Dynamic Shape Construction and Transformation with Collective Elements

Ryo Suzuki

University of Colorado Boulder
Department of Computer Science

This thesis entitled:
Dynamic Shape Construction and Transformation with Collective Elements
written by Ryo Suzuki has been approved for the Department of Computer Science

Thesis Committee:
Prof. Daniel Leithinger (Chair)
Assistant Professor of Computer Science and ATLAS Institute
University of Colorado Boulder

Prof. Mark D. Gross
Director of ATLAS Institute and Professor of Computer Science
University of Colorado Boulder

Prof. Tom Yeh
Associate Professor of Computer Science
University of Colorado Boulder

Prof. Hiroshi Ishii
Jerome B. Wiesner Professor of Media Arts and Sciences
and Associate Director of MIT Media Laboratory
Massachusetts Institute of Technology

Prof. Takeo Igarashi
Professor of Computer Science
The University of Tokyo

A THESIS SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL OF THE UNIVERSITY OF COLORADO IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN DEPARTMENT OF COMPUTER SCIENCE

July 2020

Abstract

This thesis explores dynamic and collective shape construction as a new way to *physicalize* digital information for interactive physical displays — i.e., *shape-changing displays* enabled by a swarm of collective elements. Through physical form of digital objects, the user can directly touch, grasp, and manipulate digital information through rich tangible and embodied interactions, but at the same time, such physical objects can dynamically change their shape for an interactive computer display and interface through collective shape construction and transformation with a swarm of elements. The goal of this thesis is to envision and illustrate how such an interface might support human activities by transforming physical forms at various sizes, from millimeter to meter scale.

To achieve this goal, this thesis introduces *collective shape-changing interface*, a new type of shape-changing interfaces constructed by discrete, collective, physical elements. This proposed approach promises to address the current limitations of shape-changing interfaces — wherein a swarm of modular elements enables us to decompose the large, monolithic shape-changing objects into a set of simple, distributed elements. At the same time, their swarm behaviors enable us to make an unbounded shape transformation for expressive representation. This thesis contributes to the first exploration of this new class of shape-changing interfaces and proposes two approaches: active and passive shape construction. In active shape construction, collective elements can dynamically move and reconfigure themselves to construct a three-dimensional shape. Passive shape construction instead leverages external actuation to assemble and transform collective passive objects for dynamic shape creation. I explore and demonstrate how active and passive collective shape construction can be used as a future of computer interfaces, by developing various prototypes built on top of novel hardware and software platforms. Given these investigations, I discuss the design implications and possible research directions towards the future of collective shape-changing user interfaces.

Dedication

To Tomomi and Koto.

Acknowledgments

First of all, I would like to thank my PhD thesis advisor, Daniel Leithinger. I was extremely fortunate to be your first PhD student, Daniel. I could not ask for a better advisor, and I learned a lot from your guidance, insights, vision, and great personality throughout my PhD. All of my projects were simply not possible without your tremendous help and support. Thank you so much for everything! I would also like to thank my mentor and co-advisor, Mark Gross. Mark pushes me further by encouraging me to think more broadly, radically, and deeply. His guidance and vision helped me to think big and push the boundary of my research by looking at the future of the next decades, rather than years. Also, I want to say thank you to Mark for building the ATLAS community. What you have built over the last six years — including the space, community, culture, and people — is definitely the most important driving force in all of my research. I am extremely grateful to be part of such an incredible community. I hope I will become a person of whom you and ATLAS can someday be proud. I would also like to thank Tom Yeh, for believing in me and giving me an opportunity to come to CU Boulder as a PhD student. You are such a great mentor who gives me the chance and freedom to take risks and pursue what I am truly passionate about. I am grateful to be your PhD student as well. Thank you, Tom.

Much of my research would not exist without the inspirational vision developed by Hiroshi Ishii and his Tangible Media Group at MIT over the last decades. Hiroshi has been my hero who has significantly influenced my life and ways of thinking. As a thesis committee member and as an exceptional mentor, he was extremely supportive in helping me both shape this thesis and think beyond it. Thank you for being on my thesis committee and for providing me with your continuous guidance, support, and encouragement! Your advice inspires and pushes me to be a great researcher, thinker, and philosopher. I will continuously work hard to be able to *make a mountain* like you. I would also like to thank Takeo Igarashi, for his insightful and valuable feedback

and suggestions. Discussions with Takeo have always been fruitful and insightful, making me wonder how I could ever be such a smart person like him. It was such a precious experience to be guided by him during my PhD and JST ACT-I project. Thank you for guiding me and pushing me forward! I would also like to thank Ellen Yi-Luen Do for her tremendous support. I was almost a part of her ACME lab, where she provides me the most creative, enjoyable, and inspirational space at CU Boulder. The time when I worked in ACME lab was the most creative and productive time in my life. Thank you for coming to CU ATLAS and growing such an awesome community. And, of course, I would like to thank all of the ATLAS/HCI faculty, staff, and students at CU Boulder for their support.

I would also like to thank all of my mentors during my course of PhD. First, I would like to thank Koji Yatani and Jun Kato, for giving me a chance to start my HCI research career when I was a student at the University of Tokyo. Without their help I would definitely not be here, and I would like to deeply thank you for guiding me to where I am. Thank you, Koji and Jun! I would also like to thank my intern mentor, Michael Bernstein and Bjoern Hartmann. What I learned from my internships at Stanford and UC Berkeley constitutes the most important and precious experience for me to become a successful researcher. They showed me how the best HCI labs do research, including how to polish an idea and how to write a good paper. I learned a lot from your advice and mentoring, and I was able to see how the best researchers should be. Thank you for giving me an opportunity to work with you and learn from you! I would also like to thank my mentor and collaborator, Yasuaki Kakehi, Yoshihiro Kawahara, and Ryuma Niiyama at the University of Tokyo. Their ERATO group is one of the best teams of researchers in the world, where such creative, innovative, and forward-thinking people gather together. Much of my work would simply not be possible without the incredible collaborative opportunity cultivated by the ERATO project. I am fortunate to be part of such an amazing team, and thank you for giving me a chance to do an internship with the ERATO project. Finally, I would also like to thank my industry mentors at Adobe Research: Rubaiat Habib, Li-Yi Wei, Stephen DiVerdi, Danny Kaufman, and Wilmot Li. It was such a great opportunity to work with so many talented people in Adobe.

During my PhD, I am extremely grateful to have worked with many brilliant collaborators and friends here at CU Boulder. I would particularly like to thank my colleagues and collaborators, Clement Zheng, Hooman Hedayati, Purnendu, Darren Guinness, Abigale Stangl, Jeeeun Kim, Xu Han, and HyunJoo Oh, for all of their inspiration, conversation, and collaborations. I would also like to thank my academic sibling and best friend, Layne Jackson Hubbard, for the continuous support in both my academic and personal life throughout my PhD. A big thank

you to my amazing collaborators as well: Junichi Yamaoka from the Dynablock project, Ryosuke Nakayama, Dan Liu, and Shohei Takei from the LiftTiles project, Satoshi Nakamaru from the MorphIO project, Takayuki Hirai from the ShapeBots project, Tung D. Ta and Lining Yao from the PEP project, Kevin Kuwata and Zhixian Jin from the Reactile project, James Bohn from the RoomShift project, Gustavo Soares, Andrew Head, Elena Glassman, and Loris D'Antoni from the TraceDiff project, Niloufar Salehi, Michelle Lam, and Juan Marroquin from the Atelier project, and Rajan Vaish, Neil Gaikwad, Ranjay Krishna, Geza Kovacs, and Adam Ginzberg from the Crowd Research project. I am extremely fortunate to work with you and learn from you. Thank you so much for your incredible support. I would also like to thank my friends who always support, inspire, and encourage me. I could not list all of them here, but I would particularly like to express my appreciation to Ken Nakagaki, Koya Narumi, Shohei Aoki, and Natsuki Katayama for their insights, inspiration, and friendship.

I would like to thank my parents for all of their support. Thank you for always encouraging me to keep moving forward. I would also like to thank my brother and sister for their help and support.

Finally, to Tomomi and Koto. Thank you for always supporting me, cheering me, and encouraging me. Every single step of my life was not possible without your help and understanding. These few words are not enough to tell you how much I owe you, but I just want to thank you for all of your love. Also, thank you for being with me for almost twelve years! I am looking forward to starting the next chapter of our lives together. Thank you.

Contents

List of Figures

List of Tables

1

Introduction

"The ultimate display would, of course, be a room within which the computer can control the existence of matter. A chair displayed in such a room would be good enough to sit in. Handcuffs displayed in such a room would be confining, and a bullet displayed in such a room would be fatal. With appropriate programming such a display could literally be the Wonderland into which Alice walked."
— Ivan Sutherland [1]

[1] Sutherland, I. E. (1965). The ultimate display. *Multimedia: From Wagner to virtual reality*, pages 506–508

What if computer displays can represent information not only *graphically* but also *physically*? What if such physical forms of information could be as malleable and programmable as the pixels on a computer screen? If so, it could be used as a dynamic physical medium to interact with the digital world. Ivan Sutherland, a founder of virtual and augmented reality, once envisioned that the future of computer displays would be *"a room within which the computer can control the existence of matter"* [Sutherland, 1965]. This radical vision has inspired many researchers over the decades, as such technologies could open up a new paradigm of human-machine interfaces (Figure 1.1).

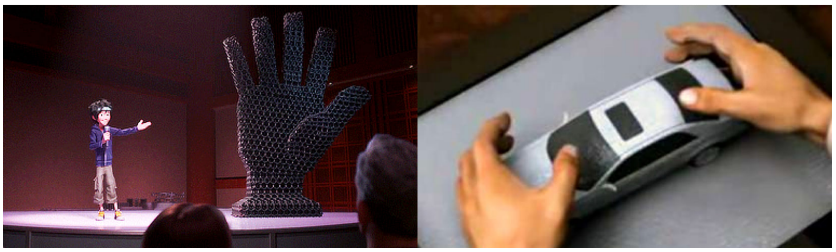


Figure 1.1: The vision of dynamic and universal shape transformation has been depicted in science fiction. Left: Microbots in Disney's Big Hero 6 movie (© Walt Disney Animation Studios). Right: Claytronics concept [Goldstein and Mowry, 2004].

From a human-computer interaction point of view, this hypothetical interface could transform every aspect of human activities. For example, by leveraging the interaction between our whole bodies and the physical world, such dynamic physical mediums could change how we design 3D models, simulate architectural designs, and understand complex systems, as we do in science museums (Figure 1.2 [2]). Dynamic physical displays could also enable a new form of remote collaboration and communication [3]. For example, a user could *physically* share an object and a space with a remote user, so that both users can modify, interact with, and collaborate through remotely shared physical objects.

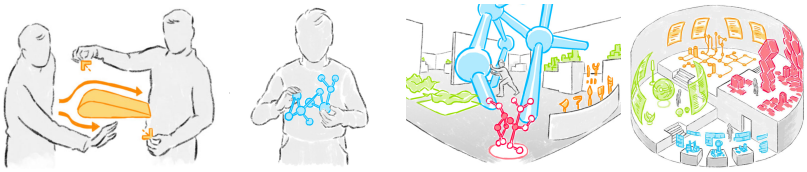


Figure 1.2: Conceptual sketches of how dynamic physical media can enhance our thoughts (presented by Bret Victor and the sketches drawn by David Hellman) [Victor, 2014a].

In our everyday life, we also unconsciously interact with a variety of physical objects and surrounding environments. If our physical environment could also adapt to users' needs, our daily lives could benefit in many ways. For instance, spaces could be reconfigured for inhabitants of small rooms, accessibility assistants could be assembled as needed (such as ramps for people with wheelchairs), and data could be physicalized for people with visual impairments to touch and understand graphs. Such interfaces could seamlessly blend digital computation into the physical world [4].

[2] Victor, B. (2014a). Humane representation of thought: A trail map for the 21st century

[3] Leithinger, D. (2015). *Grasping information and collaborating through shape displays*. PhD thesis, Massachusetts Institute of Technology

[4] Weiser, M. (1991). The computer for the 21 st century. *Scientific American*, 265(3):94–105

However, we are still far from this exciting future. Today's computer interfaces mostly focus on *screen-based* interaction, where the screens serve as a "looking window" of the digital world — the user can see digital information through the glass, but a barrier between what is inside (digital world) and what is outside (physical world) confines how we interact with the digital world. Current technologies do not allow us to directly touch, feel, grasp, and manipulate digital objects, in the same way that humans have done with physical objects for hundreds of thousands of years [Ishii and Ullmer, 1997; Victor, 2011].

