains." *Materials & Design*. 218, (Jun. 2022), 110672. DOI:https://doi.org/10.1016/j.matdes.2022.110672.

[31] Buckner, T.L., Bilodeau, R.A., Kim, S.Y. and Kramer-Bottiglio, R. 2020. Roboticizing fabric by integrating functional fibers. *Proceedings of the National Academy of Sciences*. 117, 41 (Oct. 2020), 25360–25369. DOI:https://doi.org/10.1073/pnas.2006211117.

[32] Buckner, T.L., Yuen, M.C., Kim, S.Y. and Kramer-Bottiglio, R. 2019. Enhanced Variable Stiffness and Variable Stretchability Enabled by Phase-Changing Particulate Additives. *Advanced Functional Materials*. 29, 50 (Dec. 2019), 1903368. DOI:https://doi.org/https://doi.org/10.1002/adfm.201903368.

[33] Cai, C.J., Winter, S., Steiner, D., Wilcox, L. and Terry, M. 2019. "Hello AI": Uncovering the Onboarding Needs of Medical Practitioners for Human-AI Collaborative Decision-Making. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW (Nov. 2019). DOI:https://doi.org/10.1145/3359206.

[34] Cardoso Llach, D. 2015. *Builders of the Vision: Software and the Imagination of Design*. Routledge.

[35] Chandrasekaran, A., Yadav, D., Chattopadhyay, P., Prabhu, V. and Parikh, D. 2017. It Takes Two to Tango: Towards Theory of AI's Mind. *ArXiv*. abs/1704.00717, (2017).

[36] Chen, D., Levin, D.I.W., Matusik, W. and Kaufman, D.M. 2017. Dynamics-aware numerical coarsening for fabrication design. *ACM Transactions on Graphics*. 36, 4 (Aug. 2017), 1–15. DOI:https://doi.org/10.1145/3072959.3073669.

[37] Chen, D., Levin, D.I.W., Sueda, S. and Matusik, W. 2015. Data-driven finite elements for geometry and material design. *ACM Transactions on Graphics*. 34, 4 (Jul. 2015), 1–10. DOI:https://doi.org/10.1145/2766889.

[38] Chen, M. et al. *Evaluating Large Language Models Trained on Code*.

[39] Chen, M. et al. 2021. Evaluating Large Language Models Trained on Code. (Jul. 2021).

[40] Chen, T., Pauly, M. and Reis, P.M. 2021. A reprogrammable mechanical metamaterial with stable memory. *Nature*. 589, 7842 (2021), 386–390. DOI:https://doi.org/10.1038/s41586-020-03123-5.

[41] Chen, X. "Anthony," Coros, S. and Hudson, S.E. 2018. Medley: A Library of Embeddables to Explore Rich Material Properties for 3D Printed Objects. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018), 1–12.

[42] Chen, X. "Anthony," Tao, Y., Wang, G., Kang, R., Grossman, T., Coros, S. and Hudson, S.E. 2018. Forte. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2018), 1–12.

[43] Chen, X. "Anthony," Tao, Y., Wang, G., Kang, R., Grossman, T., Coros, S. and Hudson, S.E. 2018. Forte: User-Driven Generative Design. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018), 1–12.

[44] Cheng, K., Cuvin, P., Olechowski, A. and Zhou, S. 2023. User Perspectives on Branching in Computer-Aided Design. *Proceedings of the ACM on Human-Computer Interaction*. 7, CSCW2 (Sep. 2023), 1–30. DOI:https://doi.org/10.1145/3610220.

[45] Choi, I., Culbertson, H., Miller, M.R., Olwal, A. and Follmer, S. 2017. Grabity. *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2017), 119–130.

[46] Coelho, M., Ishii, H. and Maes, P. 2008. Surflex: a programmable surface for the design of tangible interfaces. *CHI '08 Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, 2008), 3429–3434.

[47] Collins, S.H., Wiggin, M.B. and Sawicki, G.S. 2015. Reducing the energy cost of human walking using an unpowered exoskeleton. *Nature*. 522, 7555 (2015), 212–215. DOI:https://doi.org/10.1038/nature14288.

[48] Community, B.O. 2018. Blender - a 3D modelling and rendering package.

[49] Coons, S.A. 1963. An outline of the requirements for a computer-aided design system. *Proceedings of the May 21-23, 1963, spring joint computer conference on - AFIPS '63 (Spring)* (New York, New York, USA, 1963), 299.

[50] Coons, S.A. 1963. An outline of the requirements for a computer-aided design system. *Proceedings of the May 21-23, 1963, Spring Joint Computer Conference* (New York, NY, USA, 1963), 299–304.

[51] Cross, N. 2011. *Design Thinking: Understanding How Designers Think and Work*.

[52] Cui, H., Yao, D., Hensleigh, R., Lu, H., Calderon, A., Xu, Z., Davaria, S., Wang, Z., Mercier, P., Tarazaga, P. and Zheng, X. (Rayne) 2022. Design and printing of proprioceptive three-dimensional architected robotic metamaterials. *Science*. 376, 6599 (Jun. 2022), 1287–1293. DOI:https://doi.org/10.1126/science.abn0090.

[53] Dahiya, A. and Braun, D.J. 2017. Efficiently tunable positive-negative stiffness actuator. *2017 IEEE International Conference on Robotics and Automation (ICRA)* (2017), 1235–1240.

[54] David Mans 2024. UI+.

[55] Dawson, C., Vincent, J.F. V. and Rocca, A.-M. 1997. How pine cones open. *Nature*. 390, 6661 (Dec. 1997), 668–668. DOI:https://doi.org/10.1038/37745.

[56] Deng, J., Yang, H., Saini, A., Gaudenz, U.D., Yao, L., Olivier, P. and Mueller, F. 'Floyd' 2023. Dancing Delicacies: Designing Computational Food for Dynamic Dining Trajectories. *Proceedings of the 2023 ACM Designing Interactive Systems Conference* (New York, NY, USA, 2023), 244–262.

[57] Desai, A., Gulwani, S., Hingorani, V., Jain, N., Karkare, A., Marron, M., R, S. and Roy, S. 2015. Program Synthesis using Natural Language.

[58] Dijkstra, E.W. 1959. A note on two problems in connexion with graphs. *Numerische mathematik*. 1, 1 (1959), 269–271.

[59] Diller, S., Majidi, C. and Collins, S.H. 2016. A lightweight, low-power electroadhesive clutch and spring for exoskeleton actuation. *2016 IEEE International Conference on Robotics and Automation (ICRA)* (2016), 682–689.

[60] Dix, A. 2007. Designing for appropriation. *Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI...but Not as We Know It - Volume 2* (Swindon, GBR, 2007), 27–30.

[61] Dourish, P. 2003. The Appropriation of Interactive Technologies: Some Lessons from Placeless Documents. *Computer Supported Cooperative Work (CSCW)*. 12, 4 (2003), 465–490. DOI:https://doi.org/10.1023/A:1026149119426.

[62] Eris, O. 2003. ASKING GENERATIVE DESIGN QUESTIONS: A FUNDAMENTAL COGNITIVE MECHANISM IN DESIGN THINKING. (2003).

[63] Fang, C., Zhang, Y., Dworman, M. and Harrison, C. 2020. Wireality: Enabling Complex Tangible Geometries in Virtual Reality with Worn Multi-String Haptics. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2020), 1–10.

[64] Felton, S., Tolley, M., Demaine, E., Rus, D. and Wood, R. 2014. A method for building self-folding machines. *Science*. 345, 6197 (2014), 644–646. DOI:https://doi.org/10.1126/science.1252610.

[65] Ferguson, G., Allen, J. and Miller, B. 1996. TRAINS-95: Towards a mixed-initiative planning assistant. (Jul. 1996), 70–77.

[66] Fogelson, M.B., Tucker, C. and Cagan, J. 2023. GCP-HOLO: Generating High-Order Linkage Graphs for Path Synthesis. *Journal of Mechanical Design*. 145, 7 (Jul. 2023). DOI:https://doi.org/10.1115/1.4062147.

[67] Follmer, S., Leithinger, D., Olwal, A., Hogge, A. and Ishii, H. 2013. inFORM. *Proceedings of the 26th annual ACM symposium on User interface software and technology* (New York, NY, USA, Oct. 2013), 417–426.

[68] Forman, J., Tabb, T., Do, Y., Yeh, M.-H., Galvin, A. and Yao, L. 2019. ModiFiber. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2019), 1–11.

[69] Forman, J., Tabb, T., Do, Y., Yeh, M.-H., Galvin, A. and Yao, L. 2019. ModiFiber: Two-Way Morphing

Soft Thread Actuators for Tangible Interaction. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2019), 1–11.

[70] Forterre, Y., Skotheim, J.M., Dumais, J. and Mahadevan, L. 2005. How the Venus flytrap snaps. *Nature*. 433, 7024 (Jan. 2005), 421–425. DOI:https://doi.org/10.1038/nature03185.

[71] Frecker, M.I., Ananthasuresh, G.K., Nishiwaki, S., Kikuchi, N. and Kota, S. 1997. Topological Synthesis of Compliant Mechanisms Using Multi-Criteria Optimization. *Journal of Mechanical Design*. 119, 2 (Jun. 1997), 238–245. DOI:https://doi.org/10.1115/1.2826242.

[72] Frediani, G. and Carpi, F. 2020. Tactile display of softness on fingertip. *Scientific Reports*. 10, 1 (2020), 20491. DOI:https://doi.org/10.1038/s41598-020-77591-0.

[73] Fui-Hoon Nah, F., Zheng, R., Cai, J., Siau, K. and Chen, L. 2023. Generative AI and ChatGPT: Applications, challenges, and AI-human collaboration. *Journal of Information Technology Case and Application Research*. 25, 3 (Jul. 2023), 277–304. DOI:https://doi.org/10.1080/15228053.2023.2233814.

[74] Gao, Y., Huang, X., Mann, I.S. and Su, H.-J. 2020. A Novel Variable Stiffness Compliant Robotic Gripper Based on Layer Jamming. *Journal of Mechanisms and Robotics*. 12, 5 (Jun. 2020). DOI:https://doi.org/10.1115/1.4047156.

[75] Gero, K.I., Ashktorab, Z., Dugan, C., Pan, Q., Johnson, J., Geyer, W., Ruiz, M., Miller, S., Millen, D.R., Campbell, M., Kumaravel, S. and Zhang, W. 2020. Mental Models of AI Agents in a Cooperative Game Setting. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2020), 1–12.

[76] Gerovitch, S. 2004. Between human and machine: feedback, control, and computing before cybernetics. *IEEE Annals of The History of Computing - ANNALS*. 26, (Jul. 2004), 71–73. DOI:https://doi.org/10.1109/MAHC.2004.1268412.

