CompAct: Designing Interconnected Compliant Mechanisms with Targeted Actuation Transmissions

Humphrey Yang
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
hanliny@cs.cmu.edu

I-Chao Shen
The University of Tokyo
Graduate School of Information Science and Technology
Tokyo, Japan
jdilyshen@gmail.com

Nikolas Martelaro
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
nmartelaro@gmail.com

Bo Zhu
Georgia Institute of Technology
Atlanta, Georgia, USA
bo.zhu@gatech.edu

Haoran Xie
JAIST
Ishikawa, Japan
xie@jaist.ac.jp

Takeo Igarashi
The University of Tokyo
Tokyo, Japan
takeo@acm.org

Lining Yao
Mechanical Engineering Department
University of California, Berkeley
Berkeley, California, USA
liningy@berkeley.edu

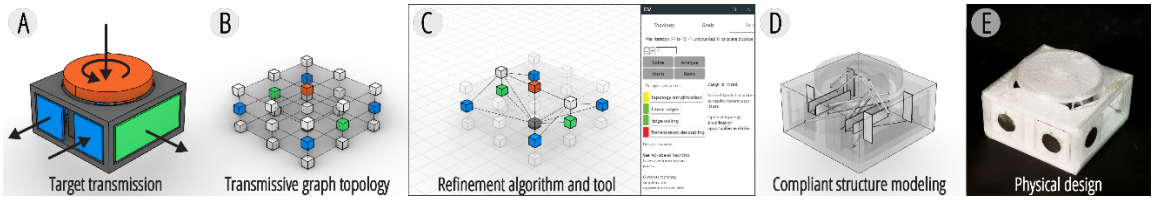

Figure 1: CompAct design pipeline and tools. (A) Given the desired kinematic transmission properties, our tool helps designers to (B) model them as a graph topology and (C) optimize it toward target transmission functions. (D) The tool then parses the optimized graph topology and procedurally generates compliant joint modeling instructions to help users produce a (E) physical design.

Abstract

Compliant mechanisms enable the creation of compact and easy-to-fabricate devices for tangible interaction. This work explores interconnected compliant mechanisms consisting of multiple joints and rigid bodies to transmit and process displacements as signals that result from physical interactions. As these devices are difficult to design due to their vast and complex design space, we developed a graph-based design algorithm and computational tool to help users program and customize such computational functions and procedurally model physical designs. When combined with active materials with actuation and sensing capabilities, these devices can also render and detect haptic interaction. Our design examples demonstrate the tool's capability to respond to relevant HCI concepts, including building modular physical interface toolkits, encrypting tangible interactions, and customizing user augmentation for accessibility. We believe the tool will facilitate the generation of new interfaces with enriched affordance.

CCS Concepts

• **CCS; Human-centered computing;**; • **Human computer interaction (HCI);**; • **Interaction devices;**; • **Haptic devices.**;

Keywords

Tangible interface, compliant mechanism, design tool, shape-changing interface

CHI '25, Yokohama, Japan

ACM ISBN 979-8-4007-1394-1/2025/04
https://doi.org/10.1145/3706598.3714307

ACM Reference Format:
Humphrey Yang, I-Chao Shen, Nikolas Martelaro, Bo Zhu, Haoran Xie, Takeo Igarashi, and Lining Yao. 2025. CompAct: Designing Interconnected Compliant Mechanisms with Targeted Actuation Transmissions. In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26–May 01, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/3706598.3714307

1 Introduction

Compliant mechanisms garner interest in HCI to enable tangible interactions. They are single-part flexible structures that afford motions through structural deformation, eliminating the need for complex hinges and bearings. They are used to build compact, easy-to-fabricate devices that provide force feedback [33, 66] or simulate hand-feel [62] in small, customizable form factors, as well as kinematic sculptures, gadgets, and toys [21, 22, 24, 38]. When arranged into an interconnected network, such mechanisms can also transmit and transform displacements at distant locations, creating mechanical circuits where displacements and forces resulting from interactions can trigger targeted physical responses.

However, unlike single-joint mechanisms, designing interconnected compliant mechanisms is inherently challenging due to the interdependencies among compliant joints. The behavior of one joint can significantly impact the entire structure. These systems also exhibit nonlinear structural behavior, where small design changes can drastically alter overall device kinematics. Moreover, with each added joint increasing the number of interdependent parameters, the design space grows exponentially with device complexity. Literature often accounts for these challenges with finite element simulation and topology optimization [26, 47, 68]. Still, such tools are computationally expensive and are often limited to planar mechanism designs with fewer attainable degrees of freedom and actuation modes. Alternatively, conversion-based methods [37, 38] require an existing linkage mechanism as input and, therefore, fall short in customizability and adaptability to new use contexts or users.

This work explores enabling customizable and interactive design of interconnected compliant mechanisms with targeted actuation behaviors, creating *CompAct* devices. We develop a tool (Figure 1) to facilitate creating interactive mechanical devices from end (input-output transmission behaviors, Figure 1A) to end (physical model, Figure 1E). The tool features an algorithm that uses graph-based and componentized kinematic structure design to save computation time (Figure 1B), making it possible to interactively (i.e., delays in the magnitude of seconds) design three-dimensional transmissive compliant mechanisms with motions in six degrees of freedom. The tool employs heuristics engines (Figure 2A) to help designers create and refine a topology toward desired transmission functions (Figure 1C) and provides procedural modeling instructions to create the compliant structure (Figure 1D).

Two design opportunities are identified with the proposed design tool. First, existing compliant mechanism design tools are typically focused on a single actuation mode and simple interactions (e.g., button presses [33, 66]) and less applicable to more complex or conditional responses that require multiple concurrent actuation modes. By contrast, the presented design tool can support users in programming multiple transmission modes into a device to respond to the user differently depending on the received manipulation. Second, while compliant mechanisms are conventionally passive in tangible interactions, inspired by the ongoing trend in HCI [7, 34], we show that integrating active materials can augment the affordance of interconnected, transmissive compliant mechanisms. Specifically, actuatable [9, 35, 57] and sensing [5, 12, 56] materials can be incorporated in a CompAct device as the input or output

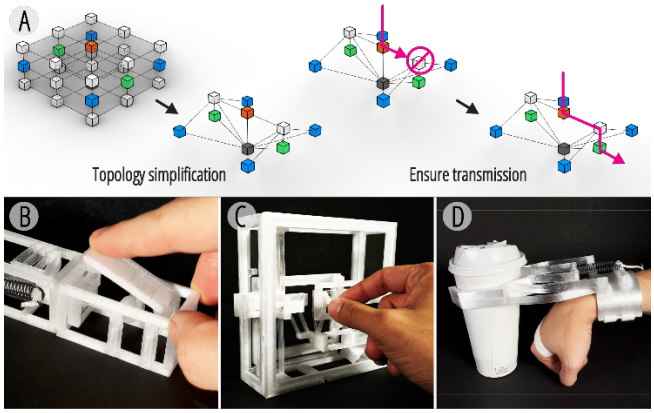

Figure 2: (A) Design heuristics for refining a transmissive graph topology, including removing redundant components (left) and resolving disrupted transmission flow (right). Design examples: (B) modular physical interface, (C) smart lock mechanism, and (D) augmentative wearable device.

of actuation. The design tool helps users to tailor – amplify and transform - simple active material behaviors (e.g., linear contraction [9], resistance changes [46]) into specific, predictable responses for tangible interactions.

Using this design tool, we explore and implement several examples that showcase the enabled HCI interaction design opportunities. These examples include modular physical interfaces that can detect and render haptic feedback and be reconfigured into various form factors depending on the use scenario (Figure 2B), a smart lock mechanism that requires a specific manipulation sequence to unlock and enables encrypted and conditional tangible interactions (Figure 2C), and customizing wearable devices that detect hand gestures to enhance and proxy object manipulation (Figure 2D). We advocate that CompAct devices are ideal for applications that require flexibility, adaptability, and minimal footprint, thus enabling functional and dynamic physical interactions to blend into everyday life. The intellectual contribution of this work includes:

1. A computational design tool to support various workflows (inverse, suggestive, forward) to produce an interconnected compliant mechanism design.
2. Numerical validation of the design tool's effectiveness by mechanically validating the generated design's performance against targeted transmissive functions.
3. Demonstrations of active material integration.
4. Design examples demonstrating the enabled design space.

2 Related work

2.1 Designing Kinematic Devices for Physical Interaction

Kinematic devices in HCI leverage mechanical structures to provide different interactivity, such as animated props [23, 51, 61], installations [27, 28], and robotic agents [13, 32, 41, 52]. These systems use mechanical linkages and hinges (e.g., linkage scissor mechanism in Xs [61]) or compliant structures (e.g., flexible truss [51])

to create mechanical features. Their design is often challenging as they require the user to consider transformative behaviors; there is more than meets the eye. For this reason, literature often provides computational tools to facilitate their design process. In the editing environment, the tools allow users to composite building blocks into an assembly and preview their collective mechanical performance using simulations. This enables a forward design workflow where the user may iteratively modify the design until reaching the desired outcome. Some tools also provide functions to help users optimize structural rigidity [27] or generate control schemes [13] for targeted motion. Still, given target kinematic functions, predicting what device compositions are required to satisfy the specifications remains difficult. CompAct contributes to this research community by providing a design tool that helps designers generate and refine a transmissive kinematic structure of compliant mechanisms. In addition to the tool, we also explore new design opportunities for transmissive kinematic structures, such as encrypting tangible interaction.

2.2 Interactive Compliant Mechanisms

Compliant mechanisms are structures that enable kinematic motions by structural deformation. They are popular in mechanical engineering for their miniaturizability, reliability, and assembly-free fabrication advantages. Examples of applications include aerospace engineering [39], microelectromechanical systems [59], and more. In HCI, researchers favor their simplicity and manufacturability in prototyping mechanical devices without requiring assembly. For instance, Metamaterial Mechanisms [21, 23] enable users to create gadgets and tools (e.g., door locks, pliers) using 3D-printable planar, grid-like structures. Broader applications also include producing action figures [38, 48] and creating haptic devices [22, 33, 62, 66]. In this work, the transmission-centric design approach allows designers to better navigate these design spaces by enabling users to prescribe conditional interaction behaviors (i.e., establishing multiple transmission modes and conditioning its feedback on the user input). For example, combining CompAct with FlexHaptics [33] or Shape-Haptics [66] may enable haptic devices whose feedback depends on how the user manipulates the input stick. Such transmission prescription also allows mechanical structures to possess embedded kinematic sensors and display functionalities without requiring external controllers, similar to the fluidic intelligence demonstrated by Venous Materials [40].

While compliant mechanisms are often passive to tangible interaction, there is a growing interest in combining them with sensing or actuating materials to better support physical interactions, such as MetaSense [12] and ReCompFig [62] using sensing materials to detect user manipulation for digital contents or ConeAct [34] and Robotic Metamaterials [7] using actuatable materials to create structures that dynamically change shape and adapt functions. In this work, we demonstrate through application examples that the presented tool can also support active material integration. Moreover, while prior literature mostly focuses on developing and characterizing individual devices or building blocks, the presented design tool will enable users to systematically generate and customize novel designs with computational guidance.