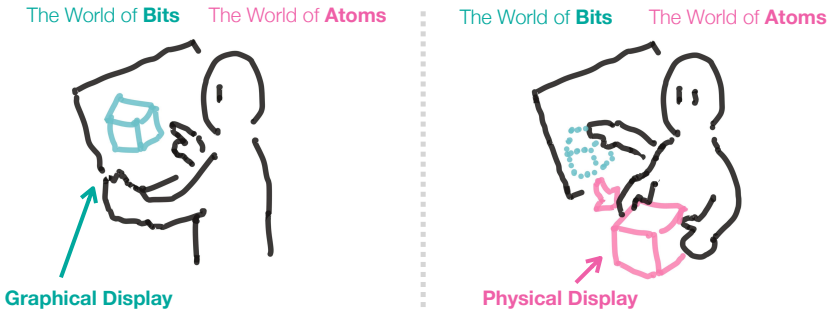


Figure 1.3: Graphical displays vs dynamic physical displays.

The goal of this thesis is to bring Sutherland's vision closer to reality by developing a new form of interactive and dynamic physical displays, and to illustrate how such an interface might support human activities by transforming physical forms and environments at various scales.

1.1 Thesis Statement

As a step toward this vision, this thesis explores dynamic and collective shape construction as a new way to *physicalize* digital information for interactive physical displays — i.e., *shape-changing displays* enabled by a swarm of collective elements. Collective elements refer to discrete physical objects that can construct a physical, three-dimensional shape. Each individual element can dynamically change its shape, position, and other

physical properties through internal or external actuation, as to collectively construct and transform the overall physical shape. This enables a new way of representing digital information. Such physical shapes allow the user not only to see information, but to touch, feel, grasp, construct, and manipulate it, in the same way that interact with physical objects. At the same time, these physical objects must also embody dynamic computation. Collective elements can dynamically and programmatically reconfigure themselves, as if they are rendered in an interactive computer display and interface.

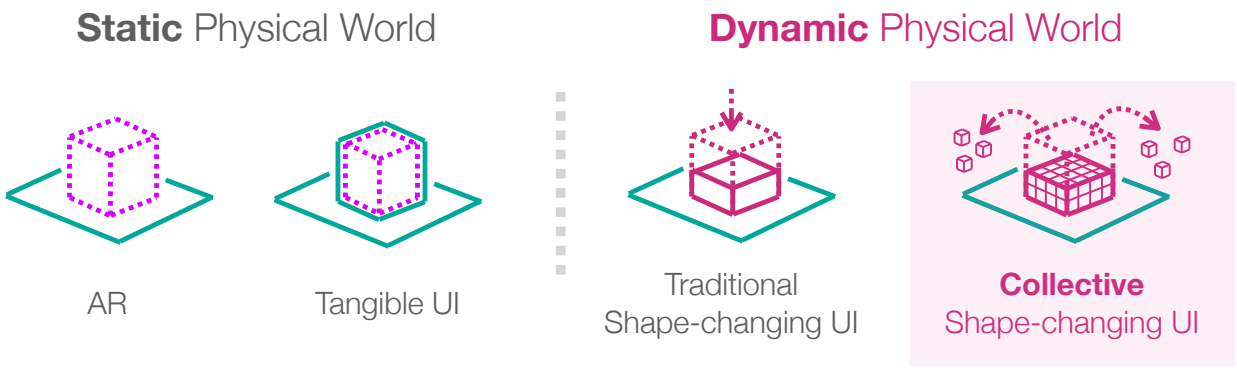


Figure 1.4: The focus of this thesis is on dynamic shape construction with collective elements. AR and Tangible UI augments the physical world (green) with virtual content (purple). Traditional shape-changing UIs dynamically change the physical world through transformable monolithic objects (red). This thesis expands this line of work by proposing dynamic physical displays constructed by a swarm of discrete collective elements.

In contrast to shape changes made of monolithic materials, constructing shapes out of discrete elements enables rich expressiveness in representing information. Like pixels on a screen, they make shapes by collectively transforming the overall structure. Additionally, their components can be simple and interchangeable, thus allowing for scale. These elements can also interact with existing environments, and they make everyday objects and environments more dynamic, adaptive, and interactive by collectively actuating and reconfiguring them in a programmable fashion.

Making shapes out of discrete collective elements is not a new idea. There is a long history of modular self-reconfigurable robots [Yim et al., 2007] and swarm robotics [Rubenstein et al.,

2014]. These areas of research have explored the idea of collective and general-purpose shape transformation for robotic applications, such as space exploration, rescue, and navigation. However, there are many critical challenges when we apply this approach for *interactive* interfaces. For example, the speed of transformation needs to be much faster than for robotic applications, as the interactive system must change and respond to the user in real-time (e.g., in seconds, not minutes or hours). Another consideration is scalability. To display meaningful information, it may require a relatively large number of elements, which often introduces implementation problems. Finally, unlike autonomous systems, interactive systems must consider the interaction between humans and objects — there remains work to be done in understanding how we might interact with such collective elements and how these interfaces could support everyday human activities.

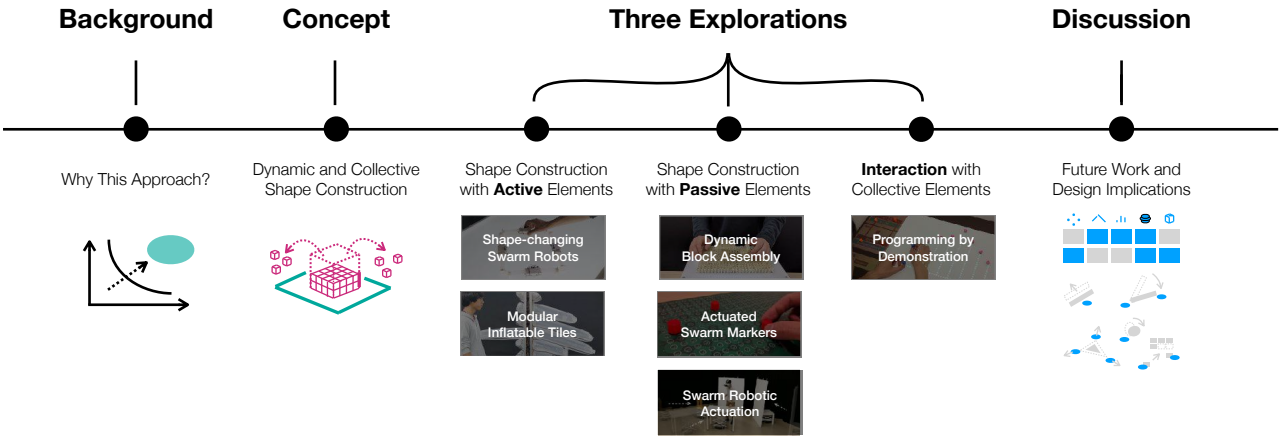


Figure 1.5: Summary of contributions of this thesis.

This thesis addresses these questions by investigating how collective shape construction and transformation can be used for *interactive computer interfaces*. To this end, this thesis introduces **collective shape-changing interfaces** [5], a new class of shape-changing interfaces constructed by a swarm of discrete physical elements. The main contribution of this thesis is the first exploration of this new class of shape-changing inter-

[5] Suzuki, R. (2019). Collective shape-changing interfaces. In *The Adjunct Publication of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19. ACM

faces in the following four domains: 1) **shape representation**: explore what types of shape representations are possible to display information, 2) **reconfiguration methods**: explore how both active and passive elements can be used to construct a shape for interactive interfaces, 3) **interaction**: explore how the user can interact with many collective elements through direct physical manipulation, 4) **applications**: explore, illustrate, and demonstrate what kind of applications are achievable for human-computer interaction.

This new class of shape-changing interfaces promises to address some of the limitations of the current shape-changing interfaces. For example, a swarm of modular elements enables us to decompose the large, monolithic shape-changing objects into a set of simple, distributed elements. This significantly contributes to the deployability of the system in everyday environments. In addition, the swarm behaviors enable us to make unbounded shape transformations for expressive representation. Through my explorations of various proof-of-concept prototypes, I demonstrate how we can push the boundary of the current shape-changing interfaces by leveraging the collective behaviors of both active and passive elements. I also demonstrate how these dynamic shapes can support a range of application scenarios, such as interactive displays, adaptive environments, dynamic data physicalization, and accessibility support for people with visual impairments. Finally, I discuss the challenges and opportunities for using this approach towards the future of dynamic physical media to augment our thinking, designing, and communicating capability through dynamic and interactive physical representation.

1.2 Thesis Contributions

This thesis makes contributions to the field of Human-Computer Interaction in the following areas:

1. A design space exploration of dynamic shape construction with collective elements
2. A new taxonomy and investigation of active and passive shape construction and transformation with collective elements
3. A novel technique for creating a dynamic shape with active shape-transformable swarm robots (e.g., ShapeBots [Suzuki et al., 2019b], LiftTiles [Suzuki et al., 2020b])
4. A novel technique for constructing 3D shapes with an assembly of passive magnetically connectable blocks (e.g., Dynablock [Suzuki et al., 2018b])
5. A novel technique for actuating passive magnetic markers with scalable electro-magnetic actuation (e.g., FluxMarker [Suzuki et al., 2017], Reactile [Suzuki et al., 2018a])
6. A novel technique for actuating existing objects to reconfigure spatial layouts (e.g., RoomShift [Suzuki et al., 2020a])
7. A novel interaction technique for programming the dynamic shape construction on a 2D surface with direct physical manipulation (e.g., Reactile [Suzuki et al., 2018a])

2

Background

"Why do people do more than just think in their heads? [...] Anyone who believes in situated, distributed, or extended cognition will have a ready explanation. Cognitive processes flow to wherever it is cheaper to perform them. The human 'cognitive operating system' extends to states, structures, and processes outside the mind and body. [...] You can harness the world to simulate processes that you cannot simulate internally or cannot simulate as well. In short, these other ways are ways of concern changing the domain and range of cognition. This is a striking claim. It suggests that as our environments and technology changes, we will be able to think about things that today are unthinkable."
— David Kirsh [1]

[1] Kirsh, D. (2010). Thinking with external representations. *AI & society*, 25(4):441–454

I argue that an interactive physical form of information display could augment the way we think, learn, design, create, and communicate ideas — beyond a graphical display. This is because physical media enables a richer and more effective way of exploring, manipulating, and sharing ideas through our bodies and physical space. Recent findings in cognitive sci-

ence and psychology show that our ability to think is directly bound to the capability of our tools and external environments, such as how appropriately they represent information [2], how directly the user can interact with them [3], and how rich modalities the user can perceive. This suggests that if tools and environments can provide more capabilities, we can further augment our thinking capability [Kirsh, 2010]. By combining the rich expressivity of physical objects with the power of computation, we can think, design, and communicate with them in unprecedented ways — just as computers and interactive displays have changed the way we think in the past decades [4].

[2] Zhang, J. and Norman, D. A. (1994). Representations in distributed cognitive tasks. *Cognitive science*, 18(1):87–122

[3] Shneiderman, B. (1993). Direct manipulation: a step beyond programming languages. *Sparks of innovation in human-computer interaction*, 17:1993

[4] Rheingold, H. (2000). *Tools for thought: The history and future of mind-expanding technology*. MIT Press

This chapter investigates the conceptual background of this argument, by reviewing the ways in which the external physical world affects our internal thinking processes and how we have used objects, tools, and spaces to augment our thinking capability. Motivated by these theoretical foundations, I then discuss why we should expand the computational medium to the dynamic physical medium, and how we should design such a new medium.

2.1 Minds Are Not Separable From Bodies and the World

2.1.1 Distributed Cognition

Since prehistoric times, people have used external representations as an extension of their thoughts. For example, even before writing was invented, humans used clay tokens to keep track of trade. This served as an extension of their memory, as external representations can store data beyond our brains. We use external representations not only to offload internal memory, but also to solve a problem (e.g., calculate with abacus), understand a problem (e.g., drawing geometry for a math

problem), support our decision making (e.g., a map and tokens for a military operation), or create an idea (e.g., brainstorm with post-it notes). Compared to thinking only in our heads, these external worlds can help us encode, process, manipulate, and store information, so that our cognitive process can be enhanced.