[77] Ginder, J.M., Nichols, M.E., Elie, L.D. and Clark, S.M. 2000. Controllable-stiffness components based on magnetorheological elastomers. *Proc.SPIE* (Jun. 2000), 418–425.

[78] Gmeiner, F., Yang, H., Yao, L., Holstein, K. and Martelaro, N. 2023. Exploring Challenges and Opportunities to Support Designers in Learning to Co-create with AI-based Manufacturing Design Tools. *Conference on Human Factors in Computing Systems - Proceedings* (Apr. 2023).

[79] Gmeiner, F., Yang, H., Yao, L., Holstein, K. and Martelaro, N. 2023. Exploring Challenges and Opportunities to Support Designers in Learning to Co-create with AI-based Manufacturing Design Tools. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2023), 1–20.

[80] Godsil, C. and Royle, G. 2001. *Algebraic Graph Theory*. Springer New York.

[81] Gong, J., Seow, O., Honnet, C., Forman, J. and Mueller, S. 2021. MetaSense: Integrating Sensing Capabilities into Mechanical Metamaterial. *The 34th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2021), 1063–1073.

[82] Gong, J., Seow, O., Honnet, C., Forman, J. and Mueller, S. 2021. MetaSense: Integrating Sensing Capabilities into Mechanical Metamaterial. *The 34th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2021), 1063–1073.

[83] Gongora, A.E., Mysore, S., Li, B., Shou, W., Matusik, W., Morgan, E.F., Brown, K.A. and Whiting, E. 2021. Designing Composites with Target Effective Young's Modulus using Reinforcement Learning. *Symposium on Computational Fabrication* (New York, NY, USA, Oct. 2021), 1–11.

[84] Grady, L.J. and Polimeni, J.R. 2010. *Discrete Calculus*. Springer London.

[85] Gräser, P., Linß, S., Harfensteller, F., Torres, M., Zentner, L. and Theska, R. 2021. High-precision and large-stroke XY micropositioning stage based on serially arranged compliant mechanisms with flexure hinges. *Precision Engineering*. 72, (Nov. 2021), 469–479. DOI:https://doi.org/10.1016/j.precisioneng.2021.02.001.

[86] Gu, J., Breen, D.E., Hu, J., Zhu, L., Tao, Y., Van de Zande, T., Wang, G., Zhang, Y.J. and Yao, L. 2019. Geodesy. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2019), 1–10.

[87] Gu, J., Lin, Y., Cui, Q., Li, X., Li, J., Sun, L., Yao, C., Ying, F., Wang, G. and Yao, L. 2022. PneuMesh: Pneumatic-driven Truss-based Shape Changing System. *Conference on Human Factors in Computing Systems - Proceedings* (Apr. 2022).

[88] Gue´rinot, A.E., Magleby, S.P. and Howell, L.L. 2004. Preliminary Design Concepts for Compliant Mechanism Prosthetic Knee Joints. *Volume 2: 28th Biennial Mechanisms and Robotics Conference, Parts A and B* (Jan. 2004), 1103–1111.

[89] Gupta, A. and O'Malley, M.K. 2006. Design of a haptic arm exoskeleton for training and rehabilitation. *IEEE/ASME Transactions on Mechatronics*. 11, 3 (2006), 280–289. DOI:https://doi.org/10.1109/TMECH.2006.875558.

[90] Hajare, R., Reddy, V. and Srikanth, R. 2022. MEMS based sensors – A comprehensive review of commonly used fabrication techniques. *Materials Today: Proceedings*. 49, (2022), 720–730. DOI:https://doi.org/10.1016/j.matpr.2021.05.223.

[91] Hatamizadeh, A., Song, Y. and Hopkins, J.B. 2018. Optimizing the Geometry of Flexure System Topologies Using the Boundary Learning Optimization Tool. *Mathematical Problems in Engineering*. 2018, (2018), 1–14. DOI:https://doi.org/10.1155/2018/1058732.

[92] Herbert, K.M., Fowler, H.E., McCracken, J.M., Schlafmann, K.R., Koch, J.A. and White, T.J. 2022. Synthesis and alignment of liquid crystalline elastomers. *Nature Reviews Materials*. Nature Research.

[93] HODDER, R. V., RIVINGTON, R.N., CALCUTT, L.E. and HART, I.R. 1989. The effectiveness of immediate feedback during the Objective Structured Clinical Examination. *Medical Education*. 23, 2 (Mar. 1989), 184–188. DOI:https://doi.org/10.1111/j.1365-2923.1989.tb00884.x.

[94] Holden, D., Duong, B.C., Datta, S. and Nowrouzezahrai, D. 2019. Subspace neural physics. *Proceedings of the 18th annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, Jul. 2019), 1–12.

[95] Holst, G.L., Teichert, G.H. and Jensen, B.D. 2011. Modeling and Experiments of Buckling Modes and Deflection of Fixed-Guided Beams in Compliant Mechanisms. *Journal of Mechanical Design*. 133, 5 (Jun. 2011). DOI:https://doi.org/10.1115/1.4003922.

[96] Hopkins, J.B. 2013. Designing hybrid flexure systems and elements using Freedom and Constraint Topologies. *Mechanical Sciences*. 4, 2 (2013), 319–331. DOI:https://doi.org/10.5194/ms-4-319-2013.

[97] Hopkins, J.B. and Culpepper, M.L. 2010. Synthesis of multi-degree of freedom, parallel flexure system concepts via Freedom and Constraint Topology (FACT) - Part I: Principles. *Precision Engineering*. 34, 2 (Apr. 2010), 259–270. DOI:https://doi.org/10.1016/j.precisioneng.2009.06.008.

[98] Hopkins, J.B. and Culpepper, M.L. 2010. Synthesis of multi-degree of freedom, parallel flexure system concepts via freedom and constraint topology (FACT). Part II: Practice. *Precision Engineering*. 34, 2 (2010), 271–278. DOI:https://doi.org/https://doi.org/10.1016/j.precisioneng.2009.06.007.

[99] Hopkins, J.B. and Culpepper, M.L. 2011. Synthesis of precision serial flexure systems using freedom and constraint topologies (FACT). *Precision Engineering*. 35, 4 (2011), 638–649. DOI:https://doi.org/https://doi.org/10.1016/j.precisioneng.2011.04.006.

[100] Hopkins, J.B. and McCalib, D. 2016. Synthesizing multi-axis flexure systems with decoupled actuators. *Precision Engineering*. 46, (2016), 206–220. DOI:https://doi.org/https://doi.org/10.1016/j.precisioneng.2016.04.015.

[101] Hopkins, J.B. and Panas, R.M. 2013. Design of flexure-based precision transmission mechanisms using screw theory. *Precision Engineering*. 37, 2 (Apr. 2013), 299–307. DOI:https://doi.org/10.1016/j.precisioneng.2012.09.008.

[102] Hopkins, J.B., Vericella, J.J. and Harvey, C.D. 2014. Modeling and generating parallel flexure elements. *Precision Engineering*. 38, 3 (2014), 525–537. DOI:https://doi.org/https://doi.org/10.1016/j.precisioneng.2014.02.001.

[103] Howell, L.L. 2013. Introduction to Compliant Mechanisms. *Handbook of Compliant Mechanisms*. John Wiley & Sons, Ltd. 1–13.

[104] Howell, L.L., Midha, A. and Norton, T.W. 1996. Evaluation of Equivalent Spring Stiffness for Use in a Pseudo-Rigid-Body Model of Large-Deflection Compliant Mechanisms. *Journal of Mechanical Design*. 118, 1 (Mar. 1996), 126–131. DOI:https://doi.org/10.1115/1.2826843.

[105] Howell, L.L., Olsen, B.M. and Magleby, S.P. 2013. Handbook of compliant mechanisms. (2013).

[106] Hu, Y., Liu, J., Spielberg, A., Tenenbaum, J.B., Freeman, W.T., Wu, J., Rus, D. and Matusik, W. 2019. ChainQueen: A Real-Time Differentiable Physical Simulator for Soft Robotics. *2019 International Conference on Robotics and Automation (ICRA)* (May 2019), 6265–6271.

[107] Huxman, C. and Butler, J. 2021. A Systematic Review of Compliant Mechanisms as Orthopedic Implants. *Journal of Medical Devices*. 15, 4 (Dec. 2021). DOI:https://doi.org/10.1115/1.4052011.

[108] Igarashi, T. and Hughes, J.F. 2001. A suggestive interface for 3D drawing. *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2001), 173–181.

[109] Ion, A., Frohnhofen, J., Wall, L., Kovacs, R., Alistar, M., Lindsay, J., Lopes, P., Chen, H.-T. and Baudisch, P. 2016. Metamaterial Mechanisms. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2016), 529–539.

[110] Ion, A., Lindlbauer, D., Herholz, P., Alexa, M. and Baudisch, P. 2019. Understanding Metamaterial Mechanisms. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2019), 1–14.

[111] Ion, A., Wall, L., Kovacs, R. and Baudisch, P. 2017. Digital Mechanical Metamaterials. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2017), 977–988.

[112] Ishii, H., Leithinger, D., Follmer, S., Zoran, A., Schoessler, P. and Counts, J. 2015. TRANSFORM. *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2015), 687–694.

[113] Jackson, J.A., Messner, M.C., Dudukovic, N.A., Smith, W.L., Bekker, L., Moran, B., Golobic, A.M., Pascall, A.J., Duoss, E.B., Loh, K.J. and Spadaccini, C.M. 2023. Field responsive mechanical metamaterials. *Science Advances*. 4, 12 (May 2023), eaau6419–eaau6419. DOI:https://doi.org/10.1126/sciadv.aau6419.

[114] Jayaraman, A., O'Brien, M.K., Madhavan, S., Mummidisetty, C.K., Roth, H.R., Hohl, K., Tapp, A., Brennan, K., Kocherginsky, M., Williams, K.J., Takahashi, H. and Rymer, W.Z. 2019. Stride management assist exoskeleton vs functional gait training in stroke. *Neurology*. 92, 3 (Jan. 2019), e263 LP-e273. DOI:https://doi.org/10.1212/WNL.0000000000006782.

[115] Jeschke, M.G., van Baar, M.E., Choudhry, M.A., Chung, K.K., Gibran, N.S. and Logsetty, S. 2020. Burn

injury. *Nature Reviews Disease Primers*. 6, 1 (2020), 11. DOI:https://doi.org/10.1038/s41572-020-0145-5.

[116] Jin, Y., Qamar, I., Wessely, M., Adhikari, A., Bulovic, K., Punpongsanon, P. and Mueller, S. 2019. Photo-Chromeleon: Re-Programmable Multi-Color Textures Using Photochromic Dyes. *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2019), 701–712.

[117] Jones, J.C. 1992. *Design Methods*.

[118] Joo, J., Robertson, D., Reich, G. and Smyers, B. Thermally-Activated Reconfigurable Wing Design Using a Non-Monolithic Compliant Mechanism. *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*.