2.3 Designing Compliant Mechanisms

Despite the growing interest in and utility of compliant mechanisms, few tools exist to create transmissive structures, and even fewer tools can integrate active materials. Available design tools often target single-joint designs [33, 62, 66]. Existing computational methods for generating interconnected compliant mechanisms also have their limitations in customizing a transmissive structure. While topology optimization [29] can generate a mechanical structure to satisfy a desired transmission behavior, it is slow to compute due to iterative finite element simulations. As a result, such method is often limited to planar structures that only afford in-plane motions [26]. The computation time also makes topology optimization less suitable for interactive and iterative design prototyping. Moreover, designing for multiple load (transmission) modes through topology optimization may lead to competing design gradients that cause the design to oscillate (i.e., repeatedly decimating and reinstating an element) or become invalid (i.e., non-manufacturable) [47, 68]. Alternatively, the rigid body replacement method [37, 38] requires an existing linkage system as input and converts it into a compliant mechanism; therefore, they do not support ground-up synthesis from arbitrary transmission specifications. By contrast, CompAct builds upon existing graph- and component-based transmissive-compliant mechanism analysis approaches [15–17] to develop a design tool. Compared to topology optimization, our tool, leveraging the freedom and constraint topology design method, affords near real-time (<5 seconds) user interaction even when handling 3D design problems, therefore more suitable for implementing interactive design tools for concept prototyping by the HCI community. The tool may also robustly attain multiple transmission modes by algebraically modeling viable flexure layouts. Compared to the rigid body replacement method, the presented design tool also allows users to customize transmission functions without requiring an existing linkage system as the input.

2.4 Active Materials in HCI

Digital design and fabrication tools empower HCI researchers to create and augment objects with functional materials [58]. For instance, actuatable materials may be used to create shape-changing interfaces that render dynamic haptic feedback [4, 6, 9, 10] or shape [34, 56]. Compared to electromechanical systems, active materials can be conveniently customized into different form factors, making them suitable for integration with daily life objects. However, these materials often have primitive behaviors (e.g., linearly contract [9, 56], swell [6]) and must rely on additional deformable structures (e.g., fabric [2, 14], paper [56, 65]) to direct their motions to attain specific transformative behaviors. Literature often provides examples of primitive structure design to guide users in applying the materials. Still, available design methods (e.g., [53]) mostly focus on designing unitary functional building blocks (i.e., such as a bending sheet) and planar or linear structures; they put less emphasis on conducting, amplifying, and compositing active material transformations for complex interactions. CompAct responds to this challenge and provides a design tool for integrating active materials in compliant mechanical structures to create complex interactions. In addition to actuatable materials, sensing materials that respond to mechanical loads or deformations (e.g., capacitive

sensors [12, 56]) may also be incorporated into CompAct devices. In this work, we use common, off-the-shelf active materials (i.e., shape-memory alloy seen in [14, 34] and rubber stretch sensors in [62]) to demonstrate this feature.

3 Background

3.1 Compliant Mechanism

Compliant mechanisms consist of rigid stages and deformable flexures. Rigid stages are bulky and resist deformation, while flexures allow deformation in specific directions due to their slender aspect ratios. The type and arrangement of the connecting flexures determine the degree of freedom (DOF) between two rigid stages. A rod flexure (Figure 3A) creates one degree of constraint (DOC) along its longitudinal axis, blocking axial translation but allowing other DOF. A blade flexure (Figure 3B) restricts translations in its plane and rotations about its face normal but permits rotations within its plane and translation along its normal. A compliant joint may include multiple flexures, and the joint's DOF will depend on the flexure layout. The freedom and constraint topology (FACT) method [15–17] models the relationship between flexure arrangements and joint mobilities. This approach represents flexural DOC and DOF as lines in 3D space (i.e., six-dimensional screw vectors), each encoded by its direction and a reference point on the line. DOF vectors can be linearly combined to generate motions between stages, forming the joint's freedom space. Conversely, constraint vectors create the constraint space, which describes the compliant joint's flexure layout. Given a desired DOF, the FACT method identifies the necessary flexure layout to achieve the targeted mobility, which can then be parsed into instructions for modeling compliant joints [62].

3.2 Transmissive Compliant Mechanism

An interconnected compliant mechanism assembles multiple rigid stages and deformable joints into a transmissive network that allows displacements and forces to flow between distal ends (Figure 3C). In such a network, a displacement along a joint's DOF causes deformation, dissipating the load. Conversely, displacements along the joint's constraint space will cause minimal deformation and flow through. This arrangement enables the mechanism to relay, negate, and transform displacements and loads to provide computational behavior. The network's transmissive behavior depends on its topology (i.e., rigid body and connectivity) and each joint's DOF/DOC configuration. Literature [50] has developed tools for analyzing such behaviors by combining the FACT method with graph theory, where the rigid stages form the nodes, and the compliant joints are the edges. Given its topology and joint configuration, this method enables the prediction of a mechanism's transmissive behaviors. However, a FACT design tool for synthesizing compliant mechanisms to achieve targeted transmissive behaviors remains unavailable. This work addresses this gap by adapting the FACT analysis algorithm to create a computational design-synthesis tool.

4 Overview of design algorithm

Given a set of transmission design goals, an interconnected compliant mechanism is designed in three subsequent steps: 1) graph topology synthesis/refinement, 2) joint displacement estimation, and 3) flexure joint modeling (Figure 4). The first two steps solve

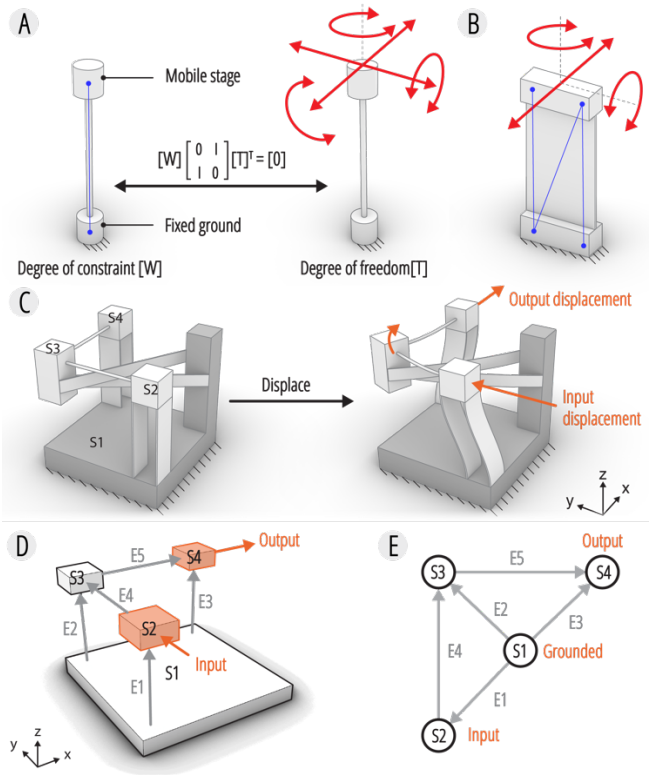

Figure 3: Compliant mechanism design through the freedom and constraint topology (FACT) method. The flexure elements used in this work include (A) rod and (B) blade flexures. A mathematical relation exists between a flexure's degrees of constraint (blue lines) and freedom (red lines). (C) A transmissive compliant mechanism consisting of interconnected joints can transmit displacements, whose (D) topology can be represented and computed as an (E) directed abstract graph. Stages (nodes) and joints (edges) are labeled S1-S4 and E1-E5, respectively.

the problem globally to create the desired transmission function, while the third step models the device joint-wise, independent from one another. In topology synthesis and refinement (steps 0&1), we seek to find and refine a transmissive graph to satisfy the desired transmissions. Only the connectivity and need for intermediate stages are determined at this step. The compliant joints' DOF/DOC is solved in step 2: joint displacement estimation, where we predict how rigid stages should displace in sync to transmit displacements between inputs and outputs. Steps 1 and 2 are repeated several times to reduce a design's redundancy until reaching a simplistic graph topology. At this point, the compliant joints' DOF/DOC requirements are identified, which are then processed using the FACT method to generate procedural instructions to guide users in step 3: flexural joint modeling to create a physical model.

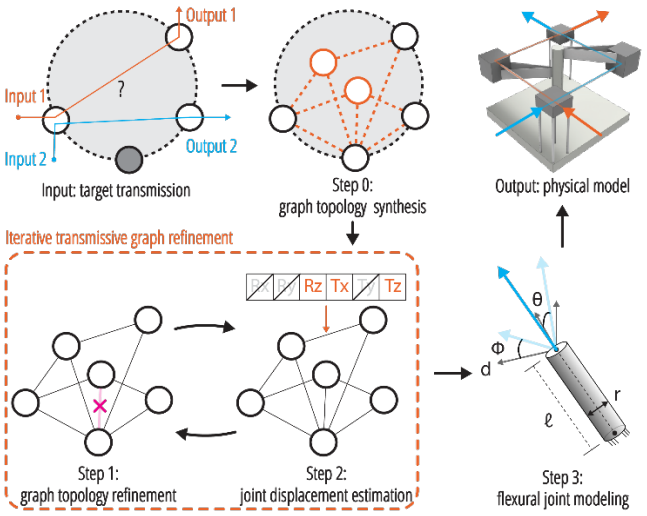

Figure 4: Interconnected compliant mechanism computational design pipeline. The algorithm starts from transmission I/O specifications and an initialized graph topology (top left). The graph topology is then iteratively modified to produce a valid design (bottom left). The resulting joint displacement estimations are parsed into flexure layouts and modeling instructions (bottom right). Users can then follow the instructions to model a physical device achieving the targeted functions (top right).

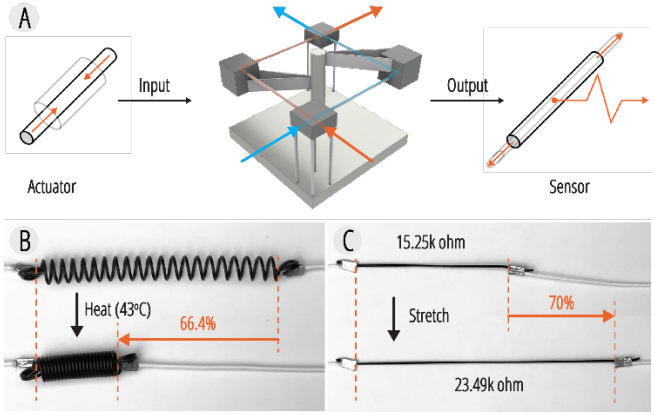

Figure 5: Active material integration in an interconnected compliant mechanism. (A) Actuators and sensors may be considered transmissions' input and output to produce or detect displacements, respectively. Example active materials: (B) shape-memory alloy spring capable of contraction upon exposure to heat and (C) conductive rubber cord sensor that changes resistance when stretched.

4.1 Representation

Transmissive Graph Topology. An interconnected compliant mechanism is represented as a directed abstract graph (Figure 3D, E). The nodes represent rigid bodies within the structure, connected by compliant joints that constitute the edges. The nodes (stages) are prescribed with boundary conditions such as fixed ends or input/output displacements. At the beginning of its design, the graph may be incomplete. I.e., only the I/O and ground stages are known, but their connectivity, including the need for intermediate stages, may be undetermined. Similarly, individual joints' DOF/DOC are also unknown factors that must be solved.

Transmission of Displacements. When describing transmissions, a stage is selected as the fixed, grounded stage with no displacement and serves as the reference frame for all motions. Each of the other stages may be prescribed with an input or output motion. A motion is described by its motional DOF: motional axis direction, position, and magnitude. If a node is not assigned with any displacement, its motion is left undetermined and will be solved by the design algorithm. A device may be prescribed with multiple independent transmission modes, acting concurrently or one at a time. It is possible to prescribe signal amplification using this schema. For example, to double the displacement of an actuation, the output is assigned twice the input's displacement magnitude.

Active Material Integration. In an interconnected compliant mechanism design, the actuators are represented and prescribed as transmission inputs (Figure 5). An actuator is placed between two rigid stages; one is considered the grounded stage, and the other is prescribed with a displacement corresponding to the actuation. This input motion should be specified with desired output motions to create transmission coupling. Similarly, sensors that sit between two stages are modeled as the transmission output, and the relative displacement between the two connected stages indicates how the sensor will be deformed in response to the transmission.