The theory of distributed cognition [5] has developed this notion by arguing that "our cognitive activity is distributed across internal human minds, external cognitive artifacts, and groups of people, and across space and time" [Zhang and Patel, 2006]. This idea was first outlined by [Cole and Engeström, 1993; Pea, 1993; Salomon, 1993], and subsequently the initial framework was gradually developed by [Hutchins, 1995] and recently introduced in the context of HCI [Hollan et al., 2000].

[5] Hutchins, E. (1995). *Cognition in the Wild*. Number 1995. MIT press

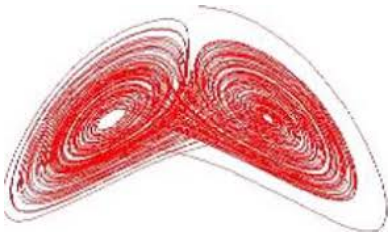


Figure 2.1: Kirsh illustrates our cognitive process is formed through interaction between internal and external processes [Kirsh, 2010].

Kirsh illustrates how this cognitive process as an interaction between internal and external processors [Kirsh, 2010] (Figure 2.1). Kirsh argues that external environments can reduce our cognitive burden by offloading the human's internal memory and processing. For example, if it is easier to interpret a problem by writing it, then one can do that instead of thinking internally alone. The analogy is with a computer system that has memory systems and scratch pads in different media and locations. Processes should migrate to wherever they are best, or most easily, performed.

The implication of distributed cognition is vast. It not only argues physical objects and environments play an important role in the human's cognitive process, but also implies that when the technology changes our environments, it can also change our ability to think. External representations allow us to encode information so that we can easily handle complex problems, as well as serves as a long-term memory, which can free our cognitive memory for other tasks.

2.2 Objects to Think With: How Objects Help Us Think

As embodied and distributed cognition suggests, we use an external world to augment our cognitive process. Physical objects particularly play a large role in this process. We use physical objects as emotional and intellectual companions to anchor memory and provoke new ideas [6]. Here, I discuss how and why physical objects can help us think leveraging 1) the power of exploration and 2) the power of constraints.

[6] Turkle, S. E. (2007). *Evocative objects: Things we think with.* MIT press

2.2.1 The Power of Exploration

An important aspect of external representation is the power of exploration [Klatzky and Lederman, 1990]. Tangible forms of information, including paper, clay, and physical models, make the idea visible and manipulatable, which allows us to explore the idea much easier. For example, when we solve a mathematical problem like $x^2 + 6x = 7$, we tend to rely on formulation and reformulation on paper [7] because, instead of processing steps in our heads, it is much easier to calculate the formula in tangible form. The physical objects allow us to reduce the cognitive load, as manipulating the encoded information provides a much faster and cheaper way than using our internal processor (i.e., brain). For example, think about if the jigsaw images on the left of Figure 2.2 can be perfectly assembled into the picture on the right. Without having tangible objects, you need to internally rearrange them in your head. However, if you have the physical pieces, this problem becomes trivial. As we can see from these examples, physical exploration can help our ability to think.

[7] $x^2 + 6x + 9 = 16$
$\Rightarrow (x+3)^2 = 16$
$\Rightarrow x + 3 = 4, -4$
$\Rightarrow x = 1, -7$

The physical manipulatives also benefit our learning and understanding of abstract concepts. Abstract concepts usually

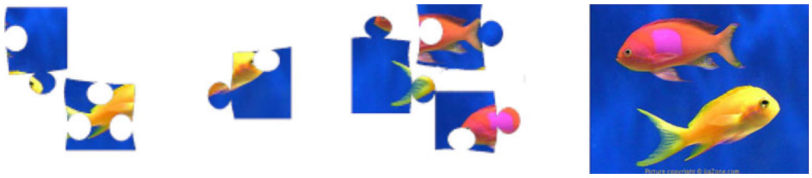


Figure 2.2: Can the jigsaw pieces on the left be assembled into the picture on the right? (The answer is no.) [Kirsh, 2010]

cannot be learned through abstract forms, rather learners need to construct and explore them by interacting through concrete materials [8]. This idea is also a basis for designing modern science museums [9]. A complex mathematical and physics concept becomes understandable if it is represented as tangible and explorable forms. The physical manipulatives allow learners to construct an underlying mathematical model through interacting with it. Such physical exploration has been also a central theme of how to learn the concept of computation. Papert argued that if a computer becomes an "object to think with", learners can naturally develop the idea of computation and programming [10].

This exploration plays a particularly important role when we work on unsolved problems. For example, in the 1950s, the identification of the structure of DNA was a scientific challenge. The connection of each base and nucleotide in the DNA has a complex double helix structure, and thinking about this three-dimensional structure in one's head or even with the paper was a challenging task. Watson, Crick, and Franklin approached this problem by making physical models to narrow down the possibilities and eventually create an accurate picture of the molecule, which was then based on a single X-ray diffraction image taken by Franklin and Gosling. Their use of the physical model allowed them to understand the structure with their hands in three-dimensional space. Thus, physical form of exploration would have helped them to approach to this unpredictable and unforeseen result.

[8] Papert, S. and Harel, I. (1991). Situating constructionism. *Constructionism*, 36(2):1–11

[9] Oppenheimer, F. (1972). The exploratorium: A playful museum combines perception and art in science education. *American Journal of Physics*, 40(7):978–984

[10] Papert, S. (1980). Mindstorms: Computers, children, and powerful ideas. *NY: Basic Books*

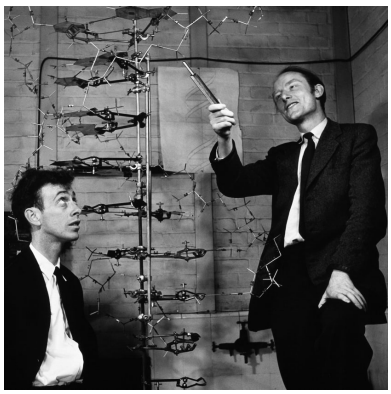


Figure 2.3: James Watson and Francis Crick with their DNA model at the Cavendish Laboratories in 1953.

As we can see, giving information physical form enables a powerful way to explore and manipulate ideas. In addition, the dynamic media could allow us to build and rebuild a model and hypothesis in much faster ways, with the power of computation. It should help foster and augment thinking by leveraging the advantages of both physical representations and computational media.

Figure 2.4: Left: Our hands are capable of rich tangible interaction [Victor, 2011]. Right: In contrast, interaction with screens is mostly through tap and swipe gesture (© Microsoft Research).

2.2.2 The Power of Constraints

Another important aspect of physical objects is the power of constraints. Constraints often provide benefits for human thinking as it can limit the choice of actions and search space, which, in turn, can reduce our internal cognitive load. In general, appropriate constraints make a faster decision making [11] and more creativity [12] in our cognitive process. Physical constraints given by an appropriate physical representation are also known to be an important factor for faster problem solving and creative process.

[11] Iyengar, S. (2010). *The art of choosing*. Twelve

[12] Smith, S. M., Ward, T. B., and Schumacher, J. S. (1993). Constraining effects of examples in a creative generation task. *Memory & cognition*, 21(6):837–845

To reveal this, Zhang and Norman asked participants to solve the problem of the Towers of Hanoi in three different settings (See Figure 2.5) [Zhang and Norman, 1994]. All of the three settings are isomorphic, in the sense that the games' abstract rules and structure are the same — they share the same four rules:

1. Rule 1) Only one object can be transferred at a time,

2. Rule 2) An object can only be transferred to a plate on which it will be the largest,
3. Rule 3) Only the largest object in a plate can be transferred to another plate,
4. Rule 4) The smallest object and the largest object can not be placed on a single plate unless the medium-sized object is also on that plate

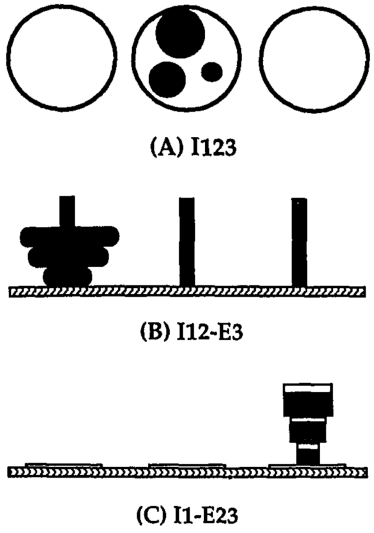


Figure 2.5: Three different settings of the same Towers of Hanoi problem. A) Waitress and Oranges, B) Waitress and Donuts, C) Waitress and Coffees [Zhang and Norman, 1994].

However, the representation of these abstractions is different. For example, when you rearrange glasses (setting C), many of these rules are obvious, because you cannot grab a smaller glass in the bottom without transferring the larger one on the top (Rule 2 and 3) due to physical constraints. However, when you rearrange oranges (setting A), there are no physical constraints to enforce these rules, as you can freely pick and transfer any size of the objects on a plate. In this case, you need to memorize these rules whenever you take action.

In this experimental test, Zhang and Norman [13] found with statistical significance that settings with more physical constraints yielded successive improvements in solution times, solution steps, and error rates. In other words, the more rules are distributed in the external representation, the easier the problem. Drawing from a series of cognitive studies, Zhang [Zhang, 1997], Norman [Norman, 1993], and Scaife and Rogers [Scaife and Rogers, 1996] discuss the importance of the physical structures in cognitive tasks, as the derived constraints from these structures can guide and even determine cognitive behavior.

Physical constraints also guide natural interaction. With constraints, humans can naturally perceive what are the possible actions and what is not. For example, if a wall prevents the door from moving backward, it can visually and physically inform the person that they should push, not pull. Thus, phys-

[13] Zhang, J. and Norman, D. A. (1994). Representations in distributed cognitive tasks. *Cognitive science*, 18(1):87–122

ical constraints are closely related to the theory of affordances — an object's properties that show the possible actions users can engage with it — which was originally introduced by Gibson [14] and developed in the context of HCI by Norman [15]. With appropriate design, the structure of physical tools can naturally guide actions, and thereby reduce the cognitive load of learning how to use tools, allowing the user to focus cognitive resources on solving the problem [Patten and Ishii, 2007].

[14] Gibson, J. J. (1979). *The ecological approach to visual perception: classic edition*. Psychology Press

[15] Norman, D. A. (1999). Affordance, conventions, and design. *interactions*, 6(3):38–43; and Norman, D. A. (2013). *The design of everyday things: Revised and expanded edition*. Basic books

2.3 Environments to Think In: How Spaces Help Us Think

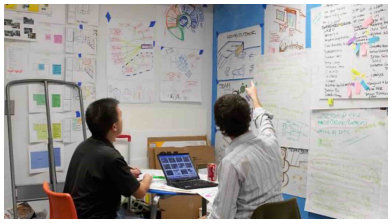


Figure 2.6: Butcher paper lines the wall of the Stanford d.school meeting room [Klemmer et al., 2006].

Physical objects are not the only external representation that contributes to our cognitive process. We also use space for sensemaking. Effective and intelligent use of space can significantly enhance our ability to think [Kirsh, 1995]. Here, I discuss how two aspects of the user of space can help us think: 1) the power of spatial reasoning and 2) the power of collaboration.

2.3.1 The Power of Spatial Reasoning

Humans have a powerful ability to understand, reason and remember the spatial relations among objects or space [16]. For example, we can perceive the meaning of a diagram by leveraging visual reasoning [Larkin and Simon, 1987; Scaife and Rogers, 1996], identify a pattern on a map with spatial thinking [Council et al., 2005; Snow, 1855], remember the position of tools in a workspace with spatial memory [Klemmer et al., 2006], and quickly scan certain targets in a scene with visual search [Duncan and Humphreys, 1989; Wolfe, 2010]. We also naturally use this capability in everyday life, such as making sense of the grouping of notes on a wall in brainstorming or

[16] Klemmer, S. R., Hartmann, B., and Takayama, L. (2006). How bodies matter: five themes for interaction design. In *Proceedings of the 6th conference on Designing Interactive systems*, pages 140–149

leaving keys or notes for ourselves close to the door to serve as later reminders.

These practices are particularly important when one needs to deal with creative and complex tasks [Czerwinski et al., 2003]. The size of the display can significantly affect search, navigation [Ball et al., 2007; Liu et al., 2014], visualization [Rønne Jakobsen and Hornbæk, 2011], and sense-making tasks [Andrews et al., 2010]. Also, the size of the workspace can also affect the versatility of the solution [Kirsh, 1995]. We can see many real-world examples where people working on complex tasks tend to use larger workspace (e.g., researchers who brainstorm with space in Stanford d.School [17], and professionals who monitor a complex situation in NASA space centers or television broadcasting studio [18].)