[119] Joo, J., Smyers, B., Beblo, R. and Reich, G. 2011. Load-bearing multi-functional structure with direct thermal harvesting for thermally activated reconfigurable wing design. *ICCM International Conferences on Composite Materials*. (Jan. 2011).

[120] Kan, V., Vargo, E., Machover, N., Ishii, H., Pan, S., Chen, W. and Kakehi, Y. 2017. Organic Primitives: Synthesis and Design of pH-Reactive Materials using Molecular I/O for Sensing, Actuation, and Interaction. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2017), 989–1000.

[121] Kazi, R.H., Grossman, T., Cheong, H., Hashemi, A. and Fitzmaurice, G. 2017. DreamSketch: Early Stage 3D Design Explorations with Sketching and Generative Design. *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2017), 401–414.

[122] Kerr, W.B. and Pellacini, F. 2010. Toward evaluating material design interface paradigms for novice users. *ACM Trans. Graph.* 29, 4 (Jul. 2010). DOI:https://doi.org/10.1145/1778765.1778772.

[123] Khan, S. and Tunçer, B. 2019. Gesture and speech elicitation for 3D CAD modeling in conceptual design. *Automation in Construction*. 106, (2019), 102847. DOI:https://doi.org/https://doi.org/10.1016/j.autcon.2019.102847.

[124] Kim, D., Gwon, M., Kim, B., Ortega-Jimenez, V.M., Han, S., Kang, D., Bhamla, M.S. and Koh, J.-S. 2022. Design of a Biologically Inspired Water-Walking Robot Powered by Artificial Muscle. *Micromachines*. 13, 4 (Apr. 2022), 627. DOI:https://doi.org/10.3390/mi13040627.

[125] Kim, Y., Parada, G.A., Liu, S. and Zhao, X. 2019. Ferromagnetic soft continuum robots. *Science Robotics*. 4, 33 (2019), eaax7329. DOI:https://doi.org/10.1126/scirobotics.aax7329.

[126] Kim, Y.-J., Cheng, S., Kim, S. and Iagnemma, K. 2013. A Novel Layer Jamming Mechanism With Tunable Stiffness Capability for Minimally Invasive Surgery. *IEEE Transactions on Robotics*. 29, 4 (2013), 1031–1042. DOI:https://doi.org/10.1109/TRO.2013.2256313.

[127] Kodnongbua, M., Jones, B., Ahmad, M.B.S., Kim, V. and Schulz, A. 2023. ReparamCAD: Zero-shot CAD Re-Parameterization for Interactive Manipulation. *SIGGRAPH Asia 2023 Conference Papers* (New York, NY, USA, 2023).

[128] Kononenko, O. and Kononenko, I. 2018. Machine Learning and Finite Element Method for Physical Systems Modeling. (Jan. 2018).

[129] Kovacs, R., Ion, A., Lopes, P., Oesterreich, T., Filter, J., Otto, P., Arndt, T., Ring, N., Witte, M., Synytsia, A. and Baudisch, P. 2018. TrussFormer. *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2018), 113–125.

[130] Kovacs, R., Ofek, E., Gonzalez Franco, M., Siu, A.F., Marwecki, S., Holz, C. and Sinclair, M. 2020. Haptic PIVOT. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*

(New York, NY, USA, Oct. 2020), 1046–1059.

[131] Kovacs, R., Seufert, A., Wall, L., Chen, H.-T., Meinel, F., Müller, W., You, S., Brehm, M., Striebel, J., Kommana, Y., Popiak, A., Bläsius, T. and Baudisch, P. 2017. TrussFab. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2017), 2606–2616.

[132] Koyama, Y., Sueda, S., Steinhardt, E., Igarashi, T., Shamir, A. and Matusik, W. 2015. AutoConnect. *ACM Transactions on Graphics*. 34, 6 (Nov. 2015), 1–11. DOI:https://doi.org/10.1145/2816795.2818060.

[133] Krekhov, A., Emmerich, K., Bergmann, P., Cmentowski, S. and Krüger, J. 2017. Self-Transforming Controllers for Virtual Reality First Person Shooters. *Proceedings of the Annual Symposium on Computer-Human Interaction in Play* (New York, NY, USA, Oct. 2017), 517–529.

[134] Krischkowsky, A., Tscheligi, M., Neureiter, K., Muller, M., Polli, A.M. and Verdezoto, N. 2015. Experiences of Technology Appropriation: Unanticipated Users, Usage, Circumstances, and Design. *Proceedings of the 14th European Conference on Computer Supported Cooperative Work* (2015).

[135] Krishnamurti, R. 1980. The arithmetic of shapes. *Environment and Planning B: Planning and Design*. 7, 4 (1980), 463–484. DOI:https://doi.org/10.1068/b070463.

[136] Krishnamurti, R. 1981. The construction of shapes. *Environment and Planning B: Planning and Design*. 8, 1 (1981), 5–40. DOI:https://doi.org/10.1068/b080005.

[137] Krouse, J.K. 1980. CAD/CAM — Bridging the gap from design to production. *IEEE Transactions on Professional Communication*. PC-23, 4 (1980), 189–196. DOI:https://doi.org/10.1109/TPC.1980.6501912.

[138] Kumar, A., Prasad, M., Janyani, V. and Yadav, R.P. 2019. Design, fabrication and reliability study of piezoelectric ZnO based structure for development of MEMS acoustic sensor. *Microsystem Technologies*. 25, 12 (Dec. 2019), 4517–4528. DOI:https://doi.org/10.1007/s00542-019-04524-x.

[139] Kumar, P., Schmidleithner, C., Larsen, N.B. and Sigmund, O. 2021. Topology optimization and 3D printing of large deformation compliant mechanisms for straining biological tissues. *Structural and Multidisciplinary Optimization*. 63, 3 (Mar. 2021), 1351–1366. DOI:https://doi.org/10.1007/s00158-020-02764-4.

[140] Kuppens, P.R., Bessa, M.A., Herder, J.L. and Hopkins, J.B. 2021. Compliant Mechanisms That Use Static Balancing to Achieve Dramatically Different States of Stiffness. *Journal of Mechanisms and Robotics*. 13, 2 (Jan. 2021). DOI:https://doi.org/10.1115/1.4049438.

[141] Kwak, B. and Bae, J. 2018. Compliant mechanosensory composite (CMC): a compliant mechanism with an embedded sensing ability based on electric contact resistance. *Smart Materials and Structures*. 27, 12 (Dec. 2018), 125003. DOI:https://doi.org/10.1088/1361-665X/aae8d3.

[142] Ladický, L., Jeong, S., Solenthaler, B., Pollefeys, M. and Gross, M. 2015. Data-driven fluid simulations using regression forests. *ACM Transactions on Graphics*. 34, 6 (Nov. 2015), 1–9. DOI:https://doi.org/10.1145/2816795.2818129.

[143] Lai, M.-J. and Wang, Y. 2021. *Sparse Solutions of Underdetermined Linear Systems and Their Applications*. Society for Industrial and Applied Mathematics.

[144] Larsson, M., Hironori Yoshida, Umetani, N. and Igarashi, T. 2020. Tsugite: Interactive Design and Fabrication of Wood Joints. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2020), 317–327.

[145] Lazarov, B.S., Schevenels, M. and Sigmund, O. 2011. Robust design of large-displacement compliant mechanisms. *Mechanical Sciences*. 2, 2 (Aug. 2011), 175–182. DOI:https://doi.org/10.5194/ms-2-175-2011.

[146] Lee, R.H., Mulder, E.A.B. and Hopkins, J.B. 2022. Mechanical neural networks: Architected materials that learn behaviors. *Science Robotics*. 7, 71 (Oct. 2022). DOI:https://doi.org/10.1126/scirobotics.abq7278.

[147] Lee, X.Y., Balu, A., Stoecklein, D., Ganapathysubramanian, B. and Sarkar, S. 2019. A Case Study of Deep Reinforcement Learning for Engineering Design: Application to Microfluidic Devices for Flow Sculpting. *Journal of Mechanical Design*. 141, 11 (Nov. 2019). DOI:https://doi.org/10.1115/1.4044397.

[148] Leithinger, D., Follmer, S., Olwal, A. and Ishii, H. 2014. Physical telepresence. *Proceedings of the 27th annual ACM symposium on User interface software and technology* (New York, NY, USA, Oct. 2014), 461–470.

[149] Li, A., Chen, R., Farimani, A.B. and Zhang, Y.J. 2020. Reaction diffusion system prediction based on convolutional neural network. *Scientific Reports*. 10, 1 (Mar. 2020), 3894. DOI:https://doi.org/10.1038/s41598-020-60853-2.

[150] Li, A., Li, H., Li, Z., Zhao, Z., Li, K., Li, M. and Song, Y. 2020. *Programmable droplet manipulation by a magnetic-actuated robot*.

[151] Li, J., Cui, M., Kim, J. and Chen, X.A. 2020. Romeo: A design tool for embedding transformable parts in 3D models to robotically augment default functionalities. *UIST 2020 - Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Oct. 2020), 897–911.

[152] Li, J., Kim, J. and Chen, X. 2019. Robiot: A Design Tool for Actuating Everyday Objects with Automatically Generated 3D Printable Mechanisms. *UIST 2019 - Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (Oct. 2019), 673–685.

[153] Li, J., Samoylov, A., Kim, J. and Chen, X. 2022. Roman: Making Everyday Objects Robotically Manipulable with 3D-Printable Add-on Mechanisms. *Conference on Human Factors in Computing Systems - Proceedings* (Apr. 2022).

[154] Liang, L., Liu, M., Martin, C. and Sun, W. 2018. A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis. *Journal of The Royal Society Interface*. 15, 138 (Jan. 2018), 20170844. DOI:https://doi.org/10.1098/rsif.2017.0844.

[155] Lin, C., Fan, T., Wang, W. and Nießner, M. 2020. Modeling 3D Shapes by Reinforcement Learning.

[156] Lin, C., Fan, T., Wang, W. and Nießner, M. 2020. Modeling 3D Shapes by Reinforcement Learning. 545–561.

[157] Lindsey, B. 2001. *Digital Gehry. Englische Ausgabe.: Material Resistance Digital Construction.* Springer Science & Business Media.

[158] Ling, M., Cao, J., Jiang, Z. and Lin, J. 2016. Theoretical modeling of attenuated displacement amplification for multistage compliant mechanism and its application. *Sensors and Actuators A: Physical*. 249, (Oct. 2016), 15–22. DOI:https://doi.org/10.1016/j.sna.2016.08.011.

[159] Lipkin, H. and Duffy, J. 1985. The Elliptic Polarity of Screws. *Journal of Mechanisms, Transmissions, and Automation in Design*. 107, 3 (Sep. 1985), 377–386. DOI:https://doi.org/10.1115/1.3260725.