4.2 Iterative Transmissive Graph Refinement

We employ an iterative refinement algorithm (Algorithm 1) to handle topology synthesis and joint kinematics configuration in designing transmissive compliant mechanisms. The algorithm first discretizes the design domain (i.e., the physical space occupied by the interconnected compliant mechanism) into a dense graph network of rigid stages connected by compliant joints, forming the input for design iteration (Figure 1A, B). To iterate the design, the algorithm then interlaces displacement estimation and heuristics-based refinement (Figure 2A). A solver (see Section 7.1: Stage Displacement Estimation) takes the initial topology as input to estimate how rigid stages should displace against each other to achieve the specified kinematic transmission, creating the DOF requirement at each joint. The displacement estimation will also expose topological redundancies and potential problems in the transmission pathway (Figure 2A), such as kinematic decoupling between the input and output stages (i.e., when output stages may move freely regardless of input motions). Several design heuristics monitor and resolve these issues (Figure 1C, see Section 7.2: Graph Topology Refinement Heuristics). The algorithm then re-iterates the modified design until producing a simplified topology and joint DOF/DOC assignment that satisfies the specified transmission functions. The outcomes are then translated into modeling instructions with the FACT method to guide users in modeling the flexural joints (Figure 1D), creating the physical design (Figure 1E).

Algorithm 1 Iterative Transmissive Graph Refinement Algorithm

```
def iterative_transmissive_graph_refinement(domain, goals,
max_iteration):
heuristics = topology_simplification_heuristics +
decoupling_fix_heuristics
graph = discretize(domain, goals) # step 0 of Figure 4 (synthesis)
iteration_count = 0
done = False
do while not done:
iteration_count += 1
 is_modified = False
   displacement = displacement_estimation(graph, goals) # step 2
of Figure 4
   for heuristic in heuristics:
      is_applicable = heuristic.check(graph, displacement)
      if(is_applicable):
         graph = heuristic.refine(graph, displacement) # step 1 of
Figure 4 (refinement)
         is_modified = True
         break
   if((not is_modified) or (iteration_count == max_iteration)):
   done = True
end do
return graph, displacement
```

5 Design Tool Walkthrough

5.1 Design Tool

We implement the iterative design algorithm 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 [8]. As a prerequisite for using the tool, the user should be able to numerically describe motions (i.e., motion axis direction, position, and magnitude) and model flexures. The tool proxies the solution-finding process, but the user needs to interpret and complete the guidance for creating a flexure layout. Hence, the user should understand the parameters constituting a flexure (i.e., orientation, position, aspect ratio, dimensions). The tool also provides visual previews and textual prompts to facilitate understanding.

A user approaches the tool with a listing of the bodies of interest and their input and output motions. The tool helps the user model the initial graph and prescribe the transmission goals, which are subsequently sent to the design solver and iterated with the refinement heuristics until a valid and simple graph topology and joint modeling requirements are found. The tool then generates modeling instructions and procedurally guides the user to model flexural joints using a schema identical to that of ReCompFig [62]. The design tool organizes these steps into different functional tabs (Figure 6).

The design tool uses an abstract visualization to represent the transmission graph topology. Rigid stages are proxied by cubic boxes, and compliant joints are represented as lines connecting rigid stages. We opted for this representation to avoid visually overloading the modeling viewport and to make the topology easier to manipulate. In the joint modeling phase, the user can replace the

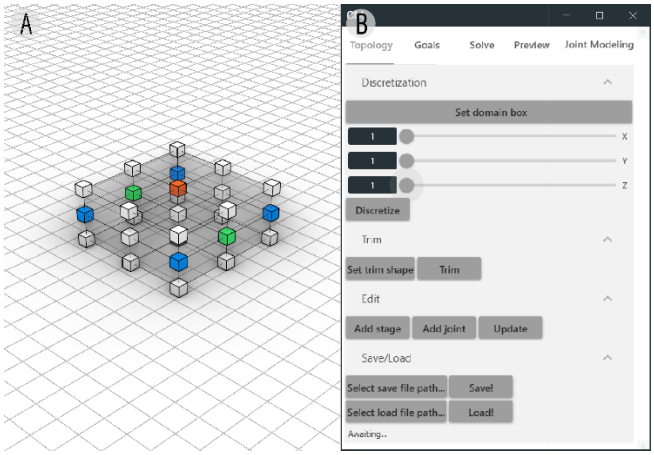

Figure 6: The design tool interface consists of a (A) model viewport and a (B) panel with modeling functions.

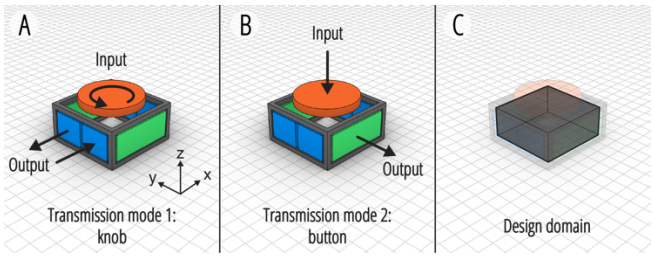

Figure 7: The physical interface design example: (A) knob and (B) button transmission modes, and (C) the specified design domain enclosed by the rigid frame. Color codes – gray: fixed stage; orange: input stage; blue: transmission mode 1 output stages; green: transmission mode 2 output stage. The following figures use a similar color-coding scheme.

rigid stages with custom geometries to better visualize the design, and the tool guides the user to model joint flexures.

We show a button-knob combo to exemplify the tool's workflow (Figure 7A, B). The design has an input stage at the top that can be twisted or pressed. These two motions are prescribed as independent transmission modes; each input action will cause different side plates to displace laterally and transmit their displacements to adjacent units (see Section 6.1: Design Examples: Modular Physical Interface Building Blocks). A coarse model is used to initiate the design problem, which contains the I/O stage positions and a rigid frame that serves as the grounded fixed end. The connectivity between rigid bodies and flexure layouts is left to be solved by the design tool, and the space enclosed by the rigid frame I/O bodies is assigned as the design domain (Figure 7C).

5.2 Phase 1. Constructing Initial Transmission Graph Topology

The design workflow starts with modeling an initial-guess graph. The user may create the topology by (Figure 8A) manually adding rigid stages (nodes) and compliant joints (edges) or by discretizing

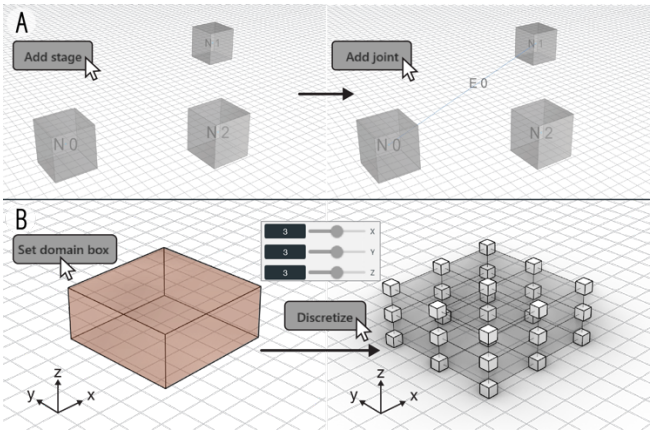

Figure 8: Topology initialization: (A) manual topology modeling and (B) topology initialization with domain discretization.

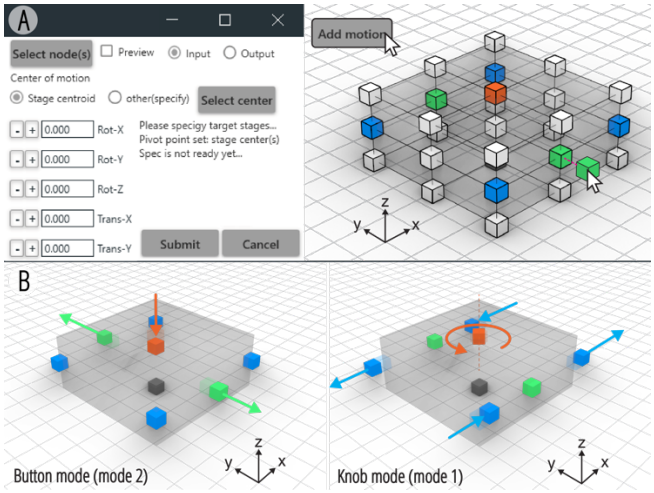

Figure 9: Goal displacement modeling: (A) goal modeling panel and adding transmission I/O to rigid stages by specifying their axis and magnitude. (B) shows the resulting transmission mode assignments for the knob-button device, including a button press driving y-axis-facing side plates to translate along the y-axis, and a knob twist driving other side plates to translate along the x-axis.

a design domain (Figure 8B). 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 8B) to establish the graph. Once discretized, the user may drag and move stages around to further modify the topology, and the edges are updated accordingly. Users may also manipulate the topology at any point in the design tool—even while using the iterative solver. The design tool also provides functions to export and import design models as a .json file.

In the button-knob example, we initialize the graph by discretizing the design domain. We use a 3x3x3 resolution for the design problem without knowing what topology is needed for the transmission design. This density provides the minimal resolution required for representing the rigid bodies of interest, and the topology will be simplified in the later steps.

5.3 Phase 2. Prescribing Target Transmission Displacements

The next step pertains to prescribing design goals over elements in the graph. The user should specify the number of transmission modes needed for the design. The design tool then populates the interface with modeling panels for each transmission mode. In the panel, the user should specify a fixed stage for a transmission mode, and a pop-up window (Figure 9A) will prompt the user to select the scoped nodes, kind (i.e., input or output), and displacement. This step can be repeated several times to prescribe multiple I/O to a transmission mode. The design tool also visualizes the prescriptions for preview, but only the prescribed stages' motions are captured.

When designing the button-knob, the two transmission modes share the same grounded stage. The bottom-center stage serves as a proxy for the rigid, fixed frame. The top-center stage is selected as the transmission input and assigned with corresponding motions. The stages on the middle layer are designated as the transmission output, and each transmission mode drives different stages to translate outward along the axis they face (Figure 9B).

5.4 Phase 3. Refining Transmissive Graph

With the initial graph topology and transmission goals, the design tool helps the user apply the iterative algorithm and produce a valid design (Figure 10A). Several indicators communicate the heuristic evaluations in the solver panel. A green light indicates that a heuristic does not apply to the current design, a yellow light suggests an opportunity to apply a topological simplification heuristic, and a red light indicates a transmission decoupling issue that must be addressed to produce a valid design.

The tool provides two design interactions to refine a transmissive graph topology - automation and manual/suggestive editing. The automation mode requires no user input but runs on a pre-defined logic agnostic of the design's contextual needs. Conversely, in the manual mode, the user has complete control over modifying the graph topology with the design tool's visual and textual hints. Still, all decisions must be made by the user, and it can be overwhelming in early design iterations where the topology is complex. The suggestive mode compromises between the two modes and allows the user to choose from suggested changes to iterate the graph topology. This way, the designer can co-steer and modify the design with the tool's scaffolding. The user may seamlessly switch between these design modes at any point, enabling flexible workflows.

5.4.1 Automation Mode. In the automation mode, the user leaves the tool to finish a design task without manual input. The user may specify the tool to run for certain iterations or indefinitely until the design is completed. The solver will terminate early if it cannot simplify the design further and finds no transmission decoupling issues. The exit status is then textually communicated to the user. All heuristic indicators should turn green in the finalized design, signaling no further changes are needed.