In addition to the size of the workspace, the ability of make spatial arrangements is also a key. Kirsh [19] identifies three benefits of spatial arrangements for the cognitive process: spatial arrangements and dynamics can simplify choice, perception, and internal computation. Persistence of physical objects make this process easier, as we can manipulate and rearrange objects with our bodies. Consider brainstorming an idea with a whiteboard or post-it notes. Notes can be easily organized and rearranged on a wall, which allows us to create a spatial meaning or explore different grouping ideas. Experts constantly rearrange items in areas for long-term, medium-term, and short-term structuring, which makes it easy to track the state of the task or notice the properties signaling what to do next [Kirsh, 1995].

[17] Doorley, S., Witthoft, S., et al. (2012). *Make space: How to set the stage for creative collaboration*. John Wiley & Sons

[18] Victor, B. (2014b). Seeing spaces. In *Talk at EG conference*

[19] Kirsh, D. (1995). The intelligent use of space. *Artificial Intelligence*, pages 31–68

2.3.2 The Power of Shareable Thoughts

Physical and spatial representation of ideas also naturally encourages communication and collaboration. Physical models

or prototypes facilitate communication within a design team, with clients, or users, by providing a concrete anchor around which discussion can occur [Klemmer et al., 2006]. Since all people can manipulate the artifact and see the results of action immediately and simultaneously, this allows more seamless collaboration among participants [Dourish, 2004], in contrast to today's collaborative practice with computers — each person stares at their own screen, and interaction with others seldom happens in a shared space [Victor, 2014a].

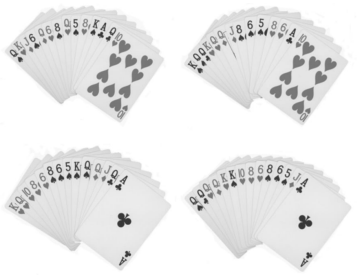


Figure 2.7: Rearrangement simplifies choice, perception, and internal computation. These four figures represent three different ways of organizing the same set of cards. The figure in the upper right shows the cards as originally dealt. The subsequent figures show the rearrangements of three subjects [Kirsh, 1996].

Particularly, architects, designers, and engineers exploit the benefits of physical artifacts, prototypes, and models, which can serve as shared objects of thought [Kirsh, 2010]. Such physical artifacts can help uncover problems or generate suggestions for new designs, thus becoming the "essential medium for information, interaction, integration, and collaboration" [Klemmer et al., 2006; Schrage, 1996] Creating intermediate tangible artifacts allows revealing tacit knowledge. Kirsh argues that the unexpected ideas emerge from these interactions because physical models allow other people to independently evaluate from its creator's original thoughts, thus they can approach the model in ways unconstrained by its creator's intention. This helps the discovery of new interpretations or ideas.

Physical objects and shared space also play an important role in collaborative learning. For example, by using jigsaw puzzles and breaks assignments into pieces that the group assembles to complete it, Aronson reveals that the physical and collaborative activity can help weaken racial cliques in forcibly integrated schools and increase the communication between students [20]. As we can see, providing learners with 'tools to think with' [Resnick et al., 1998] offers opportunities for collaborative activity among learners [Africano et al., 2004; Suzuki and Kato, 1995].

[20] Aronson, E. et al. (1978). *The jigsaw classroom.* SAGE Publications

As we can see, physical artifacts and shared space can strongly enhance collaboration. This aspect is particularly important for a creative process, as artistic and scientific revolutions often emerge from a place where collaboration happens in the physical space, such as Xerox PARC [21], Bell Labs [22], and MIT's Building 20 [23]. The significant discoveries in scientific research are emerged not from individual thinking, but from distributed reasoning — groups of scientists reason about different topics from different angles [Dunbar, 1995, 1997] Enhancing such creative and collaborative process with physical artifacts should be an important driver for thinking, learning, and discovering new ideas.

[21] Rheingold, H. (2000). *Tools for thought: The history and future of mind-expanding technology*. MIT Press

[22] Gertner, J. (2012). *The idea factory: Bell Labs and the great age of American innovation*. Penguin Publishing Group

[23] Brand, S. (1995). *How buildings learn: What happens after they're built*. Penguin Publishing Group

2.4 Design Implications for a New Computational Medium

Now that we have better understandings of how external objects and space help us think. Motivated by these theoretical foundations, we can discuss how we should expand the current computational media to better embrace these benefits.

2.4.1 Expand a Range of Possible Representations

The way of representing information affects the way of thinking [Kirsh, 1995]. For example, we are all aware that Arabic numerals are more efficient than Roman numerals for multiplication (e.g., 73 x 27 is easier than LXXIII x XXVII), and it is easier to find maxima and minima in a graph than in a table, even though both types of representations capture the same information [Playfair, 2005; Zhang and Norman, 1994]. Throughout history, humans have augmented their thinkable territory by inventing new ways of representing information [Victor, 2013]. Written languages — a way of representing verbal data

to persistent physical marks — enable to transfer knowledge regardless of time and location, mathematical notations — a way of representing an abstract concept with a concise expression — enable us to manipulate mathematical notions more easily and precisely, and data graphics — a way of representing textual data to graphical one — enable us to gain better insights into data in faster ways. As we can see, expanding the possible range of representing information can potentially transform our way of thinking.

Recent findings in cognitive science, discussed in this chapter, suggests physical and spatial representations provide powerful ways of thinking, designing, and understanding ideas by leveraging our innate capability of physical exploration and spatial reasoning. However, to date, computers have only leveraged the visual and symbolic representations — for example, when we interact with computers, we only rely on text and graphics to read, learn, think, and discuss.

How we can expand the possible representation of digital data is a key challenge and question when designing a new computational medium. We could take inspiration from the current practice in spatial (e.g., notes on a wall, bookshelves) or physical representations (e.g., physical models, block toys) that we already engage, and rebuild them by leveraging computation. As such new representations can exploit the new area of human ability and potentially augment our way of thinking in unprecedented ways, inventing new dynamic three-dimensional representations out of computer screen provides difficult yet exciting challenges for human-computer interaction research [24].

[24] Dourish, P. (2004). *Where the action is: the foundations of embodied interaction*. MIT press

2.4.2 Leverage Dynamic Computation

Dynamic and interactive representations are only available with computational media [Kay, 2011]. Computation and dynamic representation have enabled us to model and remodel various hypotheses to effectively solve problems that are difficult to think through in advance [25] and to create interactive artifacts with music, animations, and diagrams that the viewers can interactively change and explore by their own. Dynamic physical media also should maintain this dynamicity and interactivity of digital computation. To embrace these properties, they need to be as dynamic, malleable, and programmable as the pixels on a computer screen and change the state with the speed of thought. Kay described when designing such a personal dynamic medium, one of the metaphors was *"that of a musical instrument, such as a flute, which is owned by its user and responds instantly and consistently to its owner's wishes. Imagine the absurdity of a one-second delay between blowing a note and hearing it"* [26]. We expect that these systems will be responsive, as we do with other media.

[25] Licklider, J. C. (1960). Man-computer symbiosis. *IRE transactions on human factors in electronics*, (1):4–11

[26] Kay, A. and Goldberg, A. (1977). Personal dynamic media. *Computer*, 10(3):31–41

The key strength of computational media is that it can be programmed. Thus, an interesting question is "how should we program it, and how can the user interact with the programmed behaviors?" The user's programmability is one of its core in the original Dynabook vision [Kay and Goldberg, 1977], and we also need to start thinking about what would be the appropriate representation of not only for display information but also for programs of its artifacts.

2.4.3 Harness Embodied and Collaborative Interactions

Despite the prevalence of today's touchscreen interfaces, our hands can do more than tap and swipe. Since the prehistoric

era, humans have developed an incredible capability of manipulating space and objects. For example, Napier illustrates how various ways our hands can grasp objects. Not only good for grasping, our hands also have an incredible capability to sense the tactile sensation (e.g., we can sense 0.1mm thickness of hair). These capabilities are currently underused when interacting with computers, as they restrict the interaction with typing a keyboard or clicking buttons. For example, although we can easily go through a book by flipping pages and sense the remaining pages with two hands, the same ease of browsing is not well supported in e-books. Also, as we discussed, our working space is restricted in a tiny screen, despite the fact we can much effectively reason, navigate, and make sense if we use a larger area of the surrounding physical wall or desk.

We should develop a computational medium, based on humans' ability — computer interfaces should be designed to leverage human capability, rather than humans restrict their ability to computer interfaces. Based on these theoretical and practical benefits of using physical objects and space, there is a growing interest in how we can leverage embodied interaction for user interface design. With an appropriate computational medium, we can now build computing around our skills for physical interaction with objects.

I have discussed the motivations and theoretical backgrounds of why we need to start thinking about a new computational medium that goes beyond the current computer screen. The scope of this goal is vast. It spans from the developing interaction techniques with augmented physical artifacts to an empirical analysis of how they affect in practical applications. Among them, one of the difficult challenges is the development of novel technologies that can embrace dynamic computation in the physical world. Particularly, making the physical objects and space dynamic and programmable is still a remaining

problem in the field of HCI. This thesis aims to specifically contribute to this domain by proposing novel ways of displaying information with dynamically movable collective elements. While designing practical applications and investigating how these displays can effectively augment our thought could go beyond the scope of this thesis, these technologies and interaction techniques could contribute to the foundation of the next generation of interactive computational media and let the successors build applications on top of it.

Based on these findings and motivations, researchers in HCI have approached new computer interfaces over the past decades. In the next chapter, I will review how these ideas turn into the development of systems, tools, and media to enable dynamic computation in the physical world and discuss what are the challenges in the current research.

3

Related Work

"We live in a complex world, filled with myriad objects, tools, toys, and people. Our lives are spent in diverse interaction with this environment. Yet, for the most part, our computing takes place sitting in front of, and staring at, a single glowing screen attached to an array of buttons and mouse. From the isolation of our workstations, we try to interact with our surrounding environment, but the two worlds have little in common. How can we escape from the computer screen and bring these two worlds together?"
— Pierre Wellner [1]

[1] Wellner, P. (1993). Interacting with paper on the digitaldesk. *Communications of the ACM*, 36(7):87–96

During the past decades, researchers in HCI have looked into the way to permeating digital computation into the world around them and many approaches have been developed towards this goal. Ubiquitous computing and the Internet of Things leverage distributed sensors to make computers understand our activities so that they can provide in-situ, context-aware assistants for people in an environment. Augmented reality uses physical worlds as an expressive canvas on which to display information, so that we can see digital content out

of the computer screen. Tangible interfaces provide physical control and representation for the digital world, so that we can grasp and manipulate information through rich embodied interaction.

Recent research also began exploring how the physical world itself can embody the dynamic computation, so that objects and materials themselves can dynamically change to reflect the underlying computation, going beyond overlaying information on top of them. Although this is challenging, researchers across many disciplines, including robotics, mechanical engineering, material science, and human-computer interaction have been actively pursuing this direction by developing novel actuation and smart materials. For example, they have developed shape-changing interfaces [Alexander et al., 2018; Rasmussen et al., 2012] with transformable materials [Yao et al., 2013] and dynamic physical displays with shape-changing surfaces to dynamically present a shape that the user can touch [Follmer et al., 2013; Iwata et al., 2001]. Outside the context of HCI research, robotic researchers also actively explore modular self-reconfigurable robotics [Yim et al., 2007] for general-purpose transformable robots. Also, recent work leverage digital fabrication techniques and transformable materials as a way to design and built environment-responsive materials that can change its shape, including metamaterials [Ion et al., 2016; Ou et al., 2018], 4D printing [Tibbits, 2014], self-assembly and folding [Hawkes et al., 2010; Whitesides and Grzybowski, 2002], and digital materials [Hiller and Lipson, 2009; Hiller, 2011; Popescu et al., 2006].

My work presented in this dissertation is largely inspired by these emerging research fields in human-computer interaction, robotics, and other related disciplines. This chapter reviews these existing works to illustrate what has been achieved, how these ideas have been developed, and what are the remaining

challenges, when we apply these techniques for dynamic physical media. I then briefly discuss how this work can fill a gap in the current stream of research.