[160] Liu, K., Han, L., Hu, W., Ji, L., Zhu, S., Wan, Z., Yang, X., Wei, Y., Dai, Z., Zhao, Z., Li, Z., Wang, P. and Tao, R. 2020. 4D printed zero Poisson's ratio metamaterial with switching function of mechanical and vibration isolation performance. *Materials & Design*. 196, (2020), 109153. DOI:https://doi.org/https://doi.org/10.1016/j.matdes.2020.109153.

[161] Lopes, P., You, S., Cheng, L.-P., Marwecki, S. and Baudisch, P. 2017. Providing Haptics to Walls & Heavy Objects in Virtual Reality by Means of Electrical Muscle Stimulation. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2017), 1471–1482.

[162] Lu, Q., Mao, C., Wang, L. and Mi, H. 2016. LIME: LIquid MEtal Interfaces for Non-Rigid Interaction. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (New York, NY, USA, 2016), 449–452.

[163] Lu, Q., Ou, J., Wilbert, J., Haben, A., Mi, H. and Ishii, H. 2019. milliMorph -- Fluid-Driven Thin Film Shape-Change Materials for Interaction Design. *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2019), 663–672.

[164] Lu, Q., Xu, H., Guo, Y., Wang, J.Y. and Yao, L. 2023. Fluidic Computation Kit: Towards Electronic-free Shape-changing Interfaces. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2023).

[165] Lu, Q., Yu, T., Yi, S., Ding, Y., Mi, H. and Yao, L. 2023. Sustainflatable: Harvesting, Storing and Utilizing Ambient Energy for Pneumatic Morphing Interfaces. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2023).

[166] Luo, D., Gu, J., Qin, F., Wang, G. and Yao, L. 2020. E-seed. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2020), 45–57.

[167] Luo, D., Gu, J., Qin, F., Wang, G. and Yao, L. 2020. E-seed: Shape-changing interfaces that self drill. *UIST 2020 - Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Oct. 2020), 45–57.

[168] Luo, D., Maheshwari, A., Danielescu, A., Li, J., Yang, Y., Tao, Y., Sun, L., Patel, D.K., Wang, G., Yang, S., Zhang, T. and Yao, L. 2023. Autonomous self-burying seed carriers for aerial seeding. *Nature*. 614, 7948 (Feb. 2023), 463–470. DOI:https://doi.org/10.1038/s41586-022-05656-3.

[169] Ma, L.-K., Zhang, Y., Liu, Y., Zhou, K. and Tong, X. 2017. Computational design and fabrication of soft pneumatic objects with desired deformations. *ACM Transactions on Graphics*. 36, 6 (Dec. 2017), 1–12. DOI:https://doi.org/10.1145/3130800.3130850.

[170] Maddock, I.A. *DARPA's Stealth Revolution*.

[171] de Malefette, C., Qi, A., Parakkat, A.D., Cani, M.-P. and Igarashi, T. 2023. PerfectDart: Automatic Dart Design for Garment Fitting. *SIGGRAPH Asia 2023 Technical Communications* (New York, NY, USA, Nov. 2023), 1–4.

[172] Van Manen, T., Janbaz, S. and Zadpoor, A.A. 2017. Programming 2D/3D shape-shifting with hobbyist 3D printers. *Materials Horizons*. 4, 6 (Nov. 2017), 1064–1069. DOI:https://doi.org/10.1039/c7mh00269f.

[173] Mankame, N.D. and Ananthasuresh, G.K. 2002. *CONTACT AIDED COMPLIANT MECHANISMS: CONCEPT AND PRELIMINARIES*.

[174] Markvicka, E., Wang, G., Lee, Y.-C., Laput, G., Majidi, C. and Yao, L. 2019. ElectroDermis: Fully Untethered, Stretchable, and Highly-Customizable Electronic Bandages. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2019), 1–10.

[175] Martin, T., Umetani, N. and Bickel, B. 2015. OmniAD: data-driven omni-directional aerodynamics. *ACM Trans. Graph.* 34, 4 (Jul. 2015). DOI:https://doi.org/10.1145/2766919.

[176] Martin-Guerrero, J.D., Ruperez-Moreno, M.J., Martinez-Martinez, F., Lorente-Garrido, D., Serrano-Lopez, A.J., Monserrat, C., Martinez-Sanchis, S. and Martinez-Sober, M. 2016. Machine Learning for Modeling the Biomechanical Behavior of Human Soft Tissue. *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)* (Dec. 2016), 247–253.

[177] Massachusetts Institute of Technology, Servomechanisms Laboratory Records, AC 151, box X.: 2024. .

[178] Matejka, J., Glueck, M., Bradner, E., Hashemi, A., Grossman, T. and Fitzmaurice, G. 2018. Dream Lens.

Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (New York, NY, USA, Apr. 2018), 1–12.

[179] Matejka, J., Glueck, M., Bradner, E., Hashemi, A., Grossman, T. and Fitzmaurice, G. 2018. Dream Lens: Exploration and Visualization of Large-Scale Generative Design Datasets. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018), 1–12.

[180] Matsuoka, J., Berger, R.A., Berglund, L.J. and An, K.-N. 2006. An Analysis of Symmetry of Torque Strength of the Forearm Under Resisted Forearm Rotation in Normal Subjects. *Journal of Hand Surgery*. 31, 5 (May 2006), 801–805. DOI:https://doi.org/10.1016/j.jhsa.2006.02.019.

[181] Mattson, C.A. 2013. Synthesis through Rigid-Body Replacement. *Handbook of Compliant Mechanisms*. Wiley. 109–121.

[182] Megaro, V., Zehnder, J., Bächer, M., Coros, S., Gross, M. and Thomaszewski, B. 2017. A computational design tool for compliant mechanisms. *ACM Transactions on Graphics*. 36, 4 (Aug. 2017), 1–12. DOI:https://doi.org/10.1145/3072959.3073636.

[183] Mekaouche, A., Chapelle, F. and Balandraud, X. 2018. A compliant mechanism with variable stiffness achieved by rotary actuators and shape-memory alloy. *Meccanica*. 53, 10 (2018), 2555–2571. DOI:https://doi.org/10.1007/s11012-018-0844-0.

[184] Menges, A. and Ahlquist, S. 2011. Computational Design Thinking. (2011).

[185] Merriam, E.G., Jones, J.E., Magleby, S.P. and Howell, L.L. 2013. Monolithic 2 DOF fully compliant space pointing mechanism. *Mechanical Sciences*. 4, 2 (2013), 381–390. DOI:https://doi.org/10.5194/ms-4-381-2013.

[186] Milner, T.E. and Franklin, D.W. 1998. Characterization of multijoint finger stiffness: dependence on finger posture and force direction. *IEEE Transactions on Biomedical Engineering*. 45, 11 (1998), 1363–1375. DOI:https://doi.org/10.1109/10.725333.

[187] Minori, A.F., Civici, U., Shen, C., Paul, S., Bergbreiter, S., Temel, Z. and Yao, L. 2023. RevLock: A Reversible Self-Locking Mechanism Driven by Linear Actuators for Foldable Robots and Systems. *IEEE Robotics and Automation Letters*. 8, 11 (Nov. 2023), 7432–7439. DOI:https://doi.org/10.1109/LRA.2023.3315215.

[188] Mintchev, S., Salerno, M., Cherpillod, A., Scaduto, S. and Paik, J. 2019. A portable three-degrees-of-freedom force feedback origami robot for human–robot interactions. *Nature Machine Intelligence*. 1, 12 (2019), 584–593. DOI:https://doi.org/10.1038/s42256-019-0125-1.

[189] Mirhoseini, A. et al. 2021. A graph placement methodology for fast chip design. *Nature*. 594, 7862 (Jun. 2021), 207–212. DOI:https://doi.org/10.1038/s41586-021-03544-w.

[190] Miruchna, V., Walter, R., Lindlbauer, D., Lehmann, M., von Klitzing, R. and Müller, J. 2015. GelTouch: Localized Tactile Feedback Through Thin, Programmable Gel. *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (New York, NY, USA, 2015), 3–10.

[191] Mitchell, W.J., Inouye, A.S. and Blumenthal, M.S. 2003. *Beyond Productivity: Information, Technology, Innovation, and Creativity*. National Academy Press.

[192] Miyajima, H. and Mehregany, M. 1995. High-aspect-ratio photolithography for MEMS applications. *Journal of Microelectromechanical Systems*. 4, 4 (1995), 220–229. DOI:https://doi.org/10.1109/84.475549.

[193] Moore, E.F. 1962. Machine Models of Self-Reproduction. (1962).

[194] Mor, H., Yu, T., Nakagaki, K., Miller, B.H., Jia, Y. and Ishii, H. 2020. Venous Materials: Towards

Interactive Fluidic Mechanisms. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2020), 1–14.

[195] Morita, T. and Sugano, S. 1995. Design and development of a new robot joint using a mechanical impedance adjuster. *Proceedings of 1995 IEEE International Conference on Robotics and Automation* (1995), 2469–2475 vol.3.

[196] Mrowca, D., Zhuang, C., Wang, E., Haber, N., Fei-Fei, L., Tenenbaum, J.B. and Yamins, D.L.K. 2018. Flexible Neural Representation for Physics Prediction. (Jun. 2018).

[197] Mueller, J., Lewis, J.A. and Bertoldi, K. 2022. Architected Multimaterial Lattices with Thermally Programmable Mechanical Response. *Advanced Functional Materials*. 32, 1 (Jan. 2022), 2105128. DOI:https://doi.org/https://doi.org/10.1002/adfm.202105128.

[198] Muller, M., Neureiter, K., Verdezoto, N., Krischkowsky, A., Al Zubaidi-Polli, A.M. and Tscheligi, M. 2016. Collaborative Appropriation: How Couples, Teams, Groups and Communities Adapt and Adopt Technologies. *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion* (New York, NY, USA, 2016), 473–480.

[199] Muñoz, V.G., Muneta, L.M., Carrasco-Gallego, R., Marquez, J. de J. and Hidalgo-Carvajal, D. 2020. Evaluation of the circularity of recycled pla filaments for 3D printers. *Applied Sciences (Switzerland)*. 10, 24 (Dec. 2020), 1–12. DOI:https://doi.org/10.3390/app10248967.

[200] Murer, M., Maurer, B., Huber, H., Aslan, I. and Tscheligi, M. 2015. TorqueScreen. *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, Jan. 2015), 161–164.

[201] Muto, H., Gondo, Y., Inagaki, H., Masui, Y., Nakagawa, T., Ogawa, M., Onoguchi, W., Ishioka, Y., Numata, K. and Yasumoto, S. 2023. Human-body Analogy Improves Mental Rotation Performance in People Aged 86 to 97 Years. *Collabra Psychology*. 9, (Jul. 2023), 74785. DOI:https://doi.org/10.1525/collabra.74785.