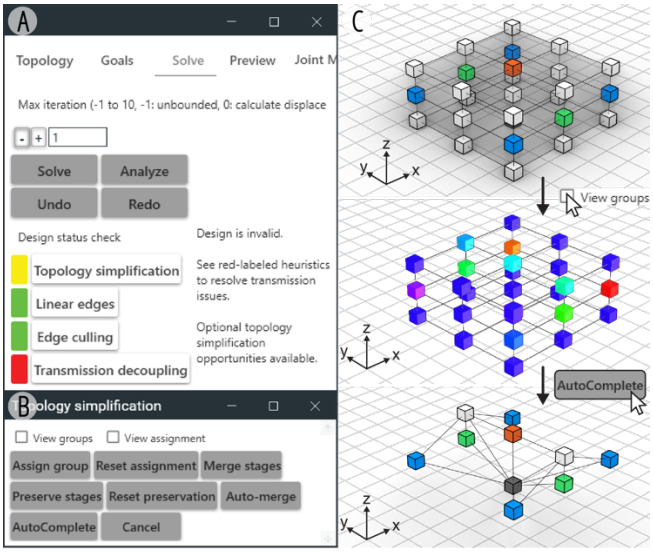

Figure 10: Graph topology simplification workflow. (A) The user clicks on the "Topology simplification" heuristic to bring up (B) a pop-up menu that helps to formulate a refinement. (C) The user previews the solver-generated group assignment and accepts the change. The "view groups" option color-codes the stages into corresponding groups to be combined.

5.4.2 Manual or Suggestive Mode. To collaboratively work with the heuristic engines, the user should click the "Analyze" button to generate a displacement estimation. The design tool then checks the graph topology and displacement estimation against each heuristic to generate their status report and refinement suggestions. The status report updates the indicator signals, and the user can click on a heuristic to formulate a refinement in a pop-up menu, as introduced below.

Topology Simplification. If a design presents an opportunity for merging synchronized stages (i.e., stages sharing the same movement, resembling a rigid connection in-between), the pop-up panel (Figure 10B) will render the stages in color-coded groups to communicate a potential merge plan (Figure 10C). If the user decides to merge stages manually, the tool will double-check the assignment to ensure the selected stages are indeed synchronized, or it will prompt the user to correct them. Alternatively, the user may flag stages for exclusion from merging, and the tool will attempt to merge synchronized stages accordingly. Lastly, should the user be satisfied with the suggested merge plan, they may also click the "AutoComplete" button to apply the merge. The linear arc and saturated joints heuristics provide the same interaction to allow the user to simplify the topology manually or suggestively (See Section 7.2 for heuristic details).

Decoupling Fix. If a decoupling issue (i.e., when the displacement flow between inputs and outputs is interrupted) is found, the design tool will highlight the decoupled edges and nodes for the user to examine (Figure 11A). Such issues are fixed by assigning a biasing displacement to decoupled edges or nodes, which forces them to displace in certain ways to reestablish displacement flows.

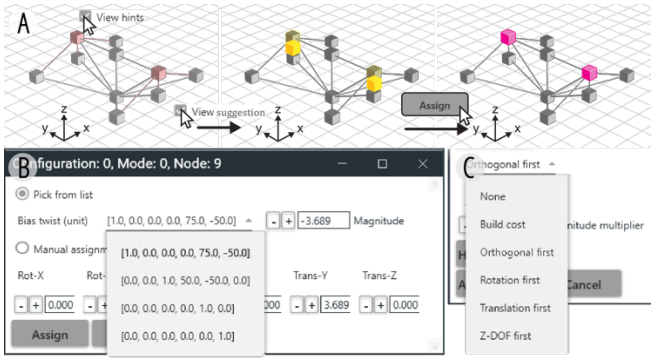

Figure 11: Decoupling fix workflow. (A) The tool helps the user inspect decoupling sites and suggests a biasing displacement. (B) The pop-up menu presents potential fixes or allows the user to specify a manually formulated fix. (C) The drop-down menu to alter the design tool's priorities in picking a fix.

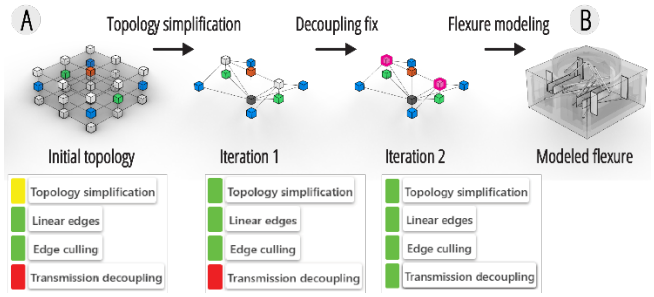

Figure 12: Button-knob design iteration history: (A) design iterations and (B) modeled flexure layout. Top row: graph topology changes across iterations. Bottom row: solver heuristics check result.

By default, the user may allow the design tool to "Auto Complete" the design using its built-in heuristics or optionally alter the heuristic engine's priorities when formulating refinements (Figure 11C, see also Section 7.2). Users attempting to fix the decoupling issues manually can pick from the design tool's suggested biasing displacements or manually specify one (Figure 11B) to supersede the heuristic engine's decision.

Both automated and suggestive modes are used to create a valid knob-button design (Figure 12A). In the first iteration, the user accepts the changes suggested by the heuristics engine to merge synchronized stages. In the subsequent step, the design tool identifies a decoupling issue in the modified graph topology. The user then examines the biasing options and picks a biasing displacement for each transmission mode-node combination from the design tool's suggestions.

5.5 Phase 4 Transmission Preview

A separate tab allows the user to visualize the stages' displacements as estimated by the solver (Figure 13A). The estimations are not limited to the finalized design. Users may use this function to preview

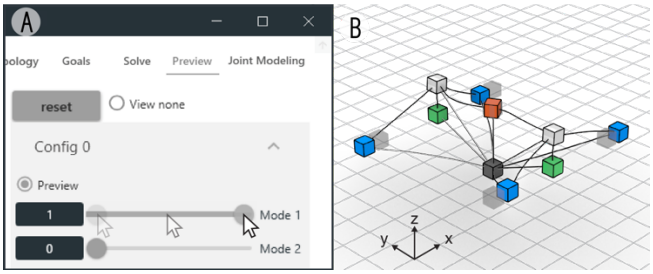

Figure 13: Transmission preview. The rigid stages are visualized using proxy boxes. However, they could be modeled into any shape in the later step. (A) The panel for previewing and combining modal displacements. (B) Previewing the knob mode's displacement, where the top stage rotates in place and the side plates shaded in blue translate laterally.

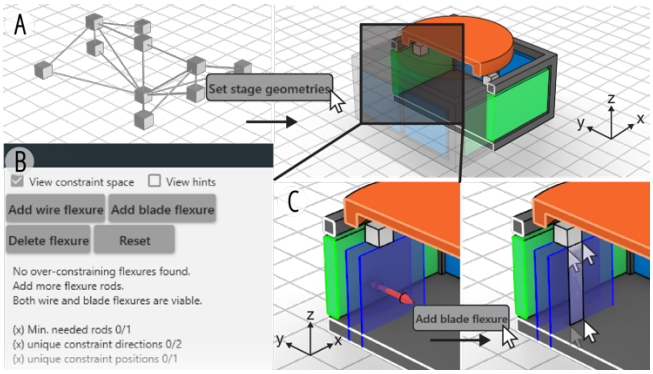

Figure 14: Flexure Modeling. (A) In this example workflow, the designer first replaces the rigid stage proxy boxes with the coarse knob model. (B) A pop-up window communicates the targeted flexural joint layout for each joint. (C) The designer then models the flexures (i.e., blades in this case) according to the procedurally updated information.

an intermediate design's displacements to inform their decision. A slider lets the user preview each transmission mode's motions from 0% to 100% displacements (Figure 13B). Multiple transmission modes may be actuated and visualized simultaneously to preview how they would behave in combination. We note that this preview is purely kinematic and does not consider collision, flexure dimension, and material limits. These factors will necessitate computationally expensive finite element methods, compromising the design tool's real-time interactivity. Still, future implementations may incorporate these features.

5.6 Phase 5. Flexural Joint Modeling

Once the solver checks and validates a design, the tool helps the user model the flexural joints constituting the transmissive graph topology. The tool parses the solver output (displacement estimations) to generate modeling instructions per joint (Figure 14). These instructions are organized into panels to help users tackle one joint at a time. Under each panel, the tool provides visual and textual prompts to inform flexure placements (Figure 14B). Users may use flexural rods or blades (i.e., thin sheet flexure) to create the needed flexure layout (Figure 14C, see Figure 12B for the completed flexure layout), and the prompts are updated in real time. Note that designers may also replace the rigid stages with custom geometries to better iterate and visualize the design (Figure 14A). 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.

6 Design Examples

The devices demonstrated in this section are designed with the computational tool and prototyped with an off-the-shelf 3D printer (Ultimaker S5 with PETG or PC filaments), active material actuators (Nitinol shape-memory alloy spring, Fuxus), and sensors (Stretchable Rubber Cord Sensor, Adafruit). The actuators are activated at 43℃ and are safe for continuous skin exposure for no more than an hour [25].

6.1 Modular Physical Interface Building Blocks

We use the design tool to create three physical interface building blocks: knob, trigger, and button (Figure 7, Figure 15A-B, Figure 16). The input element is located at the top side of the blocks, and the internal structures transmit the input motions to drive side plates to translations. The knob transmits an input rotation or press into different side plates' movements, creating two concurrent transmission modes (Figure 17A). The trigger and button are designed with a singular transmission mode and I/O pair (Figure 17B-C). These building blocks can be joined with an actuator-sensor hub (Figure 15C) via magnets on the side plates to render haptic feedback (Figure 17D) and detect motions (Figure 17E). Given that these designs are relatively simple, the design tool can automate most of the heuristic decisions (Figure 12, Figure 16) to produce a satisfactory, viable design.

The interface building blocks have modular dimensions that allow recomposition into different form factors depending on the interaction context (Figure 17F). Users may disassemble an assembly of building blocks and rearrange them into a new form factor suited for a different use case (see supplementary video). Compared to conventional electromechanical interfaces, these devices may be quieter to operate. The absence of mechanical articulations and motors makes these devices lightweight and compact. In addition to these building blocks, the design tool also enables future users to easily create more block types, potentially enabling a low-cost prototyping toolkit. In particular, the most complex unit – the knob – costs less than $2.5 USD in material and weighs shy of 30 grams. The active materials in the actuator-sensor hub cost around $14 USD ($13 USD shape-memory alloy spring, $0.3 USD rubber cord sensor) and are reusable. Anecdotally, this design example also shows that topologically and functionally complex devices may be created by modularization.

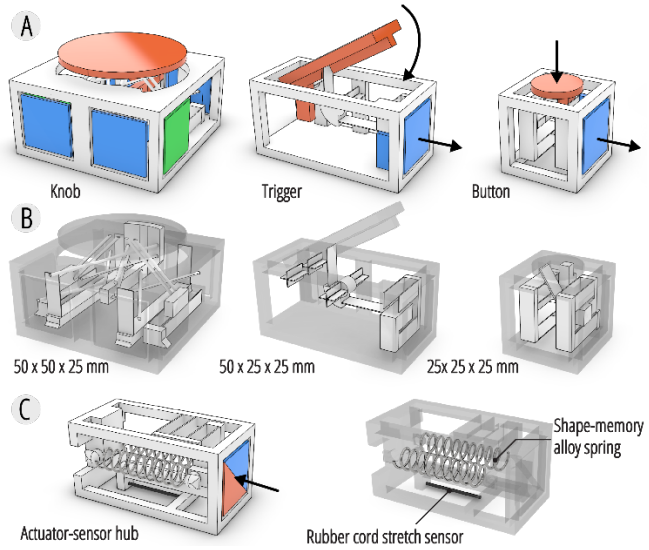

Figure 15: Modular interface building block design: (A) different types of building blocks, (B) the devices' dimensions and internal structure designed with the tool, and (C) the actuator-sensor hub.