3.1 Ubiquitous Computing and Augmented Reality

3.1.1 Calm Computing

In his seminal article, The Computer for the 21st Century, Mark Weiser envisioned computing technologies will disappear and weave themselves into the fabric of everyday life until they are indistinguishable from it [2]. He coined the term "ubiquitous computing" (and later changed to "calm computing") as a vision to illustrate this goal, where "computing is so ubiquitous that no once notices its presence". Over the past decades, computing gradually became ubiquitous and a part of our environment, as he envisioned. Smart homes and offices that can continuously monitor and help inhabitants, were originally developed through a proof-of-concept prototype e.g., MIT's House N, Georgia Tech's Aware Home, and Duke's SmartHouse [3]. Today, these technologies have spread widely and we are almost unaware of distributed sensors that continuously sense our activity and pervasive IoT devices that are embedded in environments and computationally control lighting, heating, and air conditioning.

[2] Weiser, M. (1991). The computer for the 21 st century. *Scientific American*, 265(3):94–105

[3] Ricquebourg, V., Menga, D., Durand, D., Marhic, B., Delahoche, L., and Loge, C. (2006). The smart home concept: our immediate future. In *2006 1st IEEE international conference on e-learning in industrial electronics*, pages 23–28. IEEE

Through ubiquitous computing, sensing and activity tracking have gradually disappeared, meaning that they have become invisible or not noticed by people. But, when it comes to perceiving information, the digital and physical worlds rarely merge. While we have laptops, tablets, smartphones, and large displays almost everywhere, today's ubiquity of digital computation mostly comes in the form of the ubiquity of the com-

puter screens. Over twenty years ago, Wellner illustrated this situation as *"our computing takes place sitting in front of, and staring at, a single glowing screen attached to an array of buttons and mouse"* [Wellner, 1993], but the situation has not changed much since then. We still use at most two or three screens and stare at them all the time to perceive information.

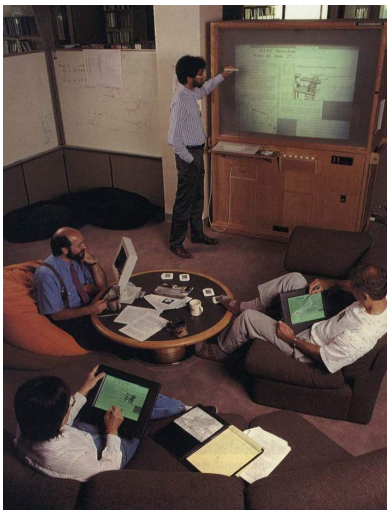


Figure 3.1: The Computer for the 21st Century [Weiser, 1991].

Weiser envisioned a display as ubiquitous as paper — it is cheap, disposable, shareable, and supports the same affordances and interactions as when we use paper. For example, he described *"Pads differ from conventional portable computers in one crucial way. Whereas portable computers go everywhere with their owners, the pad that must be carried from place to place is a failure. Pads are intended to be "scrap computers" (analogous to scrap paper) that can be grabbed and used anywhere; they have no individualized identity or importance. [...] Pads, in contrast, use a real desk. Spread many electronic pads around on the desk, just as you spread out papers. [...] Spread the many parts of the many tasks of the day out in front of you to fit both the task and the reach of your arms and eyes, rather than to fit the limitations of CRT glass-blowing."* [Weiser, 1991] However, today's reality is rather the opposite. The ubiquitous computers, such as laptops, tablets, smartphones, and large displays, are personal, expensive, and non-shareable display devices. They rarely support the same affordances when we use paper.

To achieve Weiser's original vision of "transparency and invisibility of computing and its user interfaces" [4], an alternate approach has been explored by "augmenting" the existing environments, instead of covering the entire wall, table, and surface with computer displays — that is *spatial augmented reality*.

[4] Ishii, H. (2004). Bottles: A transparent interface as a tribute to mark weiser. *IEICE Transactions on information and systems*, 87(6):1299–1311

3.1.2 Spatial Augmented Reality

Augmented reality situates virtual content in the physical world, so that the user can see and interact with the virtual content as if they exist in the real world. There are two approaches: one is overlay information with a projector or overlay information through see-through displays [Bimber and Raskar, 2006].

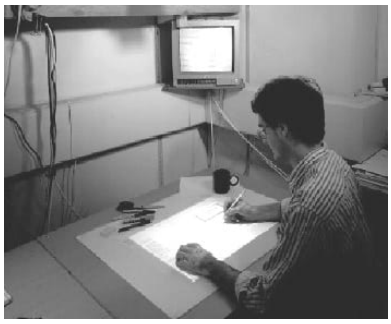


Figure 3.2: Wellner's Digital Desk [Wellner, 1993].

In the first approach, called spatial augmented reality, visual information is directly projected onto the environment, usually through the use of projectors. Wellner contributed to this approach with his DigitalDesk [5]. In his work, the system displays information overlaid on top of a paper, so that the user can use a sheet of paper as both physical paper and digital display. With this spatial projection, the user can see the dynamic and interactive content just like a computer display, but can also use it as an inexpensive, tangible, and disposable paper. Spatial augmented reality allows surfaces of the physical environment to become extensions of traditional computing environments. CAVE (Cave Automated Virtual Environment) is another early demonstration of using ordinary surfaces to display information [Cruz-Neira et al., 1993]. Raskar et al extended this idea to everyday non-flat surfaces and envisioned how this technology might enable "Office of the Future" [Raskar et al., 1998]. In "Augmented Surfaces", Rekimoto et al developed a new interaction called "pick and drop", as a metaphor of the GUI "drag and drop" in laptop computers, which allows the user to pick graphical objects and drop them onto any surface in the physical space, so that the system can seamlessly migrate between different interactive surfaces, as well as associated with physical objects [Rekimoto and Saitoh, 1999].

[5] Wellner, P. (1993). Interacting with paper on the digitaldesk. *Communications of the ACM*, 36(7):87–96

Figure 3.3: Dynamicland [Victor et al., 2018].

Since then, spatial augmented reality has been used or deployed to support interaction with objects and the physical

environment in many different application domains [6], including office application [Olwal and Wilson, 2008], entertainment [Willis et al., 2011a], sports [Alves et al., 2013; Ishii et al., 1999; Kajastila et al., 2016], learning and education [Do-Lenh et al., 2012; Furió et al., 2017], medical instruction [Meng et al., 2013; Hoang et al., 2017], and data visualization [Raskar et al., 2004]. With the advent of mobile projectors, it became possible to create handheld [Willis et al., 2011b, 2013], on-body [Harrison et al., 2010, 2011], or wearable [Xiao et al., 2018] projection mapping. Recent work also aims to make the spatial augmented reality more ubiquitous and deployable with novel touch interact techniques [Xiao et al., 2017] or enable end-users to customize the content [Xiao et al., 2013] or program with object-oriented programming [Victor et al., 2018].

[6] Jones, B. and Sohdi, R. (2012). The illustrated history of projection mapping

3.2 Tangible and Graspable User Interfaces

3.2.1 Graspable User Interfaces

Inspired by augmented reality, researchers have further explored ways to integrate the real world and computational media. One missing aspect in the early work on augmented reality was the use of physical objects as user inputs. Although augmented reality uses physical surfaces as a canvas to display digital content, most work did not fully leverage physical objects as user inputs. This idea was originally developed through Bricks [7], in which visual objects can be attached to physical bricks, so that the virtual content that has now a physical representation can be physically grasped with hands. This was one of the first demonstrations of the direct physical embodiment of the digital content. In a similar project, Hinckley et al. explore passive physical props to control cross-sections in CT scan images [Hinckley et al., 1994]. By using the physical props, the user can naturally navigate through complex

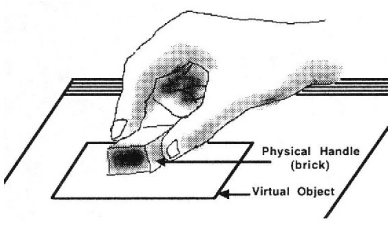


Figure 3.4: Bricks: Graspable User Interfaces [Fitzmaurice et al., 1995].

[7] Fitzmaurice, G. W., Ishii, H., and Buxton, W. A. (1995). Bricks: laying the foundations for graspable user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 442–449

three-dimensional visualization with bimanual interaction. These ideas led to a new paradigm of user interaction, called Graspable User Interfaces [8], where the user can "grasp" and "manipulate" digital content through physical handles.

[8] Fitzmaurice, G. W. (1997). *Graspable user interfaces*. PhD thesis

3.2.2 Tangible User Interfaces

Tangible User Interfaces [9] was introduced in this context to take one step further for the integration. Tangible user interfaces emphasize the direct coupling between digital content and physical objects so that physical objects can serve both as inputs and as representation of the augmented virtual content. To demonstrate this concept, Urp and I/O bulb [10] show an early compelling example. In this project, the physical building model serves as not only physical handles but also an embodied representation of the same building model in computational space, providing a tighter coupling between the digital world and the physical world. Similarly, Illuminating clay [Piper et al., 2002] and Sand Scape [Ishii et al., 2004] further expands this idea for deformable physical objects. In this project, a projected digital image on top of the clay or sand can show the simulated water runoff or erosion patterns based on the current physical shape. In these demonstrations, physical objects are no longer merely a handle of the digital content. Rather, the physical objects also serve an embodied representation of the digital content, as their physical properties, such as shape, position, materials, and textures are directly coupled with the underlying digital model.

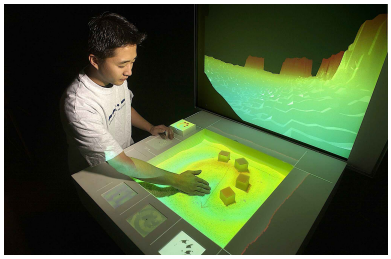


Figure 3.5: SandScape [Ishii et al., 2004].

[9] Ishii, H. and Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, pages 234–241. ACM

[10] Underkoffler, J. and Ishii, H. (1999). Urp: a luminous-tangible workbench for urban planning and design. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 386–393

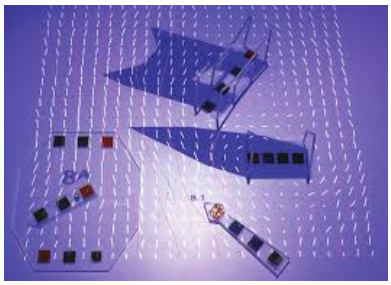


Figure 3.6: Urp and I/O Bulb [Underkoffler and Ishii, 1999].

Since Ishii and Ullmer first introduced the concept of tangible interfaces, the design space of tangible interfaces has continuously been growing, and many works have been demonstrated [Jordà et al., 2007]. At the same time, theoretical frameworks of tangible user interfaces were also developed [Ullmer and Ishii, 2000]. For example, Ullmer and Shaer et al. in-

troduce the token+constraint framework [Shaer et al., 2004; Ullmer et al., 2005] to highlight the importance of mechanical and physical constraints, and the proposed interactions are demonstrated through various projects, such as mediaBlocks [Ullmer et al., 1998] and SenseTable [Patten et al., 2001]. Fishkin provides an overview of many tangible interfaces and organizes them across the level of embodiment and other axes [Fishkin, 2004]. Other tangible interfaces have had a closer connection with visualizing and exploring digital information, by enhancing them with physical affordances and constraints [Zigelbaum et al., 2008]. Dourish gives an overview of the historical development as well as theoretical foundations for tangible user interfaces [11].

[11] Dourish, P. (2004). *Where the action is: the foundations of embodied interaction*. MIT press

3.2.3 Ambient Displays and Media

In their original paper, Ishii and Ullmer considered not only physical objects, but also the entire physical world as a user interface [Ishii and Ullmer, 1997]. In this view, the user's surrounding environment also serves as an interface to interact with the digital world. To demonstrate this second aspect — using the background environment as ambient media —, the concept of ambient media was developed. For example, in ambientRoom [Ishii et al., 1998], ambient information, such as sound, light, airflow, water flow, and physical movements as peripheral displays at the background of user attention. Similarly, Pinwheels [Ishii et al., 2001] and musicBottles [Ishii, 2004] are also designed to demonstrate this concept. The underlying motivation of these ambient displays is to design the media that promote human interaction, rather than detract from it. Recent work like AwareMirror [Fujinami et al., 2005] and Squama [Rekimoto, 2012] show some examples of this idea of ambient display.

Figure 3.7: PinWheels [Ishii et al., 2001].