[202] Nakagaki, K., Dementyev, A., Follmer, S., Paradiso, J.A. and Ishii, H. 2016. ChainFORM. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2016), 87–96.

[203] Nakagaki, K., Fitzgerald, D., Ma, Z. (John), Vink, L., Levine, D. and Ishii, H. 2019. inFORCE. *Proceedings of the Thirteenth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, Mar. 2019), 615–623.

[204] Nakayama, R., Suzuki, R., Nakamaru, S., Niiyama, R., Kawahara, Y. and Kakehi, Y. 2019. MorphIO. *Proceedings of the 2019 on Designing Interactive Systems Conference* (New York, NY, USA, Jun. 2019), 975–986.

[205] Narumi, K., Koyama, K., Suto, K., Noma, Y., Sato, H., Tachi, T., Sugimoto, M., Igarashi, T. and Kawahara, Y. 2023. Inkjet 4D Print: Self-folding Tessellated Origami Objects by Inkjet UV Printing. *ACM Transactions on Graphics*. 42, 4 (Aug. 2023). DOI:https://doi.org/10.1145/3592409.

[206] Narumi, K., Qin, F., Liu, S., Cheng, H.-Y., Gu, J., Kawahara, Y., Islam, M. and Yao, L. 2019. Self-healing UI: Mechanically and Electrically Self-healing Materials for Sensing and Actuation Interfaces. *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2019), 293–306.

[207] Nelson, T.G. and Herder, J.L. 2018. Developable compliant-aided rolling-contact mechanisms. *Mechanism and Machine Theory*. 126, (Aug. 2018), 225–242. DOI:https://doi.org/10.1016/j.mechmachtheory.2018.04.013.

[208] von Neumann, J. 1963. The General and Logical Theory of Au-tomata. (1963).

[209] Newman, M., Kwan, I., Schucan Bird, K. and Hoo, H.-T. 2021. The Impact of Feedback on Student Attainment: A Systematic Review. *Education Endowment Foundation*. (2021).

[210] Nie, Z., Jiang, H. and Kara, L.B. 2020. Stress Field Prediction in Cantilevered Structures Using Convolutional Neural Networks. *Journal of Computing and Information Science in Engineering*. 20, 1 (Feb. 2020). DOI:https://doi.org/10.1115/1.4044097.

[211] Niiyama, R., Yao, L. and Ishii, H. 2014. Weight and volume changing device with liquid metal transfer. *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction* (New York, NY, USA, Feb. 2014), 49–52.

[212] Nilsson, N.C., Zenner, A. and Simeone, A.L. 2021. Propping Up Virtual Reality With Haptic Proxies. *IEEE Computer Graphics and Applications*. 41, 5 (2021), 104–112. DOI:https://doi.org/10.1109/MCG.2021.3097671.

[213] Nisser, M., Zhu, J., Chen, T., Bulovic, K., Punpongsanon, P. and Mueller, S. 2019. Sequential Support. *Proceedings of the Thirteenth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, Mar. 2019), 669–676.

[214] Niu, M.-Q. and Chen, L.-Q. 2022. Nonlinear vibration isolation via a compliant mechanism and wire ropes. *Nonlinear Dynamics*. 107, 2 (Jan. 2022), 1687–1702. DOI:https://doi.org/10.1007/s11071-021-06588-9.

[215] Noble, D.F. 2011. *Forces of Production*.

[216] O'Connor, S. 2021. Exoskeletons in Nursing and Healthcare: A Bionic Future. *Clinical Nursing Research*. 30, 8 (Aug. 2021), 1123–1126. DOI:https://doi.org/10.1177/10547738211038365.

[217] Olberding, S., Wessely, M. and Steimle, J. 2014. PrintScreen: fabricating highly customizable thin-film touch-displays. *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2014), 281–290.

[218] Ou, J., Ma, Z., Peters, J., Dai, S., Vlavianos, N. and Ishii, H. 2018. KinetiX - designing auxetic-inspired deformable material structures. *Computers & Graphics*. 75, (Oct. 2018), 72–81. DOI:https://doi.org/10.1016/j.cag.2018.06.003.

[219] Ou, J., Skouras, M., Vlavianos, N., Heibeck, F., Cheng, C.Y., Peters, J. and Ishii, H. 2016. AeroMorph - Heat-sealing inflatable shape-change materials for interaction design. *UIST 2016 - Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Oct. 2016), 121–132.

[220] Ou, J., Yao, L., Tauber, D., Steimle, J., Niiyama, R. and Ishii, H. 2014. JamSheets: Thin Interfaces with Tunable Stiffness Enabled by Layer Jamming. *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction* (New York, NY, USA, 2014), 65–72.

[221] Papazafeiropoulos, G., Muñiz-Calvente, M. and Martínez-Pañeda, E. 2017. Abaqus2Matlab: A suitable tool for finite element post-processing. *Advances in Engineering Software*. 105, (Mar. 2017), 9–16. DOI:https://doi.org/10.1016/j.advengsoft.2017.01.006.

[222] Parasuraman, R. and Riley, V. 1997. Humans and automation: Use, misuse, disuse, abuse. *Human factors*. 39, 2 (1997), 230–253.

[223] Parasuraman, R. and Riley, V. 1997. Humans and Automation: Use, Misuse, Disuse, Abuse. *Human Factors*. 39, 2 (1997), 230–253. DOI:https://doi.org/10.1518/001872097778543886.

[224] Park, Y.-L., Chen, B., Pérez-Arancibia, N.O., Young, D., Stirling, L., Wood, R.J., Goldfield, E.C. and Nagpal, R. 2014. Design and control of a bio-inspired soft wearable robotic device for ankle–foot rehabilitation. *Bioinspiration & Biomimetics*. 9, 1 (2014), 16007. DOI:https://doi.org/10.1088/1748-3182/9/1/016007.

[225] Patek, S.N., Baio, J.E., Fisher, B.L. and Suarez, A. V. 2006. Multifunctionality and mechanical origins: Ballistic jaw propulsion in trap-jaw ants. *Proceedings of the National Academy of Sciences*. 103, 34 (Aug. 2006), 12787–12792. DOI:https://doi.org/10.1073/pnas.0604290103.

[226] Patel, D.K., Zhong, K., Xu, H., Islam, M.F. and Yao, L. 2023. Sustainable Morphing Matter: Design and Engineering Practices. *Advanced Materials Technologies*. 8, 23 (2023), 2300678. DOI:https://doi.org/https://doi.org/10.1002/admt.202300678.

[227] Peebles, D. 2019. Modelling alternative strategies for mental rotation. *Proceedings of the ICCM 2019* (United States, Oct. 2019).

[228] Pérez, J., Thomaszewski, B., Coros, S., Bickel, B., Canabal, J.A., Sumner, R. and Otaduy, M.A. 2015. Design and fabrication of flexible rod meshes. *ACM Transactions on Graphics*. 34, 4 (Jul. 2015), 1–12. DOI:https://doi.org/10.1145/2766998.

[229] Polygerinos, P., Wang, Z., Galloway, K.C., Wood, R.J. and Walsh, C.J. 2015. Soft robotic glove for combined assistance and at-home rehabilitation. *Robotics and Autonomous Systems*. 73, (2015), 135–143. DOI:https://doi.org/https://doi.org/10.1016/j.robot.2014.08.014.

[230] Poon, R. and Hopkins, J.B. 2019. Phase-Changing Metamaterial Capable of Variable Stiffness and Shape Morphing. *Advanced Engineering Materials*. 21, 12 (Dec. 2019), 1900802. DOI:https://doi.org/https://doi.org/10.1002/adem.201900802.

[231] Poppinga, S., Körner, A., Sachse, R., Born, L., Westermeier, A., Hesse, L., Knippers, J., Bischoff, M., Gresser, G.T. and Speck, T. 2016. Compliant Mechanisms in Plants and Architecture. 169–193.

[232] Pyun, K.R., Rogers, J.A. and Ko, S.H. 2022. Materials and devices for immersive virtual reality. *Nature Reviews Materials*. 7, 11 (Oct. 2022), 841–843. DOI:https://doi.org/10.1038/s41578-022-00501-5.

[233] Qamar, I.P.S., Groh, R., Holman, D. and Roudaut, A. 2018. HCI meets Material Science: A Literature Review of Morphing Materials for the Design of Shape-Changing Interfaces. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018), 1–23.

[234] Qi, J. and Buechley, L. 2010. Electronic popables: exploring paper-based computing through an interactive pop-up book. *Proceedings of the Fourth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, 2010), 121–128.

[235] Qin, F., Cheng, H.-Y., Sneeringer, R., Vlachostergiou, M., Acharya, S., Liu, H., Majidi, C., Islam, M. and Yao, L. 2021. ExoForm: Shape Memory and Self-Fusing Semi-Rigid Wearables. *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2021).

[236] Ramos, M.M., Carnaz, L., Mattiello, S.M., Karduna, A.R. and Zanca, G.G. 2019. Shoulder and elbow joint position sense assessment using a mobile app in subjects with and without shoulder pain - between-days reliability. *Physical Therapy in Sport*. 37, (May 2019), 157–163. DOI:https://doi.org/10.1016/j.ptsp.2019.03.016.

[237] Raviv, D., Zhao, W., McKnelly, C., Papadopoulou, A., Kadambi, A., Shi, B., Hirsch, S., Dikovsky, D., Zyracki, M., Olguin, C., Raskar, R. and Tibbits, S. 2014. Active Printed Materials for Complex Self-Evolving Deformations. *Scientific Reports*. 4, 1 (Dec. 2014), 7422. DOI:https://doi.org/10.1038/srep07422.

[238] Reddy, J.N. 2014. *An Introduction to Nonlinear Finite Element Analysis, 2nd Edn*. Oxford University Press.

[239] Reissner, L., Fischer, G., List, R., Taylor, W.R., Giovanoli, P. and Calcagni, M. 2019. Minimal detectable difference of the finger and wrist range of motion: comparison of goniometry and 3D motion analysis. *Journal of Orthopaedic Surgery and Research*. 14, 1 (2019), 173. DOI:https://doi.org/10.1186/s13018-019-1177-y.

[240] Retelny, D., Bernstein, M.S. and Valentine, M.A. 2017. No Workflow Can Ever Be Enough: How

Crowdsourcing Workflows Constrain Complex Work. *Proc. ACM Hum.-Comput. Interact.* 1, CSCW (Dec. 2017). DOI:https://doi.org/10.1145/3134724.

[241] Robertson, B., Walther, J. and Radcliffe, D. 2007. Creativity and the Use of CAD Tools: Lessons for Engineering Design Education From Industry. *Journal of Mechanical Design*. 129, (Jul. 2007). DOI:https://doi.org/10.1115/1.2722329.