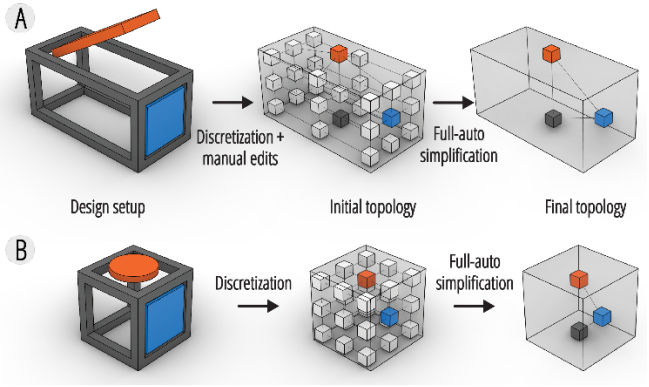

Figure 16: The interface building block design iterations: (A) the trigger and (B) the button. From left to right: initial coarse model, initial graph topology, simplified (final topology).

6.2 Smart Lock Mechanism

In this example, we use the design tool to create a lock mechanism wholly made of a compliant mechanism and only unlockable by manipulating the handle in a correct sequence of actions. The lock consists of an input handle, a lock pin, and a latch (Figure 18A). When closed, the latch and lock pin extend into the lock block, mechanically preventing it from unlocking. To unlock the mechanism, the input handle should be pushed upward to disengage the lock pin from the hook, followed by (Figure 18B) rotating the handle downward to move the lock pin into a clear passageway. The lock is then opened by pushing the handle forward to retract the latch and lock pin. A stretchable sensor is attached between the fixed frame and the latch stage. The sensor is stretched when retracting

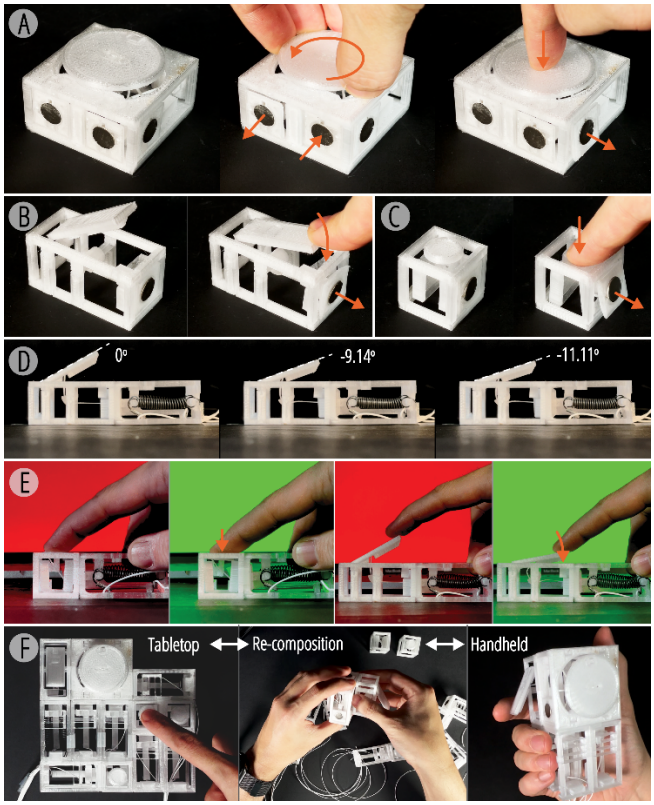

Figure 17: Modular physical interface building block prototypes: (A) knob, (B) trigger, and (C) button. (D) When connected with an actuator-sensor hub, the shape-memory spring contracts to provide force and displacement feedback at the input stage. Similarly, (E) the sensor is stretched when displacements transmit through the side panel, triggering events (changing background screen color). (F) The modular building blocks can be reconfigured into different form factors depending on the interaction context.

the latch, causing a resistance change that a microcontroller can detect to trigger further events.

The device is designed with an initial sketch model to establish the design domain and bodies of interest (Figure 19). Three transmission modes are specified to create the design problem, each creating a pin and latch movement in Figure 18B. The flexures are modeled according to the design tool's instructions, and the rigid bodies are modeled to connect between flexures and provide the intended function. The physical lock pin and latch movements can be seen in Figure 20. Qualitatively, our design tool enables the design of more complex behaviors than metamaterial mechanisms [21, 23, 24]. The door latch in [21] is limited to planar motions and a singular transmission mode. In contrast, this example demonstrates 3D motions (i.e., out-of-plane X-rotation and Y-translation) and concurrent transmissions, creating computational behavior where the output bodies' motions are conditioned on the input's displacement. Such a design affords an "encrypted" tangible interaction with the user: the latch may only be unlocked by users who

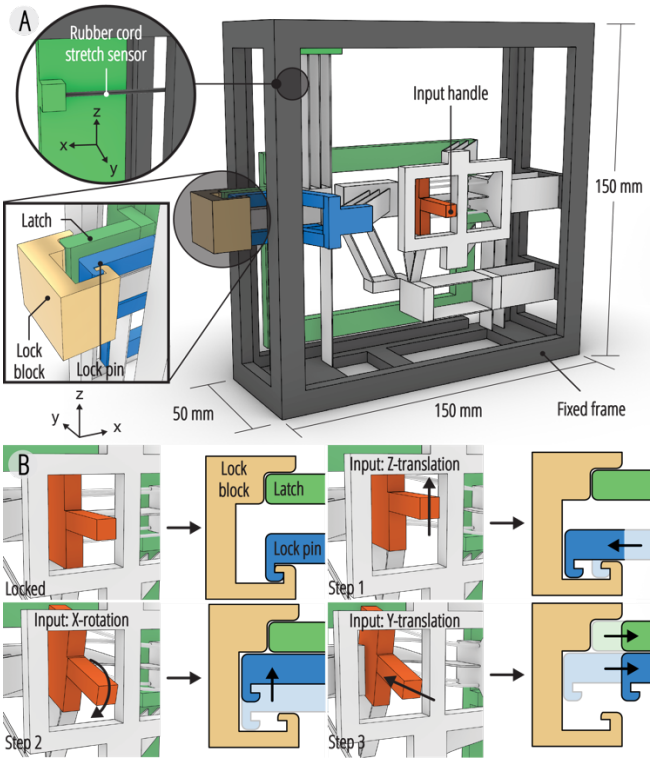

Figure 18: Smart lock mechanism design. (A) The lock comprises a transmissive structure and a sensor. The latch and lock pin extend into a lock block, preventing it from opening. (B) The lock may only be disengaged by moving the handle in three consecutive steps.

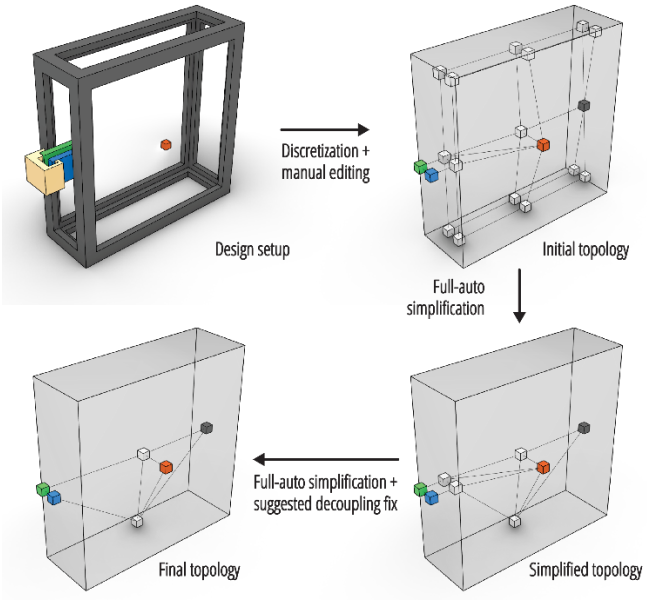

Figure 19: Smart lock design process. The design tool is used to discretize a space inside of the fixed frame to produce a $3 \times 2 \times 3$ rigid body array. The latch and lock pins are manually added to the graph topology. The input handles' position and connectivity are also adjusted in this step. The design tool suggests topology simplification in the first two iterations, which the designer accepts, and the tool is allowed to automate the decisions. At the third iteration, the design tool identifies a transmission decoupling issue. The user resolves this problem by picking from the tool's suggested changes.

know about the unlocking sequence (Figure 20). We speculate that such features may enable the community to create tangible security and mechanical computation.

6.3 Wearable Haptic Augmentations

The algorithm and design tool enable the customization of devices that augment the wearer's ability to interact with objects. In this example, we contextualize in upper-body motor impairment [31] and presume a user with limited hand motor functions (i.e., no finger mobility and wrist extension; capable of wrist flexion). A device is designed to be worn at the right hand to proxy a grasping motion with wrist flexion as input (Figure 21, Figure 22A). Flexing the wrist (i.e., rotating the hand toward the palm) will stretch the sensors, changing their resistance (Figure 21B, Figure 22B). A controller monitors this change and relays current to the shape-memory springs, causing them to contract and close the grasper (Figure 21C, Figure 22C). Compared to directly transmitting the input motion into the output, such a modulated device can allow users to use their dexterous muscles to generate a signal, and the actuator proxies the output to generate the required forces (Figure 22D).

In this design, the input and output motions are modeled as individual transmission modes. While the tool is allowed to automate most of the decisions, we more frequently edit the graph topology to prevent generating stages and joints that collide with the hand (Figure 23). The final design weighs 108.14 grams and costs less than $34 USD in material. The rigid stages contribute to most of the device weight, and we speculate that switching to a material with a higher specific strength (i.e., elastic modulus over density) and topology optimization [3] may further reduce its footprint. Other than the accessibility scenario, we also foresee that similar exoskeletal devices may enable proxying and simulating kinematic experiences of a different skeletal frame (e.g., downscaling the wearer's finger motion range [44]), and the provided design tool will enable interaction designers to customize devices with varying kinematic transmission and user factors.

7 Algorithm Details

This section provides a conceptual description of the components behind the iterative design algorithm. We note that users may use the design tool without a strong understanding of the mathematical principles, and this section provides a conceptual introduction to guide (re)implementations and facilitate understanding the design

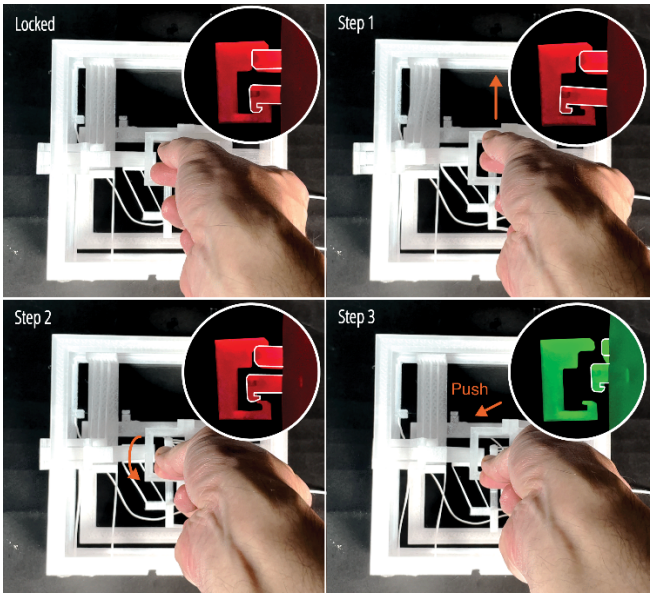

Figure 20: The smart lock mechanism unlocking sequence taken from a physical prototype. Inset images show a rubber cord sensor detecting an unlocking event at the final step to change the ambient light color from red to green.