3.2.4 Constructive Assemblies

Another major domain of tangible interfaces is constructive assemblies. Drawn by inspiration from building blocks, the constructive assemblies employ modular, electronically instrumented building blocks for tangible interfaces, which is closely related to this thesis' theme. As one of the earliest examples, Aish demonstrated a "building block system" for modeling a 3D model in computer-aided architectural design by physically assembling it [Aish, 1979]. One interesting aspect of these systems is that they were used to explore not only the geometric structure of buildings, but also some of the more abstract resulting properties and parameters of the building [Aish and Noakes, 1984].

Figure 3.8:
AlgoBlock [Suzuki and Kato, 1995].

Constructive blocks have been also utilized for learning programming, in which the blocks can support computational thinking, offering tangible representations for abstract concepts such as program flow and variables. These blocks help children develop a natural understanding of abstract concepts which can be difficult to gain through textual and symbolic representation. AlgoBlock is an example of tangible programming blocks [12]. By representing a command with different types of blocks, children can physically construct a program and execute commands with embedded buttons.

[12] Suzuki, H. and Kato, H. (1995). Interaction-level support for collaborative learning: Algoblock—an open programming language. In *The First International Conference on Computer Support for Collaborative Learning*, CSCL '95, pages 349–355, Hillsdale, NJ, USA. L. Erlbaum Associates Inc

Constructive assembly becomes a powerful way of thinking and exploring three-dimensional structures. Compared to seeing a 2D projection of a 3D structure in a flat screen and manipulating it through continuously changing the view angle, the physical exploration has advantages in both understanding as well as manipulating the structure. Posey is one example of geometric construction with hub and struts [Weller et al., 2008]. These hub and struts have embedded arrays of infrared LEDs and photosensors that can sense the connection and angle be-

tween elements, so that it can be used for a tangible input of exploring three-dimensional structures of molecular structure. These physical inputs also help creating animated motion of 3D puppetry and character animation [Jacobson et al., 2014; Held et al., 2012]. In contrast to these passive building blocks, Topobo [13] demonstrated programmable active blocks, which can not only sense the motion, but also actuate. With Topobo, the user builds structures by connecting blocks, then program their movement through physical demonstration. Each block can remember and recall the motion as the demonstrated programmed behavior, which can later be used to animate the model.

[13] Raffle, H. S., Parkes, A. J., and Ishii, H. (2004). Topobo: a constructive assembly system with kinetic memory. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 647–654

Figure 3.9: Topobo [Raffle et al., 2004].

These building blocks have advantages over objects with static predefined structures — like LEGO blocks, endless combinations of blocks allows the rich expression of the resulting object, thus does not limit the user's creativity. Given this property, using collective discrete elements could be one of the promising approach for universal shape construction, and explore the design space of interactive and dynamic shape construction, beyond manual assembly.

3.3 Actuated Tangible and Active Haptic Interfaces

3.3.1 Actuated Tangible User Interfaces

Tangible and graspable user interfaces are promising, but these systems often face a challenge of digital-physical discrepancy — the physical manipulation can change the digital representation, but the digital computation cannot change the physical representation of passive and static objects. For example, consider SandScape [Ishii et al., 2004]. While the projected image can dynamically change when the user reshapes the terrain,

but when the computer model is updated, such as simulating the creation of rivers or showing different mountains, it cannot change the physical shape. This division works well as long as all changes to the model are formed by the users hands, but the coupling breaks down, when the computer tries to update its model.

To address this limitation, a growing number of researchers have been exploring actuated tangible user interfaces, as the next logical step. In actuated tangible user interfaces, physical objects are not merely augmented with digital overlays but are themselves dynamic and self-reconfigurable, so that they can change their physical properties to reflect the state of the underlying computation [Poupyrev et al., 2007].

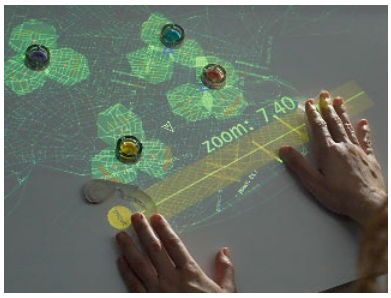


Figure 3.10: PICO [Patten and Ishii, 2007].

For example, PICO [14] use an array of electromagnetic coils for the 2D movement of tracked tangibles underneath a table to computationally move magnetic tokes on top, thus tokens can move their position dynamically to synchronize the position of virtual contents. Similarly, Madgets [Weiss et al., 2010] extends this approach with multi-functional tokens that can be moved, rotated and have their physical state altered using driven by a magnet array. Other techniques for actuation include ultrasonic waves [Marshall et al., 2012], magnetic levitation [Lee et al., 2011a], actuated magnets [Bailly et al., 2016; Lee et al., 2011b], and wheeled and vibrating robots [Nowacka et al., 2013].

Poupyrev et al. [15] present an overview of actuated tangible user interfaces. They define actuated interfaces as *interfaces in which physical components move in a way that can be detected by the user*, and classify a possible type of movements as 1) change in spatial position, change in the speed of motion, 3) change in surface texture, and 4) change in force applied to the user.

As physical objects move and transform, the border between

[14] Patten, J. and Ishii, H. (2007). Mechanical constraints as computational constraints in tabletop tangible interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 809–818. ACM

[15] Poupyrev, I., Nashida, T., and Okabe, M. (2007). Actuation and tangible user interfaces: the vaucanson duck, robots, and shape displays. In *Proceedings of the 1st International Conference on Tangible and embedded interaction*, pages 205–212. ACM

robots and actuated tangibles blur. However, actuated tangibles have a stronger focus on the user interaction aspect, whereas robots are often meant to be a mechanical worker. Poupyrev et al [Poupyrev et al., 2007] discuss actuated tangibles that can expand the application space to the following five areas, some of which are not well explored in robotics: 1) aesthetics, 2) information communication, 3) mechanical work, 4) controls — data consistency, and 5) people-to-people communication.

Figure 3.11: LivingDesktop [Bailly et al., 2016].

3.3.2 Active Haptic Interfaces

One important category of actuated tangible interfaces is haptic interfaces. These active devices, including hand-held controllers [Choi et al., 2017] and table-top surfaces [Iwata et al., 2001], can simulate the haptic sensation of virtual objects. Recently, active haptic interfaces have attracted attention, particularly to provide haptics for virtual reality and telepresence applications [Stone, 2000]. These devices are used to simulate the tactile sensation of virtual textures [Benko et al., 2016], simulate the shape of virtual objects [16], and to move passive proxy objects encountered-type haptic feedback [He et al., 2017b].

3.4 Shape-changing Interfaces

One interesting aspect of actuated tangible interfaces is their ability to change physical shape to present information or provide affordances. Shape-changing interfaces have been developed to focus on this aspect for user interaction [Coelho and Zigelbaum, 2011; Rasmussen et al., 2012]. Shape-changing interfaces are defined as a class of interfaces that 1) use the physical change of shape or change in materiality as input and/or output, 2) are interactive and computationally controlled, 3) are self-actuated and/or user-actuated, 4) convey information,

[16] Siu, A. F., Gonzalez, E. J., Yuan, S., Ginsberg, J. B., and Follmer, S. (2018). Shapeshift: 2d spatial manipulation and self-actuation of tabletop shape displays for tangible and haptic interaction. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 291. ACM

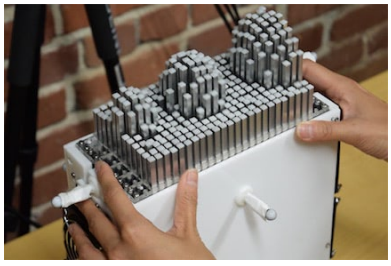


Figure 3.12: ShapeShift [Siu et al., 2018].

meaning, or affect [Alexander et al., 2018]. These interactive interfaces can change their shape to display information [Coelho and Maes, 2009], provide affordances [Hemmert et al., 2010], inform the state [Kim et al., 2008], and express emotion [Togler et al., 2009]. Change of shape can be either through shape transformation of a single material [Niiyama et al., 2014] or overall shape transition with a collective element [ART+COM, 2008]. In this sense, shape-changing interfaces focuses more on output aspect of actuated objects, compared to previously described actuated tangible interfaces.

The technology to achieve shape changes varies from mechanical actuation [17], electromagnetism [Patten and Ishii, 2007], to natural airflow [Ishii et al., 2001]. Particularly, in recent years, pneumatically-actuated soft materials have attracted attention [Follmer et al., 2012; Niiyama et al., 2015; Ou et al., 2016; Yao et al., 2013]. Unlike traditional rigid materials, soft materials have a unique advantage of rich and expressive deformation capability, which contributes to the ergonomics, functionalities, and aesthetics of such an interface.

For example, PneUI [18] is one of the earliest explorations of this class of interfaces. These shape and stiffness changing capabilities promise many different applications, such as shape displays (e.g., Colorise [Fujii et al., 2018]), interactive toys (e.g., FoamSense [Nakamaru et al., 2017], SqueezePulse [He et al., 2017a]), haptic interfaces for VR (e.g., ForceJacket [Delazio et al., 2018] and PuPoP [Teng et al., 2018]), and accessibility (e.g., Soft Exosuit [Asbeck et al., 2015]).

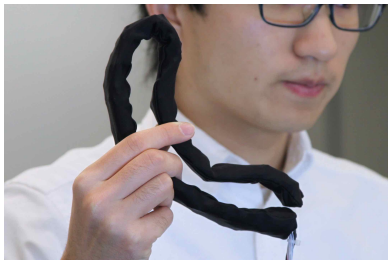


Figure 3.13: LineFORM [Nakagaki et al., 2015].

[17] Nakagaki, K., Follmer, S., and Ishii, H. (2015). Lineform: Actuated curve interfaces for display, interaction, and constraint. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 333–339. ACM

Figure 3.14: PneUI [Yao et al., 2013].

[18] Yao, L., Niiyama, R., Ou, J., Follmer, S., Della Silva, C., and Ishii, H. (2013). Pneui: Pneumatically actuated soft composite materials for shape changing interfaces. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 13–22, New York, NY, USA. ACM

3.4.1 Shape Displays

Another important category of shape-changing interfaces is shape displays. Shape displays present a physical form using shape-changing surfaces. By using an array of actuated pins,

shape displays can create an arbitrary 2.5D shape by computationally changing the height of each pin, similar to computationally changing the color of each pixel of a graphical display. One potential that shape displays have is its general-purpose shape transformation capability. Unlike other shape-changing interfaces, shape displays can represent a physical shape without a limitation of the original shape.

Thus, they can be used for many applications, such as haptic displays that can render visual objects physically [Siu et al., 2018; Iwata et al., 2001], information visualization to show data in tangible 3D bar graphs [Taher et al., 2015], geographic visualization to simulate a terrain [Leithinger and Ishii, 2010], and construct a shape by actuating passive blocks [Schoessler et al., 2015].

inFORM [19] demonstrates how shape displays can present affordances dynamically by creating various interface objects such as buttons and sliders, as well as actuating existing objects on the display. Built on top of this, Follmer provides a conceptual foundation for *dynamic physical affordances* [20]. Vink et al. further illustrates how these dynamic physical affordances can support various scenarios for our everyday activities through adaptive shape-changing furniture [Vink et al., 2015]. Recent development of shape displays also promise for mobile and graspable [Jang et al., 2016], self-movable [Siu et al., 2018], higher foce [Nakagaki et al., 2019] and higher-resolution shape displays [Zhang and Follmer, 2018].

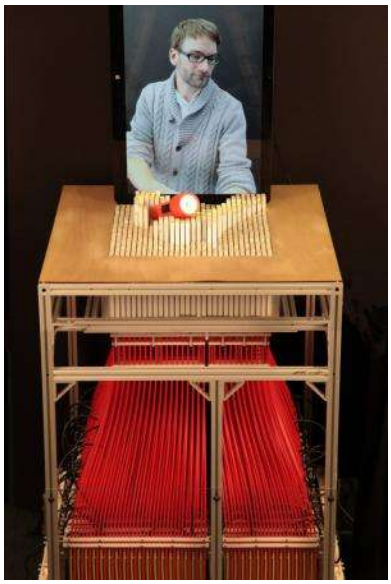


Figure 3.15: inFORM [Follmer et al., 2013; Leithinger et al., 2014].

Figure 3.16: Kinetic Blocks [Schoessler et al., 2015].