[242] Robertson, B.F. and Radcliffe, D.F. 2009. Impact of CAD tools on creative problem solving in engineering design. *Computer-Aided Design*. 41, 3 (2009), 136–146. DOI:https://doi.org/https://doi.org/10.1016/j.cad.2008.06.007.

[243] Rodríguez-Fernández, A., Lobo-Prat, J. and Font-Llagunes, J.M. 2021. Systematic review on wearable lower-limb exoskeletons for gait training in neuromuscular impairments. *Journal of NeuroEngineering and Rehabilitation*. 18, 1 (2021), 22. DOI:https://doi.org/10.1186/s12984-021-00815-5.

[244] Rozvany, G.I.N. ed. 1997. *Topology Optimization in Structural Mechanics*. Springer Vienna.

[245] Ryu, N., Lee, W., Kim, M.J. and Bianchi, A. 2020. ElaStick. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2020), 1035–1045.

[246] Sahoo, D.R., Neate, T., Tokuda, Y., Pearson, J., Robinson, S., Subramanian, S. and Jones, M. 2018. Tangible Drops. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2018), 1–14.

[247] Sahoo, D.R., Neate, T., Tokuda, Y., Pearson, J., Robinson, S., Subramanian, S. and Jones, M. 2018. Tangible Drops: A Visio-Tactile Display Using Actuated Liquid-Metal Droplets. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018), 1–14.

[248] Sanchez-Gonzalez, A., Heess, N., Springenberg, J.T., Merel, J., Riedmiller, M., Hadsell, R. and Battaglia, P. 2018. Graph networks as learnable physics engines for inference and control. (Jun. 2018).

[249] Sanchez-Gonzalez, A., Heess, N., Springenberg, J.T., Merel, J., Riedmiller, M., Hadsell, R. and Battaglia, P. 2018. Graph networks as learnable physics engines for inference and control. *35th International Conference on Machine Learning, ICML 2018* (2018).

[250] Sanchez-Villamañan, M. del C., Gonzalez-Vargas, J., Torricelli, D., Moreno, J.C. and Pons, J.L. 2019. Compliant lower limb exoskeletons: a comprehensive review on mechanical design principles. *Journal of NeuroEngineering and Rehabilitation*. 16, 1 (2019), 55. DOI:https://doi.org/10.1186/s12984-019-0517-9.

[251] Sarakoglou, I., Garcia-Hernandez, N., Tsagarakis, N.G. and Caldwell, D.G. 2012. A High Performance Tactile Feedback Display and Its Integration in Teleoperation. *IEEE Transactions on Haptics*. 5, 3 (2012), 252–263. DOI:https://doi.org/10.1109/TOH.2012.20.

[252] Sareen, H., Umapathi, U., Shin, P., Kakehi, Y., Ou, J., Maes, P. and Ishii, H. 2017. Printflatables: Printing human-scale, functional and dynamic inflatable objects. *Conference on Human Factors in Computing Systems - Proceedings* (May 2017), 3669–3680.

[253] Sattari Sarebangholi, M. and Najafi, F. 2022. Design of a path generating compliant mechanism using a novel rigid-body-replacement method. *Meccanica*. 57, 7 (Jul. 2022), 1701–1711. DOI:https://doi.org/10.1007/s11012-022-01527-3.

[254] Savage, V., Schmidt, R., Grossman, T., Fitzmaurice, G. and Hartmann, B. 2014. A series of tubes. *Proceedings of the 27th annual ACM symposium on User interface software and technology* (New York, NY, USA, Oct. 2014), 3–12.

[255] Sawicki, G.S., Beck, O.N., Kang, I. and Young, A.J. 2020. The exoskeleton expansion: improving walking and running economy. *Journal of NeuroEngineering and Rehabilitation*. 17, 1 (2020), 25. DOI:https://doi.org/10.1186/s12984-020-00663-9.

[256] Saxena, A. and Ananthasuresh, G.K. 2001. Topology Synthesis of Compliant Mechanisms for Nonlinear Force-Deflection and Curved Path Specifications. *Journal of Mechanical Design*. 123, 1 (Mar. 2001), 33–42. DOI:https://doi.org/10.1115/1.1333096.

[257] Schön, D.A. 1983. The Reflective Practitioner. (1983).

[258] Schulz, A., Sung, C., Spielberg, A., Zhao, W., Cheng, R., Grinspun, E., Rus, D. and Matusik, W. 2017. Interactive robogami: An end-to-end system for design of robots with ground locomotion. *The International Journal of Robotics Research*. 36, 10 (2017), 1131–1147. DOI:https://doi.org/10.1177/0278364917723465.

[259] Schulz, A., Wang, H., Grinspun, E., Solomon, J. and Matusik, W. 2018. Interactive exploration of design trade-offs. *ACM Trans. Graph.* 37, 4 (Jul. 2018). DOI:https://doi.org/10.1145/3197517.3201385.

[260] Schulz, A., Xu, J., Zhu, B., Zheng, C., Grinspun, E. and Matusik, W. 2017. Interactive design space exploration and optimization for CAD models. *ACM Transactions on Graphics*. 36, 4 (Aug. 2017), 1–14. DOI:https://doi.org/10.1145/3072959.3073688.

[261] Scilingo, E.P., Bianchi, M., Grioli, G. and Bicchi, A. 2010. Rendering Softness: Integration of Kinesthetic and Cutaneous Information in a Haptic Device. *IEEE Transactions on Haptics*. 3, 2 (2010), 109–118. DOI:https://doi.org/10.1109/TOH.2010.2.

[262] Sesek, R.F., Khalighi, M., Bloswick, D.S., Anderson, M. and Tuckett, R.P. 2007. Effects of prolonged wrist flexion on transmission of sensory information in carpal tunnel syndrome. *The Journal of Pain*. 8, 2 (2007), 137–151.

[263] Shan, W., Lu, T. and Majidi, C. 2013. Soft-matter composites with electrically tunable elastic rigidity. *Smart Materials and Structures*. 22, 8 (2013), 85005. DOI:https://doi.org/10.1088/0964-1726/22/8/085005.

[264] Shaw, L.A., Sun, F., Portela, C.M., Barranco, R.I., Greer, J.R. and Hopkins, J.B. 2019. Computationally efficient design of directionally compliant metamaterials. *Nature Communications*. 10, 1 (2019), 291. DOI:https://doi.org/10.1038/s41467-018-08049-1.

[265] Shigeyama, J., Hashimoto, T., Yoshida, S., Narumi, T., Tanikawa, T. and Hirose, M. 2019. Transcalibur. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2019), 1–11.

[266] Shimohara, S., Lee, R.H. and Hopkins, J.B. 2022. Compliant mechanisms that achieve binary stiffness along multiple degrees of freedom. *Journal of Composite Materials*. 57, 4 (Dec. 2022), 645–657. DOI:https://doi.org/10.1177/00219983221146262.

[267] Shrobe, H. 1993. Understanding Linkages. *Proceedings of the National Conference on Artificial Intelligence* (Jan. 1993), 620–625.

[268] Simon, H.A. 1996. *The Sciences of the Artificial, 3rd Edition*. The MIT Press.

[269] Siviy, C., Baker, L.M., Quinlivan, B.T., Porciuncula, F., Swaminathan, K., Awad, L.N. and Walsh, C.J. 2023. Opportunities and challenges in the development of exoskeletons for locomotor assistance. *Nature Biomedical Engineering*. 7, 4 (2023), 456–472. DOI:https://doi.org/10.1038/s41551-022-00984-1.

[270] Skouras, M., Thomaszewski, B., Coros, S., Bickel, B. and Gross, M. 2013. Computational design of actuated deformable characters. *ACM Transactions on Graphics*. 32, 4 (Jul. 2013), 1–10. DOI:https://doi.org/10.1145/2461912.2461979.

[271] Slade, P., Kochenderfer, M.J., Delp, S.L. and Collins, S.H. 2022. Personalizing exoskeleton assistance while walking in the real world. *Nature*. 610, 7931 (2022), 277–282. DOI:https://doi.org/10.1038/s41586-022-05191-1.

[272] Smith, P.H. 2004. *The Body of the Artisan: Art and Experience in the Scientific Revolution*.

[273] Song, Y., Panas, R.M., Chizari, S., Shaw, L.A., Jackson, J.A., Hopkins, J.B. and Pascall, A.J. 2019. Additively manufacturable micro-mechanical logic gates. *Nature Communications*. 10, 1 (Feb. 2019), 882. DOI:https://doi.org/10.1038/s41467-019-08678-0.

[274] Stamm, T.A., Machold, K.P., Smolen, J.S., Fischer, S., Redlich, K., Graninger, W., Ebner, W. and Erlacher, L. 2002. Joint protection and home hand exercises improve hand function in patients with hand osteoarthritis: A randomized controlled trial. *Arthritis Care & Research*. 47, 1 (Feb. 2002), 44–49. DOI:https://doi.org/https://doi.org/10.1002/art1.10246.

[275] Stops, L., Leenhouts, R., Gao, Q. and Schweidtmann, A.M. 2023. Flowsheet generation through hierarchical reinforcement learning and graph neural networks. *AIChE Journal*. 69, 1 (Jan. 2023). DOI:https://doi.org/10.1002/aic.17938.

[276] Su, H.-J., Shi, H. and Yu, J. 2012. A Symbolic Formulation for Analytical Compliance Analysis and Synthesis of Flexure Mechanisms. *Journal of Mechanical Design*. 134, 5 (Apr. 2012). DOI:https://doi.org/10.1115/1.4006441.

[277] Suebnukarn, S., Haddawy, P., Rhienmora, P., Jittimanee, P. and Viratket, P. 2010. Augmented Kinematic Feedback from Haptic Virtual Reality for Dental Skill Acquisition. *Journal of Dental Education*. 74, 12 (Dec. 2010), 1357–1366. DOI:https://doi.org/https://doi.org/10.1002/j.0022-0337.2010.74.12.tb05011.x.

[278] Sun, F. and Hopkins, J.B. 2017. Mobility and Constraint Analysis of Interconnected Hybrid Flexure Systems Via Screw Algebra and Graph Theory. *Journal of Mechanisms and Robotics*. 9, 3 (Mar. 2017). DOI:https://doi.org/10.1115/1.4035993.

[279] Sun, J. and Zhao, J. 2019. An Adaptive Walking Robot With Reconfigurable Mechanisms Using Shape Morphing Joints. *IEEE Robotics and Automation Letters*. 4, 2 (Apr. 2019), 724–731. DOI:https://doi.org/10.1109/LRA.2019.2893439.

[280] Sun, T., Yang, S. and Lian, B. 2020. *Finite and Instantaneous Screw Theory in Robotic Mechanism*. Springer Singapore.

[281] Sun, W., Williamson, A.S., Sukhnandan, R., Majidi, C., Yao, L., Feinberg, A.W. and Webster-Wood, V.A. 2023. Biodegradable, Sustainable Hydrogel Actuators with Shape and Stiffness Morphing Capabilities via Embedded 3D Printing. *Advanced Functional Materials*. 33, 36 (Sep. 2023). DOI:https://doi.org/10.1002/adfm.202303659.