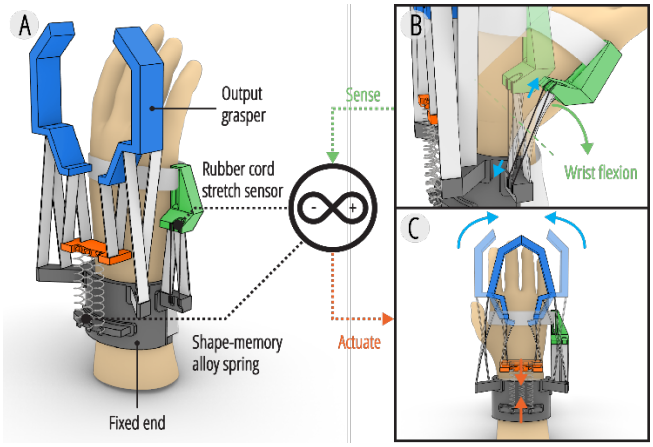

Figure 21: Wearable device for user augmentation: (A) device schematics, (B) input motion, (C) output motion.

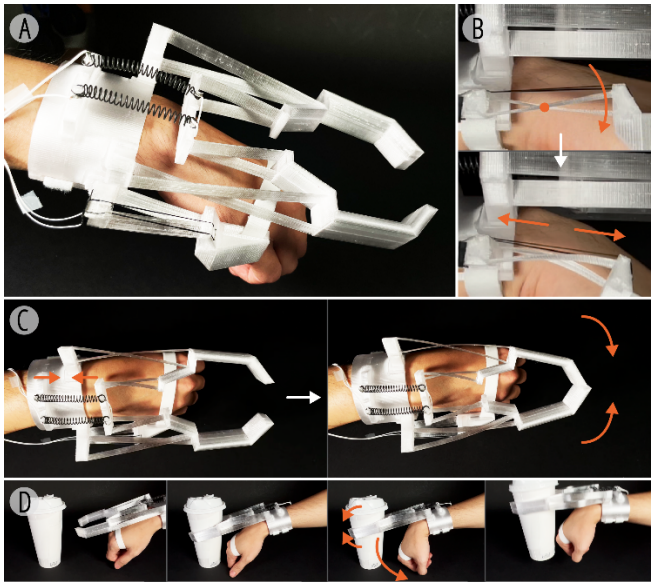

Figure 22: Wearable grasper prototype: (A) Overall view, (B) sensor extension by wrist flexion, (C) closing grippers by actuator contraction, and (D) using the device to grasp and manipulate objects.

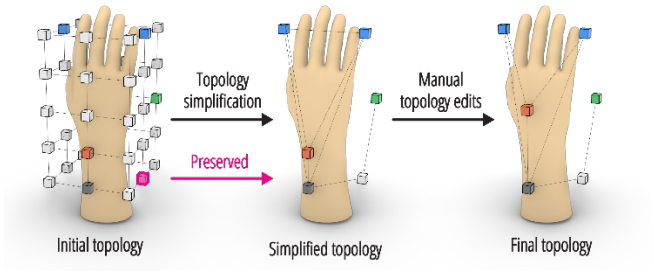

Figure 23: Wearable grasper design iteration. The space around the hand is discretized into a 3x3x5 network of rigid stages. The stages colliding with the hand are removed to create the initial graph topology. The design tool suggests and automates topology simplification, but a stage is preserved to prevent the joints from collapsing and colliding with the hand. Next, the graph topology is manually edited to allow more actuator space and to remove a joint between the output bodies. The tool identifies no decoupling issue.

tool's interface options. Detailed mathematical models are provided in the supplementary materials. In the following part, we will explain the algorithm using the simple transmission design shown in Figure 3C-E as an example. The matrices representing its abstract graph topology can be found in Figure 24.

7.1 Joint Displacement Estimation

Given a graph topology, the joint displacements required to satisfy a transmission are found by modeling and solving several constraint equations. The constraint equations pertain to the graph topology's inherent connectivity and the I/O bodies' target displacements. These constraint equations concatenate into a linear system of equations, which allows us to numerically solve how joints in the topology should displace to carry out transmissions. Below is a summary of the joint displacement estimation algorithm. See Appendix A.8 for the algorithm's mathematical formulation.

7.1.1 Input. The transmissive graph topology is defined by its signed adjacency and incidence matrices, as well as the stages centroids and a pivot point per compliant joint. Multiple (>=1)

Figure 24: The graphical representation of the interconnected compliant mechanism shown in Figure 3C-E. The rigid stages and compliant joints' connectivity are represented by a (A) directed adjacency matrix $[J]$ and an (B) incidence matrix $[C]$, respectively. The entries in the matrix indicate the direction of connection – 0: not connected; -1: sender of connection; 1: receiver of connection.

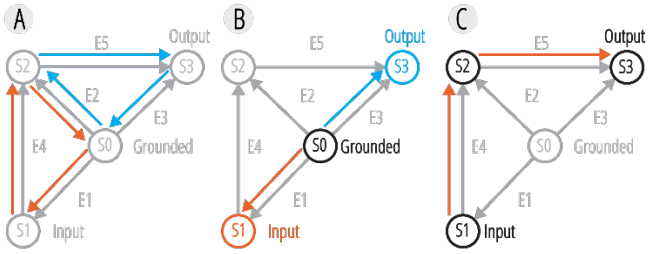

Figure 25: Interconnected compliant mechanism design constraints using the specification and graph topology in Figure 3C-E as an example: (A) kinematic loop constraints, (B) stage mobility constraints, and (C) transmission constraints.

kinematic transmission modes may be prescribed to the transmissive graph, each defined by setting one stage (node) as the fixed end and a set of I/O prescriptions. The inputs and outputs are defined by a target stage index and a displacement (as a DOF twist vector). See also Appendix A.1: Topology Representation.

7.1.2 Step 1. Model joint freedom space. For each joint in the graph topology, model their possible displacement DOF as screw vectors using their pivot points (i.e., the midpoint between adjacent stages). These available DOF may be linearly combined to produce a displacement, which will be calculated later. See also Appendix A.3: Joint Modeling. When a joint's DOF is undetermined, we assume all six DOF are available, creating the full freedom space.

7.1.3 Step 2. Modal kinematic constraints. For each transmission mode, construct three types of kinematic constraint equations. The constraints accumulate displacements along connected edges in the graph, and the summed value should equate to a targeted kinematic relation between the starting and ending stages. See also Appendix A.5: Constraint Modeling.

- **Kinematic loop constraints**. A transmissive graph topology often contains multiple closed loops (Figure 25A). The displacements along the edges in a loop must sum to zero, which indicates that the stages do not displace against themselves and obey the rigid body assumption (Figure 25A).
- **Stage mobility constraints**. The displacements accumulated from the fixed stage to an input or output should equate to the prescribed motion (Figure 25B).
- **Transmission constraints.** For each input and output pair, the accumulative displacement between the transmission pathway should equate the difference between them (i.e., the output's displacements minus the input's, Figure 25C).

7.1.4 Step 3. Model problem linear system. The kinematic constraints and joint freedom spaces are composited into a system of linear equations per transmission mode. The linear system readily describes all desired kinematic relations and the topology's intrinsic properties. See also Appendix A.2: Numerical Model, A.4: System Kinematics Modeling, and A.6: Constructing Linear Systems.

7.1.5 Step 4. Solve the design problem. A transmission estimation that requires a small number of joint DOF is found by solving the linear system using L1 minimization. If the design problem involves multiple transmission modes, the minimization target may be adjusted to promote reusing joint DOF across transmission modes. See also Appendix A.7 Solving Design Requirements.

7.1.6 Step 5. Identify the required DOF.. A joint's mobility requirement (i.e., DOF) to enable a transmission mode is found by analyzing the corresponding DOF velocities found in the previous step. In principle, a design problem involving multiple transmission modes will produce one DOF requirement per mode and joint. See also Appendix A.7 Solving Design Requirements.

7.1.7 Output. The algorithm outputs each joint's mobility requirement as a collection of DOF vectors describing their displacement under each mode. These vectors are used to visualize how rigid bodies are displaced under a transmission mode. A joint's corresponding DOF vector(s) forms a freedom space that is used to generate flexure modeling instructions using the FACT method [16, 17] and ReCompFig's single-joint modeling tool [62].

7.2 Graph Topology Refinement Heuristics

The design refinement heuristics fall under two categories: topology simplification and decoupling fix. The order of applying these rules is deliberately structured to prioritize topology simplification over decoupling fixes. The former eliminates redundant joints and stages in the early stage, reducing computation time while decimating the design problem's complexity. We note that Topology Simplifications are optional in producing a valid design, but Decoupling Fix is mandatory. Each heuristic has a check to evaluate its applicability and a refine function to alter the graph topology. See Appendix A.8: Topology Simplification Heuristics and A.9: Decoupling Fix for more information and visual examples.

7.2.1 Topology Simplification: Merging Synchronized Stages. Based on the displacement estimations, adjacent stages sharing the same displacement can be optionally merged into a single body. These synchronized stages have no relative displacement, resembling rigid connections. Consequently, these stages could be combined into a rigid body without compromising the transmission functions.

7.2.2 Topology Simplification: Collapsing Linear Arcs. A linear arc is a subset of serially connected edges within a graph. In graph-based kinematics analysis, these arcs have identical design implications as a single joint: the DOF/DOC of a serially connected arc can

be equivalently established by a single compliant joint [18]. Thus, they can be combined to simplify the graph topology.

7.2.3 Topology Simplification: Removing Saturated Joints. An edge within a transmissive graph topology may be removed if all six DOF are used. In this case, the edge must afford zero DOC between the connected stages, leading to no legal flexure placement. Hence, the joint could be removed.

7.2.4 Decoupling Fix. The input and output stages' motions are decoupled when the input can be displaced independently without incurring displacement of the output node. This issue often occurs in the transmission pathway between input and output nodes, where an intermediate stage is displaced in specific ways to cancel out the flow of motions. To resolve this issue, a biasing displacement should be prescribed to the decoupling site to re-establish the displacement flow.

8 Numerical validation

8.1 Test Setup

We use finite element analysis (FEA) to validate the effectiveness of the algorithm and design tool. Unlike physical experiments (e.g., using Instron machines), FEA enables the simultaneous setup of various load conditions and measurement of multiple output displacements, providing a repeatable method for analyzing transmissive compliant mechanisms with multiple I/Os. Although slower to compute, FEA offers accurate evaluations and is often used as an alternative to physical prototyping in mechanical engineering to evaluate complex mechanical systems.

Two devices are designed using the tool (Figure 26), with rigid stages modeled as simple geometries to maintain generalizability. The flexures are modeled according to the tool's recommendations. We validate the designs using Ansys's static structural analysis with isotropic material settings (Ultimaker Polycarbonate, elastic modulus: 2579 MPa, Poisson ratio: 0.3). In the tests, displacement loads are applied to the input stage, and the output displacements are measured against design specifications.

8.2 Targeted Transmission

The first device verifies whether the generated design carries out the specified transmission functions (Figure 26A). It converts the input stage's X-translation into Y-translation and Z-rotation at two distal stages (Figure 27A, B). The results (Figure 27C) show good agreement with the transmission target. Both output stages are displaced along the targeted DOF with at least an order of magnitude greater motions than the unwanted ones. However, due to compliant mechanisms' inherent nonlinear behaviors, the output motions were not consistently proportional to the input—an expected outcome for mechanisms with large deflections (e.g., >10% of flexure length; our examples have 20%) [19, 55]. Still, minimal displacements occur along non-targeted DOFs, indicating that the device effectively minimizes undesired transmissions.

8.3 Concurrent Transmission Modes

The second device verifies the tool's ability to generate designs affording multiple simultaneous transmission modes (Figure 26B).