[19] Follmer, S., Leithinger, D., Olwal, A., Hogge, A., and Ishii, H. (2013). inform: Dynamic physical affordances and constraints through shape and object actuation. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 417–426, New York, NY, USA. ACM

[20] Follmer, S. S. W. (2015). *Dynamic physical affordances for shape-changing and deformable user interfaces*. PhD thesis, Massachusetts Institute of Technology

3.4.2 Large-scale Shape-changing Interfaces

Recent shape-changing or actuated tangible user interfaces have a great potential to augment this static physical environment into a dynamic one. But most current shape-changing research focuses on the scale of a human hand. Some other

work also investigated larger-scale (e.g., body-scale, room-scale, building-scale) shape-changing interfaces as an adaptive environment or architectural robot [21].

For example, the Muscle Tower project [Oosterhuis and Biloria, 2008] is a tall, open structural tower made from aluminum tubes and pneumatic muscles, which can engage a passer-by as it leans and bends towards the moving person. Similar to Muscle Tower, Dress Room [Vallgårda, 2014] responds to the location/position of an inhabitant, measured through pressure sensors integrated into the floor. Open Columns [Khan, 2010] and ExoBuilding [Schnädelbach et al., 2010] are ceiling-mounted actuators that respond to variations in carbon-dioxide levels in interior spaces or the user's physiological data. These works are designed to map data into architectural space as an ambient but physical data representation. For more general-purpose information display through shape change, Hypo-surface [Goulthorpe, 2006] demonstrated one of the earliest architecture-scale shape displays. More recently, MegaFaces [Khan, 2014], Tangible Pixels [Tang et al., 2011], and Tile-PoP [Teng et al., 2019] propose a similar concept by using mechanical linear actuator arrays.

[21] Gross, M. D. and Green, K. E. (2012). Architectural robotics, inevitably. *interactions*, 19(1):28–33

Figure 3.17: TilePoP [Teng et al., 2019].

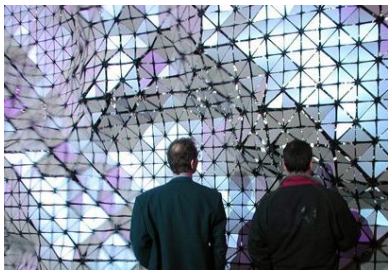


Figure 3.18: Aegis HypoSurface [Goulthorpe, 2006].

3.4.3 Modular Shape-changing Interfaces

One of the challenges of general-purpose shape-changing interfaces is the complexity of the system. In particular, shape displays often require large, heavy, complex mechanical devices that are difficult to carry or embed in the environment. By taking inspiration from the modular robotic research [Yim et al., 2007], recent research in HCI has also applied for modular shape-changing interfaces. For example, ShapeClip [Hardy et al., 2015] allows a designer to construct different geometries of shape-changing interfaces. Changibles [Roudaut et al., 2014] and Cubimorph [Roudaut et al., 2016] are shape-changing

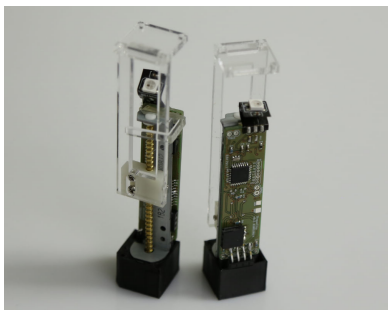


Figure 3.19: ShapeClip [Hardy et al., 2015].

robots that leverage a modular and reconfigurable design to achieve different geometries. ChainFORM [Nakagaki et al., 2016] integrates modular sensing, display, and actuation to enhance interactions. Programmable building blocks like Topobo [Raffle et al., 2004] can also be seen as modular shape-changing interfaces.

Outside the context of HCI, there is also a rich body of modular robots that can be used for user interfaces. For example, Robot Pebbles [Gilpin et al., 2010] are 1cm size robots to create a stable object with electro-permanent magnets, M-Block [Romanishin et al., 2013], Roombots [Sprowitz et al., 2010] and Flight Assembled Architecture [Augugliaro et al., 2014] can construct furniture or buildings with multiple modular robots. This work also takes inspiration from these works and expands the potential of modular, collective elements as general-purpose shape-changing interfaces.

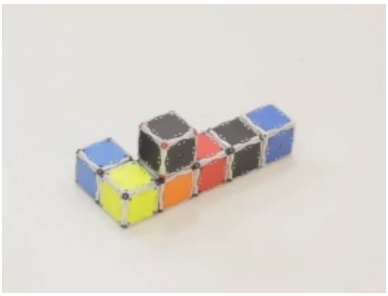


Figure 3.20: M-Block [Romanishin et al., 2013].

Figure 3.21: Zooids: Swarm User Interfaces [Le Goc et al., 2016; Kim and Follmer, 2017].

3.4.4 Swarm User Interfaces

Finally, recent research explores a new class of user interfaces that exploits a swarm of robots and actuated elements. Swarm user interfaces (Swarm UIs) are proposed as novel dynamic tangible interfaces that leverage many collective movable physical elements (e.g., 10-30) for interactions [22]. Swarm user interfaces have several advantages over existing shape-changing interfaces. For example, swarm UIs are not constrained in a specific place as they can move freely on a surface [Le Goc et al., 2016; Suzuki et al., 2018a], on a wall [Kim and Follmer, 2017], on body [Dementyev et al., 2016], or even in mid-air [Braley et al., 2018]. Also, a swarm of robots provides scalability in shape change as it comprises many interchangeable elements. The number of elements can also be flexibly changed which contributes to both scalability and expressiveness of displaying information.

[22] Le Goc, M., Kim, L. H., Parsaei, A., Fekete, J.-D., Dragicevic, P., and Follmer, S. (2016). Zooids: Building blocks for swarm user interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 97–109. ACM

Swarm UIs transform their overall shape by collectively rearranging individual, usually identical units. The ability to heterogeneously transform individual shapes can expand the range of expressions, interactions, and affordances. This can be useful in many tasks that current swarm UIs support, such as geometric expressions [Le Goc et al., 2016], iconic shapes and animations [Kim and Follmer, 2017] (e.g., animated arrow shape), education and scientific visualizations [Özgür et al., 2017] (e.g., visualization of magnetic fields or sine waves), physical data representations [Le Goc et al., 2019] (e.g., line graphs, bar graphs, network graphs), accessibility [Guinness et al., 2018] (e.g., tactile maps), and tangible UI elements [Patten and Ishii, 2007; Patten, 2014] (e.g., a controller and a slider). Inspired by this work, my work also contributes to this line of work by exploring broader design space and illustrating the potential use through scenarios.

3.5 Analysis of Related Work

Many of the existing works in shape-changing interfaces are motivated and inspired by Sutherland's vision of Ultimate Display [Sutherland, 1965], Toffoli's Programmable Matter [Toffoli and Margolus, 1991], Goldstein's Claystronics [Goldstein and Mowry, 2004], and Ishii's Radical Atoms [Ishii et al., 2012] — "a room within which the computer can control the existence of matter" [Sutherland, 1965] and user interaction with "a new kind of matter capable of changing form dynamically" [Ishii et al., 2012].

Figure 3.22: Radical Atoms Vision [Ishii et al., 2012].

Although it is challenging, there are many promising applications, as we reviewed. Particularly, general-purpose shape-changing interfaces, that can transform into any kind of shapes have a large potential. Unlike shapes that can only change

into only one or two states, general-purpose interfaces can work as a physical display, that can transform into arbitrary shapes. We can see this as an analogy of a graphical display. Graphical display has revolutionized the interface paradigm by enabling many possible applications. The transition from the 9-digit display or character-based screen to general-purpose bit-mapped displays that can simulate arbitrary 2D graphics enables the new paradigm of computer interfaces (i.e., graphical user interfaces), and significantly expands what we can do with computers.

In the same way, dynamic physical displays also could provide a foundation for many potential applications that can be built on top of it, such as enabling a tangible information display [Poupyrev et al., 2004], physicalizing data [Taher et al., 2015], providing general-purpose haptic sensation [Iwata et al., 2001], and providing dynamic physical affordances [Follmer et al., 2013; Vink et al., 2015].

In history, physical shape displays are mostly demonstrated through an array of actuated pins that can dynamically create a shape out of tabletop surfaces. As we review, its history started from the 1990s [Iwata et al., 2001], and since then, significant advances have been made to make a shape with the faster and higher resolution of the pin displays. (Leithinger provides an overview of the history of the development of shape displays [23].)

[23] Leithinger, D. (2015). *Grasping information and collaborating through shape displays*. PhD thesis, Massachusetts Institute of Technology

However, the current systems and design architectures have several limitations. First, they often require a large, mechanically complex devices that are difficult to embed in existing environments. They tend to have a large space underneath the table to store pins, thus the device often becomes a large and heavy table. It is similar to the graphical display that used to be very large, thick, and heavy a cathode-ray tube display. As

displays become more portable, compact, and mobile, the application domains can be much expanded for many different scenarios. Second, the current design of shape display cannot create a graspable shape. This is because the actuated pins are fixed and embedded in a table, thus the generated shape is attached to the table and cannot grab wit hands. It also limits the possible interaction with shape displays, as the user cannot easily pick and move the generated shape. This is why many of the existing works often use a token (e.g., red ball in inForm [Follmer et al., 2013]) as a proxy for controller or pointer when interacting with the shape display.

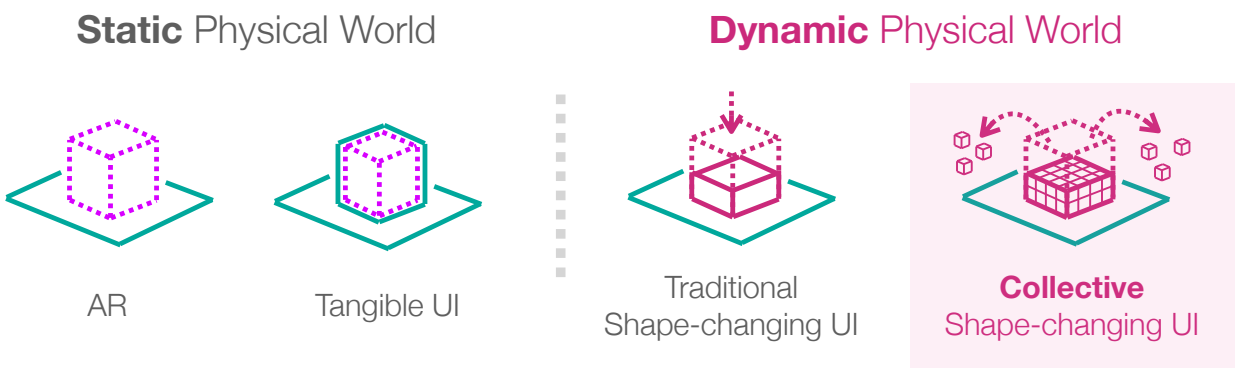


Figure 3.23: The scope and relationship between existing concepts of augmented and dynamic physical interfaces. The focus of this thesis is on dynamic physical interfaces constructed from discrete collective elements.

In this thesis, I aim to explore how we can construct a dynamic shape with different approaches. More specifically, this thesis explores the dynamic shape construction and transformation with collective discrete elements. Using discrete collective elements can potentially address the limitations stated above. For example, collective elements enable a simple, modular design, which can decompose large, monolithic devices into a set of simple, distributed elements. Thus, it could simplify the mechanically complex device and make the device more distributed and deployable to the existing environments. Also, these discrete elements can serve graspable objects or tokens, that allow the user to interact with a rich set of interactions, such as grasping, picking, moving, and playing with hands.

On the other hand, it does not reduce the expressivity and general-purposes of the display. Since it consists of collective elements, the constructed shape is not bound to one or two states. By reconfiguring the constructed elements, it is possible to make a rich variety of shapes that are not limited to the original shape.

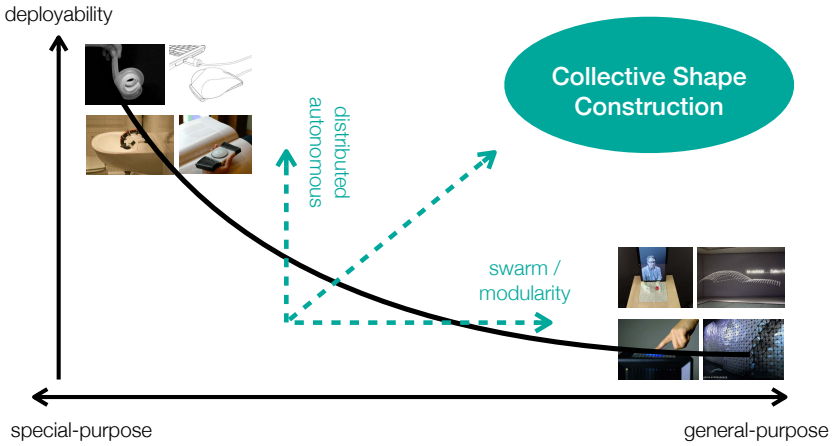


Figure 3.24: Collective shape-changing interfaces can achieve both deployable and general-purpose shape transformation by leveraging a swarm of modular elements.