[282] Sutherland, I.E. 1963. Sketchpad: a man-machine graphical communication system. *Proceedings of the May 21-23, 1963, Spring Joint Computer Conference* (New York, NY, USA, 1963), 329–346.

[283] Suzuki, R., Hedayati, H., Zheng, C., Bohn, J.L., Szafir, D., Do, E.Y.L., Gross, M.D. and Leithinger, D. 2020. RoomShift: Room-scale Dynamic Haptics for VR with Furniture-moving Swarm Robots. *Conference on Human Factors in Computing Systems - Proceedings* (Apr. 2020).

[284] Suzuki, R., Ofek, E., Sinclair, M., Leithinger, D. and Gonzalez-Franco, M. 2021. HapticBots: Distributed Encountered-type Haptics for VR with Multiple Shape-changing Mobile Robots. *UIST 2021 - Proceedings of the 34th Annual ACM Symposium on User Interface Software and Technology* (Oct. 2021), 1269–1281.

[285] Suzuki, R., Zheng, C., Kakehi, Y., Yeh, T., Do, E.Y.-L., Gross, M.D. and Leithinger, D. 2019. ShapeBots: Shape-changing Swarm Robots. *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2019), 493–505.

[286] Swindells, C., Unden, A. and Sang, T. 2003. TorqueBAR. *Proceedings of the 5th international conference on Multimodal interfaces* (New York, NY, USA, Nov. 2003), 52–59.

[287] Taboada, C., Brunetti, A.E., Pedron, F.N., Neto, F.C., Estrin, D.A., Bari, S.E., Chemes, L.B., Lopes, N.P.,

Lagorio, M.G. and Faivovich, J. 2017. Naturally occurring fluorescence in frogs. *Proceedings of the National Academy of Sciences*. 114, 14 (2017), 3672–3677. DOI:https://doi.org/10.1073/pnas.1701053114.

[288] Tahouni, Y., Qamar, I.P.S. and Mueller, S. 2020. NURBSforms: A Modular Shape-Changing Interface for Prototyping Curved Surfaces. *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, 2020), 403–409.

[289] Tan, X., Chen, S., Wang, B., Tang, J., Wang, L., Zhu, S., Yao, K. and Xu, P. 2020. Real-time tunable negative stiffness mechanical metamaterial. *Extreme Mechanics Letters*. 41, (2020), 100990. DOI:https://doi.org/https://doi.org/10.1016/j.eml.2020.100990.

[290] Tao, Y., Do, Y., Yang, H., Lee, Y.-C., Wang, G., Mondoa, C., Cui, J., Wang, W. and Yao, L. 2019. Morphlour. *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2019), 329–340.

[291] Tao, Y., Do, Y., Yang, H., Lee, Y.C., Wang, G., Mondoa, C., Cui, J., Wang, W. and Yao, L. 2019. Morphlour: Personalized flour-based morphing food induced by dehydration or hydration method. *UIST 2019 - Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (2019), 329–340.

[292] Tao, Y., Wang, G., Zhang, C., Lu, N., Zhang, X., Yao, C. and Ying, F. 2017. WeaveMesh: A low-fidelity and low-cost prototyping approach for 3D models created by flexible assembly. *Conference on Human Factors in Computing Systems - Proceedings* (2017), 509–518.

[293] Teyssier, J., Saenko, S. V, van der Marel, D. and Milinkovitch, M.C. 2015. Photonic crystals cause active colour change in chameleons. *Nature Communications*. 6, 1 (2015), 6368. DOI:https://doi.org/10.1038/ncomms7368.

[294] Then Mozhi, G., Dhanalakshmi, K. and Choi, S.-B. 2023. Design and Control of Monolithic Compliant Gripper Using Shape Memory Alloy Wires. *Sensors*. 23, 4 (Feb. 2023), 2052. DOI:https://doi.org/10.3390/s23042052.

[295] Tomiyama, T., Umeda, Y. and Yoshikawa, H. 1993. A CAD for Functional Design. *CIRP Annals*. 42, 1 (1993), 143–146. DOI:https://doi.org/https://doi.org/10.1016/S0007-8506(07)62412-3.

[296] Trott, P. 2008. *Innovation management and new product development*. Financial Times Prentice Hall.

[297] Tsai, H.-R., Rekimoto, J. and Chen, B.-Y. 2019. ElasticVR. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2019), 1–10.

[298] Tsang, S., Balakrishnan, R., Singh, K. and Ranjan, A. 2004. A suggestive interface for image guided 3D sketching. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2004), 591–598.

[299] Tummala, Y., Wissa, A., Frecker, M. and Hubbard, J.E. 2014. Design and optimization of a contact-aided compliant mechanism for passive bending. *Journal of Mechanisms and Robotics*. 6, 3 (Jun. 2014). DOI:https://doi.org/10.1115/1.4027702.

[300] Turkkan, O.A., Venkiteswaran, V.K. and Su, H.-J. 2018. Rapid conceptual design and analysis of spatial flexure mechanisms. *Mechanism and Machine Theory*. 121, (Mar. 2018), 650–668. DOI:https://doi.org/10.1016/j.mechmachtheory.2017.11.025.

[301] UltiMaker Thingiverse: things tagged with “compliant mechanism”: .

[302] Umetani, N., Igarashi, T. and Mitra, N.J. 2012. Guided exploration of physically valid shapes for furniture design. *ACM Trans. Graph.* 31, 4 (Jul. 2012). DOI:https://doi.org/10.1145/2185520.2185582.

[303] Umetani, N., Koyama, Y., Schmidt, R. and Igarashi, T. 2014. Pteromys: interactive design and

optimization of free-formed free-flight model airplanes. *ACM Trans. Graph.* 33, 4 (Jul. 2014). DOI:https://doi.org/10.1145/2601097.2601129.

[304] Umetani, N., Panotopoulou, A., Schmidt, R. and Whiting, E. 2016. Printone: interactive resonance simulation for free-form print-wind instrument design. *ACM Trans. Graph.* 35, 6 (Dec. 2016). DOI:https://doi.org/10.1145/2980179.2980250.

[305] Umetani, N., Takayama, K., Mitani, J. and Igarashi, T. 2011. A Responsive Finite Element Method to Aid Interactive Geometric Modeling. *IEEE Computer Graphics and Applications*. 31, 5 (2011), 43–53. DOI:https://doi.org/10.1109/MCG.2010.46.

[306] Vanswearingen, J.M. 1983. Measuring Wrist Muscle Strength. *Journal of Orthopaedic & Sports Physical Therapy*. 4, 4 (Apr. 1983), 217–228. DOI:https://doi.org/10.2519/jospt.1983.4.4.217.

[307] Vink, L., Kan, V., Nakagaki, K., Leithinger, D., Follmer, S., Schoessler, P., Zoran, A. and Ishii, H. 2015. TRANSFORM as Adaptive and Dynamic Furniture. *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2015), 183–183.

[308] Volkov, A.G., Adesina, T., Markin, V.S. and Jovanov, E. 2008. Kinetics and Mechanism of *Dionaea muscipula* Trap Closing. *Plant Physiology*. 146, 2 (Feb. 2008), 323–324. DOI:https://doi.org/10.1104/pp.107.108241.

[309] Vyas, D., Poelman, W., Nijholt, A. and De Bruijn, A. 2012. Smart material interfaces. *CHI '12 Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, May 2012), 1721–1726.

[310] Wakita, A., Nakano, A. and Kobayashi, N. 2010. Programmable blobs: a rheologic interface for organic shape design. *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, 2010), 273–276.

[311] Wang, G. et al. 2023. ThermoFit: Thermoforming Smart Orthoses via Metamaterial Structures for Body-Fitting and Component-Adjusting. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 1 (Mar. 2023). DOI:https://doi.org/10.1145/3580806.

[312] Wang, G., Cheng, T., Do, Y., Yang, H., Tao, Y., Gu, J., An, B. and Yao, L. 2018. Printed Paper Actuator. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2018), 1–12.

[313] Wang, G., Tao, Y., Capunaman, O.B., Yang, H. and Yao, L. 2019. A-line. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, May 2019), 1–12.

[314] Wang, G., Tao, Y., Capunaman, O.B., Yang, H. and Yao, L. 2019. A-line: 4D Printing Morphing Linear Composite Structures. *Conference on Human Factors in Computing Systems - Proceedings* (2019), 1–12.

[315] Wang, G., Yang, H., Yan, Z., Gecer Ulu, N., Tao, Y., Gu, J., Kara, L.B. and Yao, L. 2018. 4DMesh. *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2018), 623–635.

[316] Wang, G., Yang, H., Yan, Z., Ulu, N.G., Tao, Y., Gu, J., Kara, L.B. and Yao, L. 2018. 4DMesh: 4D printing morphing non-developable mesh surfaces. *UIST 2018 - Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (2018), 623–635.

[317] Wang, G., Yao, L., Wang, W., Ou, J., Cheng, C.-Y. and Ishii, H. 2016. xPrint: A Modularized Liquid Printer for Smart Materials Deposition. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2016), 5743–5752.

[318] Wang, W., Yao, L., Zhang, T., Cheng, C.-Y., Levine, D. and Ishii, H. 2017. Transformative Appetite. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA,

May 2017), 6123–6132.

[319] Wang, Y., Li, L., Hofmann, D., Andrade, J.E. and Daraio, C. 2021. Structured fabrics with tunable mechanical properties. *Nature*. 596, 7871 (2021), 238–243. DOI:https://doi.org/10.1038/s41586-021-03698-7.

[320] Wang, Y., Xiao, M., Xia, Z., Li, P. and Gao, L. 2023. From Computer-Aided Design (CAD) Toward Human-Aided Design (HAD): An Isogeometric Topology Optimization Approach. *Engineering*. 22, (2023), 94–105. DOI:https://doi.org/https://doi.org/10.1016/j.eng.2022.07.013.

[321] Wang, Z.J., Hong, W., Wu, Z.L. and Zheng, Q. 2017. Site-Specific Pre-Swelling-Directed Morphing Structures of Patterned Hydrogels. *Angewandte Chemie International Edition*. 56, 50 (2017), 15974–15978. DOI:https://doi.org/https://doi.org/10.1002/anie.201708926.

[322] Webster-Wood, V.A., Guix, M., Xu, N.W., Behkam, B., Sato, H., Sarkar, D., Sanchez, S., Shimizu, M. and Parker, K.K. 2023. Biohybrid robots: recent progress, challenges, and perspectives. *Bioinspiration and Biomimetics*. 18, 1 (Jan. 2023). DOI:https://doi.org/10.1088/1748-3190/ac9c3b.