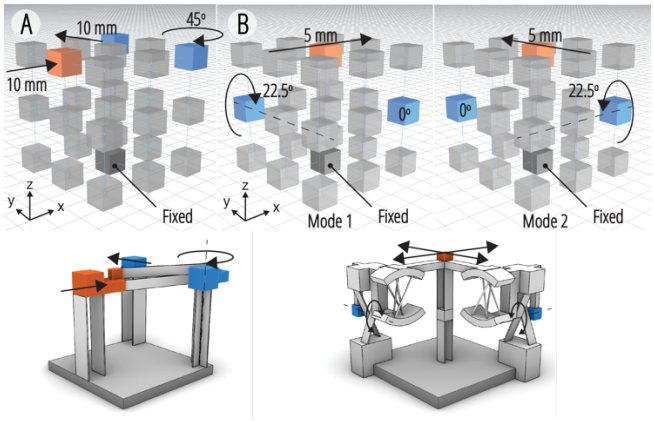

Figure 26: Devices for validation. Images show design specs (left) and the modeled test samples (right). (A) A simple device consisting of one kinematic input and two outputs. (B). A dual-transmission device consisting of one kinematic input and two potential outputs. The input's XY-translations drive separate outputs.

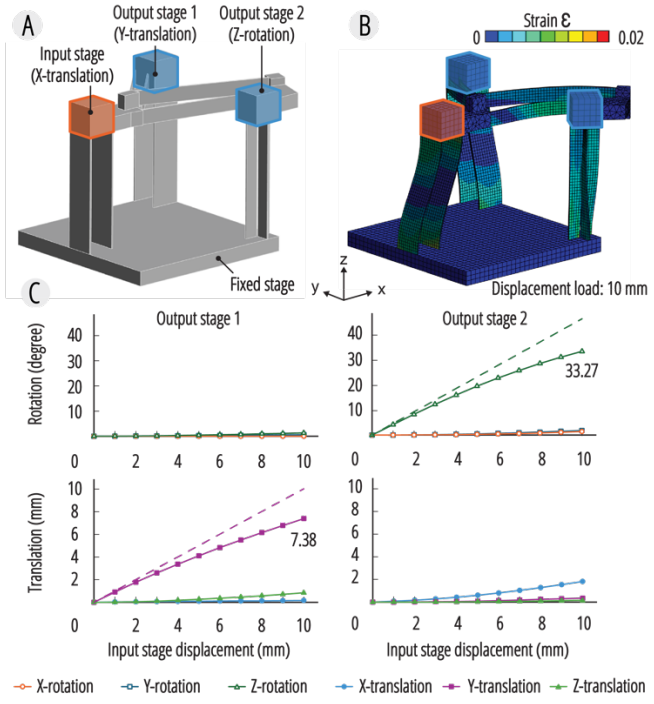

Figure 27: Simple IO device simulation result - (A) test setup, (B) visual results, and (C) measured displacements along each DOF. Dashed lines in (C) show the expected displacement curve. Solid lines show simulation results.

This device maps two independent translations (X & Y) at the input stage to corresponding rotations along the other axis, forming two transmission pathways (Figure 28A). An X-translation of the input stage triggers a Y-axis rotation of output stage 2, while a

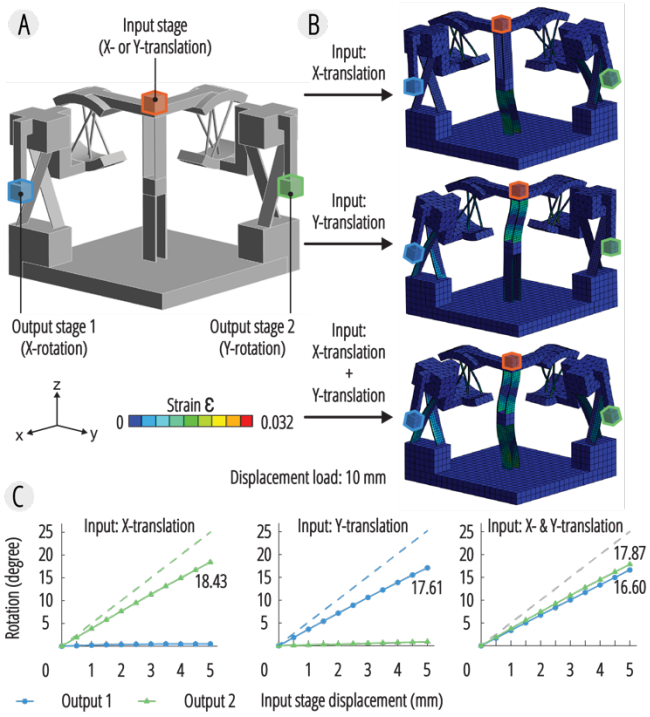

Figure 28: Double IO device simulation result - (A) test setup, (B) visual results, and (C) measured output stage displacements. Each row shows a different input condition; from top to bottom, the input stage moves along X-translation, Y-translation, and both translations simultaneously. Dashed lines in (C) show the expected displacement curve. Solid lines show simulation results.

Y-translation triggers an X-axis rotation of output stage 1. The transmission modes can be activated independently or in combination, depending on the input stage's displacement DOF (Figure 28B). Results indicate that the transmission modes are effectively decoupled: when one transmission mode is activated, the other output stage remains nearly stationary (<0.87°, 5.1% of the mobile output stage's displacement). When both transmission modes are activated simultaneously (i.e., diagonal movement along the X-Y plane), both output stages move with comparable magnitudes, demonstrating that the device can concurrently support multiple independent transmissions.

9 Discussion AND Design Guidelines

9.1 Topological Complexity

The topological complexity needed for a design depends on the transmission specifications. Generally, the more required transmission modes and varying displacements, the more compliant joints and stages are needed. As a guideline, there should be more edges than concurrent modes (e.g., one joint for unimodal designs, two joints with one intermediate stage for bimodal designs). An intermediate stage between I/O nodes can compound displacements, enabling multiple *modal* displacements to arithmetically recombine.

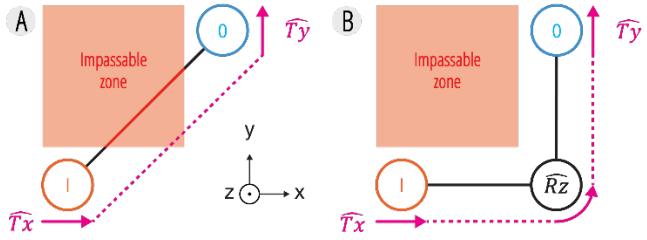

Figure 29: Using intermediate nodes to circumvent impassable spaces. (A) The edge between the input (X-translation) and output (Y-translation) trespasses the designated impassable zone, invalidating the design. However, (B) the impassable zone can be circumvented by rerouting the gear train via an intermediate stage (Z-rotation) to redirect the displacements.

Additional intermediate stages may also be useful in redirecting transmissions around corners and limited spaces between I/O stages (Figure 29), such as the wearable device example Figure 23.

9.2 Scalability

CompAct design can potentially scale beyond the mesoscale (centimeter scale) demonstrated here. Compliant mechanisms are commonly used at smaller scales, such as in micro-electromechanical systems [59]. However, fabricating smaller-scale devices presents challenges in fabrication resolution due to the flexures' slender geometries. At larger scales, compliant mechanisms are used in automobile springs [64] and building structures [45]. Nonetheless, it is worth noting that scaling up CompAct devices will introduce a challenge in material selection. Flexures need to be thickened to withstand the device weight and interaction forces that scale cubically with size, but their stiffness also scales quartically with increased dimensions, presenting a tradeoff between robustness and compliance. Hence, when scaling compact designs to a meter scale, using materials with higher elastic moduli may be more viable. As an example, steel is 100x stiffer but only 7x denser than polycarbonate plastic, making it possible to keep flexures slender and compliant without much slacking to its own weight.

9.3 Kinetics Design

The design tool presented in this work is focused on kinematics design, relating only to a device's displacement behavior. The flexure layout recommended by the design tool maximizes compliance for the targeted motions and stiffness for the undesired ones. Still, certain load-sensitive application scenarios, such as rendering precise haptic feedback or using small-force actuators (e.g., thermoplastic residual stress [63]), may require considering device stiffness. In principle, the longer and thinner a flexure is, the easier it is to displace it. A flexure's dimension may also affect its robustness as the fatigue threshold is a function of the cyclic load. The higher the strain, the less robust a flexure is (failing with fewer load cycles). Moreover, the maximum strain in a bent flexure increases with thickness [11]. Hence, it is advised to keep flexures as thin as possible to avoid premature flexure failure. Anecdotally, this work's

devices made from polycarbonate with a max strain no higher than 5% could withstand hundreds of load cycles during test runs and documentation, in line with that suggested by literature [20]. The devices are also robust against haptic interactions. The modular interface (Figure 17) and smart doorknob (Figure 20) are made with 1 mm diameter rod flexures and 0.2-0.3 mm thick blade flexures, and they could sustain typical physical interface manipulation forces in the range of 25N [69]. Still, the flexures are susceptible to rupture or permanent deformation if exposed to excessive forces. Future works may consider combining the CompAct tool with a screw algebraic kinetic model [55] to generate, iterate, and optimize the flexural layout and geometry design, potentially achieving end-to-end device design automation for both kinematics and kinetics.

9.4 Mechanical design limitations

This work builds on a conventional flexure design and shares similar limitations (e.g., limited motion range, no continuous movement). While these devices can transmit displacements between distal parts, they cannot threshold signals or decouple I/O motions to create digital behaviors like those of Digital Metamaterial Mechanisms [21]. For instance, an AND gate with inputs of 1 and 0 should output 0 (where 1 indicates a displacement and 0 indicates no displacement), a condition that inherently involves I/O decoupling and is incompatible with the current algorithm. Future work may explore using contact-aided compliant mechanisms [36, 43, 54] for signal thresholding. This enables displacements to transmit only when rigid stages move past a certain threshold and come into contact with another rigid body. Incorporating stiffness-changing materials [67] may also allow stimuli-responsive transmission reconfiguration (e.g., buckling under displacement when softened, transmitting when hardened), enhancing a device's versatility and adaptability. Anecdotally, we also note that joints that require a screw motion (i.e., simultaneously rotating about and displacing along the same axis) to transmit forces usually have smaller motion ranges and higher stiffness. Thus, users should avoid creating designs where two adjacent stages rotate and translate along the same axis to increase a device's mobility and compliance.

9.5 Design Tool Interaction

The design tool provides multiple modes to refine a CompAct design. Anecdotally, we found that automation effectively reduces the graph topology complexity in the early iterations, removing redundant components to make the graph topology easier to manipulate and reason. The manual mode allows users to fix design issues that arise from the context-agnostic algorithm, as seen in the wearable device design example, for preserving rigid stages to avoid collision. Still, making effective mechanical design changes may require certain design intuition or demonstration [11], and we speculate that the options and complexity in the manual and suggestive mode may overwhelm less experienced users. Moreover, in manual and suggestive modes, the tool does not assess or communicate a modification's long-term impact on the final design. If the design grows complex or over-constrained, the user may also become frustrated with managing and assessing their decisions to iterate the design. Still, further research is required to evaluate how designers interact with such a tool. While this work mainly focuses on developing an enabling tool, its usability and learnability may also be examined in future research.

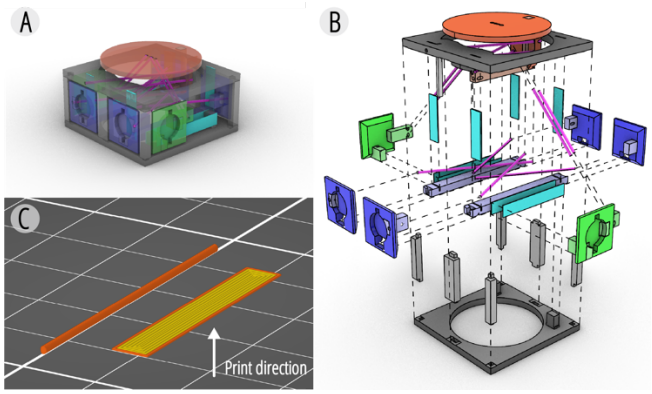

Figure 30: Device design and fabrication preparation using the knob-button design as an example. (A) The knob button model shown in figure 15B-left was segmented into multiple parts for fabrication. The model is color-coded, with each color corresponding to a rigid body or a type of flexure. (B) An exploded view of the parts assembly. A rigid body may be divided into multiple parts (annotated with different shades of the same color) that sit flatly on the 3D printer platform. Colors: magenta, rod flexures; cyan, blade flexures. (C) Flexures are printed flat on the print bed to ensure continuity along the load direction.

10 Future Work

This work mainly focuses on designing interconnected compliant mechanisms, but the fabrication challenge also bottlenecks their broader impact. The flexures' extreme aspect ratios make them prone to deformations in additive manufacturing processes when printed as a whole, compromising device precision and quality. In this work, we circumvent this issue by decomposing the devices into smaller parts that can lay flat on the build platform and assembling them post-fabrication using plastic-weld glue (Figure 30). While it is a manual process, this approach ensures good flexure quality and dimensions. Still, to reduce assembly labor, we speculate that support-free fabrication methods like powder-sintering [39] and embedded printing [1] may monolithically produce interconnected compliant mechanisms without compromising their precision. Alternatively, slacking and vibration during fabrication could be minimized by planar flexure layout [60] or cutting-away supports. Leveraging transformative active materials (e.g., self-folding structures [42] or 4D printing [57]) may also support printing devices in more fabricable forms that self-assemble into a 3D functional shape.

The transmissions of interconnected compliant mechanisms can be viewed as a form of mechanical computing [30, 49], where displacements are signals, and the structure is the processing medium that does not rely on electrical circuits. Integrating active materials further augments their ability to detect and respond to ambient interactions or adapt their functions without a mediating digital processor. This design paradigm has pros and cons that set it apart

from conventional electronic IoT. While they may be limited in computation speed and versatility, they could be less expensive and more resilient to environmental hazards (e.g., outdoor, humid). Speculating on its implications in HCI, CompAct's broader applications may include distributed sensors (energy harvesting and environment monitoring), wearable technologies (lightweight and reconfigurable exoskeleton), and input devices. These design spaces may be further supported by providing active material selection and suggestion features in the design interface, creating future development opportunities. More design guidelines may also come from extensive uses of the design tool.

11 Conclusion

This work develops a computational tool for designing interconnected compliant mechanisms with transmissive behaviors. These systems use deformable joints to transmit and negate forces and displacements, resulting in structures capable of processing displacements as signals. Devices based on such mechanisms can provide interaction-dependent haptic response and enable mechanical logic. While challenging to design, this work's computational tool facilitates the customization and modeling of devices. The design tool allows users to prescribe device behaviors by specifying the transmission input and output. The interactive workflow helps users plan out a device's mechanical structure and guides users in creating a physical design through procedurally generated modeling instructions. The proposed design system also paves the way for integrating sensing and actuation materials to augment a device's affordance further. Using this tool, we create several design examples to demonstrate the enabled design opportunities. These include modular physical interfaces with haptic I/O functions, a smart lock that can provide encrypted tangible interaction, and a wearable device that can proxy gestures and help users manipulate objects. We believe the proposed design concept, algorithm, and design tool (available at https://github.com/morphing-matter-lab/CompAct) will empower HCI researchers to navigate new design spaces while enriching interactivity through physical interfaces. Envisioning future implications, we also anticipate that our contribution will pave the way for embodied computation and interaction to integrate seamlessly into everyday life.

Acknowledgments

This work is partially supported by the National Science Foundation award numbers CAREER-2427455, 2427553, 2118924, and 2433313. Author H.Y. acknowledges the Translational Fellowship by Carnegie Mellon University's Center for Machine Learning and Health. We want to thank Koya Narumi for the logistical support and feedback regarding this research. The authors are also grateful for the reviewers' comments that helped improve the paper.

References

[1] 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.

[2] 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.

[3] 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.

[4] Cheng, T., Park, J.W., Li, J., Ramey, C., Lin, H., Abowd, G.D., Brum Medeiros, C., Oh, H. and Giordano, M. 2022. PITAS: Sensing and Actuating Embedded Robotic Sheet for Physical Information Communication. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2022).

[5] Cheng, T., Tabb, T., Park, J.W., Gallo, E.M., Maheshwari, A., Abowd, G.D., Oh, H. and Danielescu, A. 2023. Functional Destruction: Utilizing Sustainable Materials' Physical Transiency for Electronics Applications. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2023).

[6] Cheng, T., Zhang, Z., Zong, B., Zhao, Y., Chang, Z., Kim, Y., Zheng, C., Abowd, G.D. and Oh, H. 2023. SwellSense: Creating 2.5D interactions with micro-capsule paper. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2023).

[7] Cui, Z., Wang, S., Han, V.Y., Rae-Grant, T., Yang, W.Y., Zhu, A., Hudson, S.E. and Ion, A. 2024. Robotic Metamaterials: A Modular System for Hands-On Configuration of Ad-Hoc Dynamic Applications. *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2024).

[8] David Mans 2024. UI+.

[9] Forman, J., Kilic Afsar, O., Nicita, S., Lin, R.H.-J., Yang, L., Hofmann, M., Kothakonda, A., Gordon, Z., Honnet, C., Dorsey, K., Gershenfeld, N. and Ishii, H. 2023. FibeRobo: Fabricating 4D Fiber Interfaces by Continuous Drawing of Temperature Tunable Liquid Crystal Elastomers. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2023).

[10] 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.

[11] 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).

[12] 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.

[13] 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).

[14] Haynes, A.C. and Steimle, J. 2024. Flextiles: Designing Customisable Shape-Change in Textiles with SMA-Actuated Smocking Patterns. *Proceedings of the CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2024).

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] Howell, L.L., Olsen, B.M. and Magleby, S.P. 2013. Handbook of compliant mechanisms. (2013).

[20] Hughes, J.M., Lugo, M., Bouvard, J.L., McIntyre, T. and Horstemeyer, M.F. 2017. Cyclic behavior and modeling of small fatigue cracks of a polycarbonate polymer. *International Journal of Fatigue*. 99, (2017), 78–86. DOI:https://doi.org/https://doi.org/10.1016/j.ijfatigue.2016.12.012.

[21] 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.

[22] Ion, A., Kovacs, R., Schneider, O.S., Lopes, P. and Baudisch, P. 2018. Metamaterial Textures. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018), 1–12.

[23] 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.

[24] 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.

[25] 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.

[26] Kirmse, S., Campanile, L.F. and Hasse, A. 2021. Synthesis of compliant mechanisms with selective compliance – An advanced procedure. *Mechanism and Machine Theory*. 157, (2021), 104184. DOI:https://doi.org/https://doi.org/10.1016/j.mechmachtheory.2020.104184.

[27] 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.

[28] [28] Kovacs, R., Rambold, L., Fritzsche, L., Meier, D., Shigeyama, J., Katakura, S., Zhang, R. and Baudisch, P. 2021. Trusscillator: a System for Fabricating Human-Scale Human-Powered Oscillating Devices. *The 34th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2021), 1074–1088.

[29] 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.

[30] 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.

[31] Li, F.M., Liu, M.X., Zhang, Y. and Carrington, P. 2022. Freedom to Choose: Understanding Input Modality Preferences of People with Upper-body Motor Impairments for Activities of Daily Living. *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility* (New York, NY, USA, 2022).

[32] 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).

[33] Lin, H., He, L., Song, F., Li, Y., Cheng, T., Zheng, C., Wang, W. and Oh, H. 2022. FlexHaptics: A Design Method for Passive Haptic Inputs Using Planar Compliant Structures. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2022).

[34] Lin, Y., Gonzalez, J.T., Cui, Z., Banka, Y.R. and Ion, A. 2024. ConeAct: A Multistable Actuator for Dynamic Materials. *Proceedings of the CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2024).

[35] 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.

[36] Mankame, N.D. and Ananthasuresh, G.K. 2002. *CONTACT AIDED COMPLIANT MECHANISMS: CONCEPT AND PRELIMINARIES*.

[37] Mattson, C.A. 2013. Synthesis through Rigid-Body Replacement. *Handbook of Compliant Mechanisms*. Wiley. 109–121.

[38] 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.

[39] 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.

[40] 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.

[41] Nakagaki, K., Leong, J., Tappa, J.L., Wilbert, J. and Ishii, H. 2020. HERMITS: Dynamically Reconfiguring the Interactivity of Self-propelled TUIs with Mechanical Shell Add-ons. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2020), 882–896.

[42] 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.

[43] 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.

[44] Nishida, J., Matsuda, S., Matsui, H., Teng, S.-Y., Liu, Z., Suzuki, K. and Lopes, P. 2020. HandMorph: a Passive Exoskeleton that Miniaturizes Grasp. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2020), 565–578.

[45] 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.

[46] Roberts, P., Zadan, M. and Majidi, C. 2021. Soft Tactile Sensing Skins for Robotics. *Current Robotics Reports*. 2, 3 (Jul. 2021), 343–354. DOI:https://doi.org/10.1007/s43154-021-00065-2.

[47] Rong, J., Rong, X., Peng, L., Yi, J. and Zhou, Q. 2021. A new method for optimizing the topology of hinge-free and fully decoupled compliant mechanisms with multiple inputs and multiple outputs. *International Journal for Numerical Methods in Engineering*. 122, 12 (Jun. 2021), 2863–2890. DOI:https://doi.org/https://doi.org/10.1002/nme.6644.

[48] 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.

[49] 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.

[50] 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.

[51] Sun, L., Li, J., Chen, Y., Yang, Y., Yu, Z., Luo, D., Gu, J., Yao, L., Tao, Y. and Wang, G. 2021. FlexTruss: A Computational Threading Method for Multi-material, Multi-form and Multi-use Prototyping. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2021).

[52] 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.

[53] Talyan, A. 2023. Kirisense: making rigid materials bendable and functional. (Sep. 2023).

[54] 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.

[55] 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.

[56] 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.

[57] 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.

[58] 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.

[59] 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.

[60] 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*.

[61] Xu, V. and Nakagaki, K. 2023. Xs: Interactive Scissor Mechanisms as Portable and Customizable Shape-Changing Interfaces. *Proceedings of the Seventeenth International Conference on Tangible, Embedded, and Embodied Interaction* (New York, NY, USA, 2023).

[62] 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).

[63] 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.

[64] [64] Yu, Z., Liu, Y., Tinsley, B. and Shabana, A.A. 2015. Integration of Geometry and Analysis for Vehicle System Applications: Continuum-Based Leaf Spring and Tire Assembly. *Journal of Computational and Nonlinear Dynamics*. 11, 3 (Oct. 2015). DOI:https://doi.org/10.1115/1.4031151.

[65] Zheng, C., Gyory, P. and Do, E.Y.-L. 2020. Tangible Interfaces with Printed Paper Markers. *Proceedings of the 2020 ACM Designing Interactive Systems Conference* (New York, NY, USA, 2020), 909–923.

[66] Zheng, C., Yong, Z.Z., Lin, H., Oh, H. and Yen, C.C. 2022. Shape-Haptics: Planar & Passive Force Feedback Mechanisms for Physical Interfaces. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2022).

[67] 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).

[68] Zhu, B., Zhang, X. and Fatikow, S. 2014. Design of single-axis flexure hinges using continuum topology optimization method. *Science China Technological Sciences*. 57, 3 (2014), 560–567. DOI:https://doi.org/10.1007/s11431-013-5446-4.

[69] 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.