[323] Weigel, M., Lu, T., Bailly, G., Oulasvirta, A., Majidi, C. and Steimle, J. 2015. iSkin: Flexible, Stretchable and Visually Customizable On-Body Touch Sensors for Mobile Computing. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (New York, NY, USA, 2015), 2991–3000.

[324] Weiser, M. 1991. The Computer for the 21st Century. *Scientific American*. 265, 3 (Sep. 1991), 94–104. DOI:https://doi.org/10.1038/scientificamerican0991-94.

[325] Weiser, M. 1999. The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.* 3, 3 (Jul. 1999), 3–11. DOI:https://doi.org/10.1145/329124.329126.

[326] Wenjing Ye, Mukherjee, S. and MacDonald, N.C. 1998. Optimal shape design of an electrostatic comb drive in microelectromechanical systems. *Journal of Microelectromechanical Systems*. 7, 1 (Mar. 1998), 16–26. DOI:https://doi.org/10.1109/84.661380.

[327] Wessely, M., Tsandilas, T. and Mackay, W.E. 2016. Stretchis: Fabricating Highly Stretchable User Interfaces. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (New York, NY, USA, 2016), 697–704.

[328] Wienert, V., Sick, H. and zur Mühlen, J. 1983. Local thermal stress tolerance of human skin. *Anasthesie, Intensivtherapie, Notfallmedizin*. 18, 2 (Apr. 1983), 88–90.

[329] Wojcikowski, K. and Kirk, L. 2013. Immediate detailed feedback to test-enhanced learning: An effective online educational tool. *Medical Teacher*. 35, 11 (Nov. 2013), 915–919. DOI:https://doi.org/10.3109/0142159X.2013.826793.

[330] Worgan, P., Reuss, K. and Mueller, S. 2019. Integrating Electronic Components into Deformable Objects Based on User Interaction Data. *Proceedings of the Thirteenth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, Mar. 2019), 345–350.

[331] Wu, C., Zhao, H., Nandi, C., Lipton, J.I., Tatlock, Z. and Schulz, A. 2019. Carpentry compiler. *ACM Trans. Graph.* 38, 6 (Nov. 2019). DOI:https://doi.org/10.1145/3355089.3356518.

[332] Xie, Y.M., Burry, J., Lee, T.U., Ma, J., Lee, M. and Tachi, T. 2023. *Design and Evaluation of Compliant Hinges for Deployable Thick Origami Structures*.

[333] Xu, H., Li, Y., Chen, Y. and Barbič, J. 2015. Interactive Material Design Using Model Reduction. *ACM Transactions on Graphics*. 34, 2 (Mar. 2015), 1–14. DOI:https://doi.org/10.1145/2699648.

[334] Yang, C., Ye, W. and Li, Q. 2022. Review of the performance optimization of parallel manipulators.

Mechanism and Machine Theory. 170, (2022), 104725. DOI:https://doi.org/https://doi.org/10.1016/j.mechmachtheory.2022.104725.

[335] Yang, H., Johnson, T., Zhong, K., Patel, D., Olson, G., Majidi, C., Islam, M. and Yao, L. 2022. ReCompFig: Designing Dynamically Reconfigurable Kinematic Devices Using Compliant Mechanisms and Tensioning Cables. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2022).

[336] Yang, H., Luo, D., Qian, K. and Yao, L. 2021. Freeform Fabrication of Fluidic Edible Materials. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2021).

[337] Yang, H., Qian, K., Liu, H., Yu, Y., Gu, J., McGehee, M., Zhang, Y.J. and Yao, L. 2020. SimuLearn. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, Oct. 2020), 71–84.

[338] Yao, J., Kaufman, D.M., Gingold, Y. and Agrawala, M. 2017. Interactive Design and Stability Analysis of Decorative Joinery for Furniture. *ACM Transactions on Graphics*. 36, 2 (Apr. 2017), 1–16. DOI:https://doi.org/10.1145/3054740.

[339] Yao, L., Niiyama, R., Ou, J., Follmer, S., Della Silva, C. and Ishii, H. 2013. PneUI. *Proceedings of the 26th annual ACM symposium on User interface software and technology* (New York, NY, USA, Oct. 2013), 13–22.

[340] Yao, L., Ou, J., Cheng, C.-Y., Steiner, H., Wang, W., Wang, G. and Ishii, H. 2015. bioLogic. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2015), 1–10.

[341] Yao, L., Ou, J., Cheng, C.Y., Steiner, H., Wang, W., Wang, G. and Ishii, H. 2015. Biologic: Natto cells as nanoactuators for shape changing interfaces. *Conference on Human Factors in Computing Systems - Proceedings* (2015), 1–10.

[342] Yu, Y., Liu, H., Qian, K., Yang, H., McGehee, M., Gu, J., Luo, D., Yao, L. and Zhang, Y.J. 2020. Material characterization and precise finite element analysis of fiber reinforced thermoplastic composites for 4D printing. *Computer-Aided Design*. 122, (May 2020), 102817. DOI:https://doi.org/10.1016/j.cad.2020.102817.

[343] Yu, Y., Liu, H., Qian, K., Yang, H., McGehee, M., Gu, J., Luo, D., Yao, L. and Zhang, Y.J. 2020. Material characterization and precise finite element analysis of fiber reinforced thermoplastic composites for 4D printing. *CAD Computer Aided Design*. 10, 1 (2020), 3894. DOI:https://doi.org/10.1016/j.cad.2020.102817.

[344] Yu, Y., Qian, K., Yang, H., Yao, L. and Zhang, Y.J. 2022. Hybrid IGA-FEA of fiber reinforced thermoplastic composites for forward design of AI-enabled 4D printing. *Journal of Materials Processing Technology*. 302, (Apr. 2022), 117497. DOI:https://doi.org/10.1016/j.jmatprotec.2022.117497.

[345] Zappetti, D., Jeong, S.H., Shintake, J. and Floreano, D. 2019. Phase Changing Materials-Based Variable-Stiffness Tensegrity Structures. *Soft Robotics*. 7, 3 (Dec. 2019), 362–369. DOI:https://doi.org/10.1089/soro.2019.0091.

[346] Zehnder, J., Knoop, E., Bächer, M. and Thomaszewski, B. 2017. Metasilicone. *ACM Transactions on Graphics*. 36, 6 (Dec. 2017), 1–13. DOI:https://doi.org/10.1145/3130800.3130881.

[347] Zeng, X., Hurd, C., Su, H.-J., Song, S. and Wang, J. 2020. A parallel-guided compliant mechanism with variable stiffness based on layer jamming. *Mechanism and Machine Theory*. 148, (2020), 103791. DOI:https://doi.org/https://doi.org/10.1016/j.mechmachtheory.2020.103791.

[348] Zenner, A. and Kruger, A. 2017. Shifty: A Weight-Shifting Dynamic Passive Haptic Proxy to Enhance

Object Perception in Virtual Reality. *IEEE Transactions on Visualization and Computer Graphics*. 23, 4 (Apr. 2017), 1285–1294. DOI:https://doi.org/10.1109/TVCG.2017.2656978.

[349] Zhang, G., He, H. and Katabi, D. 2019. Circuit-GNN: Graph Neural Networks for Distributed Circuit Design. *Proceedings of the 36th International Conference on Machine Learning* (Jan. 2019), 7364–7373.

[350] Zhang, M., Xie, S.Q., Li, X., Zhu, G., Meng, W., Huang, X. and Veale, A.J. 2018. Adaptive Patient-Cooperative Control of a Compliant Ankle Rehabilitation Robot (CARR) With Enhanced Training Safety. *IEEE Transactions on Industrial Electronics*. 65, 2 (2018), 1398–1407. DOI:https://doi.org/10.1109/TIE.2017.2733425.

[351] Zhang, Q., Zhang, K. and Hu, G. 2016. Smart three-dimensional lightweight structure triggered from a thin composite sheet via 3D printing technique. *Scientific Reports*. 6, 1 (Feb. 2016), 22431. DOI:https://doi.org/10.1038/srep22431.

[352] Zhang, R., McNeese, N.J., Freeman, G. and Musick, G. 2021. "An Ideal Human": Expectations of AI Teammates in Human-AI Teaming. *Proc. ACM Hum.-Comput. Interact.* 4, CSCW3 (Jan. 2021). DOI:https://doi.org/10.1145/3432945.

[353] Zhang, Z., Xu, Z., Emu, L., Wei, P., Chen, S., Zhai, Z., Kong, L., Wang, Y. and Jiang, H. 2023. Active mechanical haptics with high-fidelity perceptions for immersive virtual reality. *Nature Machine Intelligence*. (2023). DOI:https://doi.org/10.1038/s42256-023-00671-z.

[354] Zhao, A., Xu, J., Konaković-Luković, M., Hughes, J., Spielberg, A., Rus, D. and Matusik, W. 2020. RoboGrammar. *ACM Transactions on Graphics*. 39, 6 (Dec. 2020), 1–16. DOI:https://doi.org/10.1145/3414685.3417831.

[355] Zhao, A., Xu, J., Salazar, J., Wang, W., Ma, P., Rus, D. and Matusik, W. 2022. Graph Grammar-Based Automatic Design for Heterogeneous Fleets of Underwater Robots. *2022 International Conference on Robotics and Automation (ICRA)* (May 2022), 3143–3149.

[356] Zheng, Z., Yao, Y., Liu, J.A., Sun, Y. and Yeow, J.T.W. 2019. Highly sensitive CMUT-based humidity sensors built with nitride-to-oxide wafer bonding technology. *Sensors and Actuators B: Chemical*. 294, (Sep. 2019), 123–131. DOI:https://doi.org/10.1016/j.snb.2019.05.003.

[357] Zhong, K., Fernandes Minori, A., Wu, D., Yang, H., Islam, M.F. and Yao, L. 2023. EpoMemory: Multi-State Shape Memory for Programmable Morphing Interfaces. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2023).

[358] Zhong, Q., Zhu, J., Fish, F.E., Kerr, S.J., Downs, A.M., Bart-Smith, H. and Quinn, D.B. 2021. Tunable stiffness enables fast and efficient swimming in fish-like robots. *Science Robotics*. 6, 57 (Aug. 2021), eabe4088–eabe4088. DOI:https://doi.org/10.1126/scirobotics.abe4088.

[359] Zhu, J.-H., Zhang, W.-H. and Xia, L. 2016. Topology Optimization in Aircraft and Aerospace Structures Design. *Archives of Computational Methods in Engineering*. 23, 4 (Dec. 2016), 595–622. DOI:https://doi.org/10.1007/s11831-015-9151-2.

[360] Zoeller, A.C. and Drewing, K. 2020. A Systematic Comparison of Perceptual Performance in Softness Discrimination with Different Fingers. *Attention, Perception, & Psychophysics*. 82, 7 (2020), 3696–3709. DOI:https://doi.org/10.3758/s13414-020-02100-4.