The idea of making shapes with discrete elements is not new. There is much existing research in robotics and mechanical engineering. For example, there is a long history of modular self-reconfigurable robots and digital materials to explore this idea for general-purpose shape construction and transformation. However, applying these ideas for interactive user interfaces presents a different set of challenges and requirements. For example, despite the decades of research, these approaches are rarely seen in interactive systems, as it suffers from slow transformation speed. In existing systems, the overall shape change often takes minutes to hours. This is not acceptable for an interactive system, as real-time interaction happens in seconds, not minutes or hours. Also, it often does not consider the human aspects, such as how we can interact with the constructed objects. This is also a unique aspect of interactive devices. Moreover, existing work often provides a point solution, but reviewing and analyzing different approaches can

give us a good understanding and inspiration to explore a new type of physical displays that are currently not well explored.

Therefore, in the next chapter, I will provide a foundation of dynamic shape construction and transformation with collective elements by exploring the design space representations of shape and methods to achieve shape constructions for each approach. These explorations allow us to tap into broader design space to achieve the dynamic shape display and further explore alternative approaches.

4

Dynamic Shape Construction with Collective Elements

"We live in a three-dimensional world, but displays and printers restrict information to two-dimensional surfaces. [...] These problems can all be fixed by dismantling the real barrier, the one between digital information and our physical world. We're made out of atoms, and will continue to be for the foreseeable future. All of the bits in the world are of no use unless they can meet us out here on our terms. The very notion that computers can create a virtual reality requires an awkward linguistic construction to refer to "real" reality, what's left outside the computer. That's backward. Rather than replace our world, we should first look to machines to enhance it."
— Neil Gershenfeld [1]

[1] Gershenfeld, N. (1999). *When things start to think.* Henry Holt and Co., Inc

This chapter introduces dynamic shape construction with collective elements. The scope of this thesis specifically focuses on shape construction with discrete elements, rather than monolithic materials. Each individual element can dynamically change its shape, position, and any other physical property

through either internal or external actuation, such that they can collectively construct and transform their overall physical shape dynamically.

The promise of collective shape construction is its generalizability for information display — in contrast to single-purpose shape transformation, it has potential for universal shape construction and transformation into an arbitrary physical shape, just like pixels on a graphical display.

This chapter first reviews the possible representations of dynamic physical shape and the transformation method of each representation. I discuss the benefits and limitations of each representation in terms of the flexibility of shape construction and from a user interaction perspective. Then, I explore the design space of constructing elements and draw a continuum between the shape constructed by active self-actuated elements and the shape constructed by passive externally-actuated elements. I also discuss the trade-offs of each approach and show that it is possible to make dynamic shape construction with the passive collective elements.

4.1 Design Space of Shape Representation

The goal of the dynamic physical display is to represent digital data as a tangible shape. However, representing data is not restricted to a single method, rather there are various ways of representing information. For example, in data visualization, the same numerical data can be represented in various forms, such as bar graphs, line graphs, pie charts, treemaps, scatter plots, and box plots, etc. Each of these graphs embody the same information, but the way of visualizing the information is different. Similarly, a shape — either physical or graphical — can also have various ways to represent geometric information.

In computer graphics, for example, the display can represent the same 3D object with either spatially distributed points like a point cloud, or with a stack of sliced layers, a set of voxels, a collection of vertices, edges, and faces in polygon mesh, and a set of lines or curves in a wireframe. Again, all of these representations present the same shape information, but different representations have advantages and limitations in terms of use cases and applications.

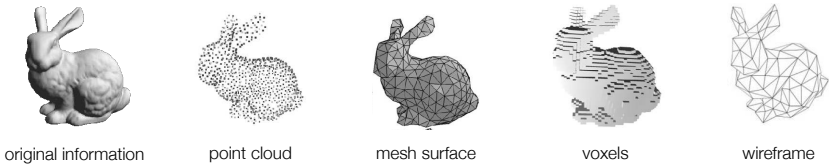


Figure 4.1: Different types of shape representations in computer graphics

This is also true for physical shape representation constructed by discrete elements. There is not one way to construct a physical shape, rather there are various forms of shape creation. For example, consider creating a static shape, say Stanford Bunny, with a collection of physical elements. We can think of various ways to construct the Stanford Bunny. For example, one can make the Stanford Bunny by assembling the LEGO blocks, similar to the voxel representation in 3D models. Or, one can make the object by stacking sheets of paper or plywood layer-by-layer to create an object using a tool like Autodesk 123D Make. Alternatively, one could make the shape with a connected hub and struts using a construction kit like Zometool, similar to wireframe representation. It is also possible to create the shape as a 2.5D relief of a pin art toy like Fleming's Pinscreen toy or as a 3D drawing of a wire bending art or 3D doodlers. It is even possible to create an arbitrary 3D shape by folding a sheet of paper like Origami to construct the geometry [Demaine and Tachi, 2017].

As we can see, there are many ways to construct a physical

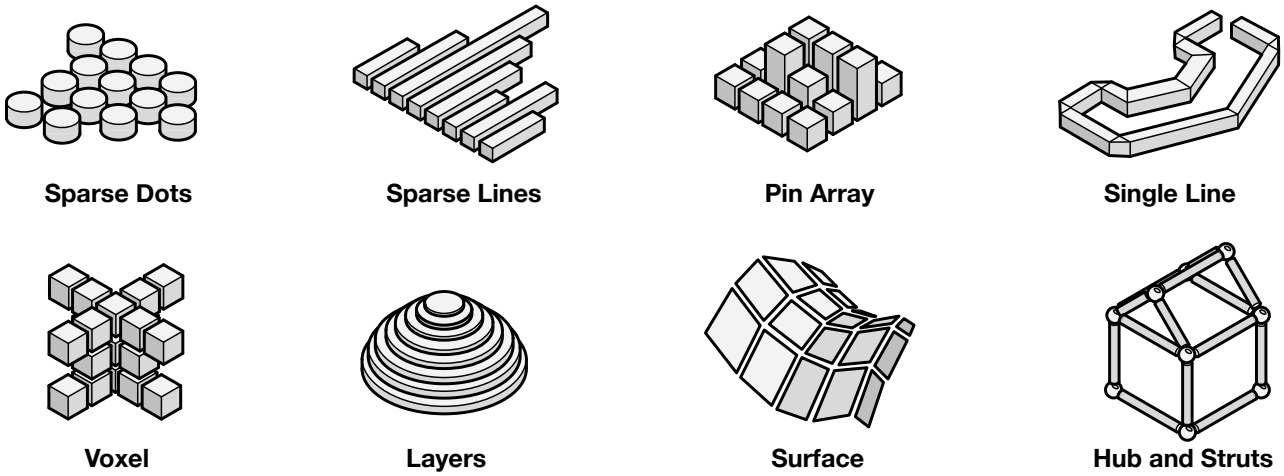


Figure 4.2: Different types of shape representations

shape. Therefore, it is an interesting to explore *what types of shape representations are possible* with dynamic collective elements. Although the literature of physical shape-changing displays mostly focuses on representations with an array of actuated pins, I can identify at least eight different types of representations that can be used to create geometries, as depicted in Figure 4.2: 1) Sparse Dots, 2) Sparse Lines, 3) Pin Array, 4) Single Line, 5) Voxel, 6) Layers, 7) Surface, 8) Hub and Struts. Some representations are previously well explored, but some are not.

To better understand how each approach constructs a shape, again examine the construction of the Stanford Bunny. Sparse dots create a shape by spatially distributing the collective elements in a space or on a surface, similar to point cloud representation. By individually changing the position, the overall shape can also transform. Sparse lines are a similar representation, but they utilize fixed or transformable lines to make a shape, similar to wireframe representation. The pin array creates a shape similar to a pin art toy, by individually transforming each pin to generate a relief shape on a surface. The single line makes a shape by bending each line segment. The

voxel constructs a shape out of collective blocks or lattice structure, and the layer creates a shape out of a transformable sheet of layers. The surface constructs a shape by bending or transforming each polygonal element that composes the surface. And, the hub and struts create shapes by transforming or constructing lines of each mesh.

Note that I do not argue that this is the only way to classify the possible representations, and I certainly think there might be other types of classifications or missing shape representations. However, this is at least useful to investigate each approach and discuss the benefits and disadvantages. In this classification and presented examples, I focus on shape construction with a collection of *discrete* elements, as this is the specific focus of this thesis. For each representation, I also illustrate possible discrete elements as well as an example project. By collectively moving, transforming, or changing their physical property, these elements can construct an overall physical shape. Here, I describe each representation in more detail.

4.1.1 Sparse Dots Representation

Sparse dots refer to a shape category that consists of spatially aligned dot elements. Sparse dots represent shapes in a similar manner as a point cloud or as dot graphics in computer graphics.

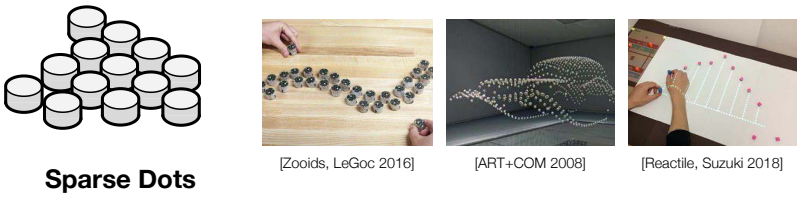


Figure 4.3: Sparse dots representation. (e.g., Zooids [Le Goc et al., 2016], BMW Kinetic Sculpture [ART+COM, 2008], and Reactile [Suzuki et al., 2018a]).

Similarly, these physical shapes can be represented on a 2D surface [Le Goc et al., 2016; Alonso-Mora et al., 2012; Ruben-

stein et al., 2014] or in 3D space [ART+COM, 2008; Braley et al., 2018], and each element can be self-actuated [Braley et al., 2018; Kim and Follmer, 2017] or externally actuated with environmental force or actuated objects [ART+COM, 2008; Ochiai et al., 2014; Omirou et al., 2015; Suzuki et al., 2018a].

The methods of dynamically constructing and transforming sparse dot shapes are 1) free movements of elements in the 2D surface, 2) constrained movements of elements in a vertical direction, 3) free movements of elements in 3D space. The shape of sparse dots can be presented through either dense objects, contours dots, or vertex. Since each element of sparse dots can freely move in space, it has the benefit of the dynamicity of overall shape transformation. However, the overall shape is not physically connected, thus the shape is represented through visuals.

4.1.2 Sparse Lines Representation

Similarly, sparse lines consist of a shape with spatially aligned line segments. Sparse lines are also represented on a 2D surface [Suzuki et al., 2019b] or in 3D space [Torres, 2014; Leder and Weber, 2018; Leder et al., 2019].

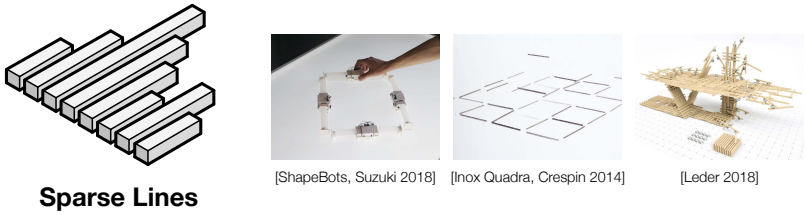


Figure 4.4: Sparse lines representation. (e.g., ShapeBots [Suzuki et al., 2019b], 4Net Inox Quadra [Crespin, 2014], and Robotic Timber Construction [Leder and Weber, 2018].)

And, sparse lines share similar properties with sparse dots. For example, because the line elements can freely move in 2D or 3D space, it is also easy to dynamically transform the overall form from one shape to another [Crespin, 2014]. Compared to sparse dots, sparse lines can construct more visually expres-

sive shapes, such as contour lines. Also, by leveraging the mechanical property of lines, such interfaces can provide unique affordances with mechanical constraints.

The methods of transformation can be 1) free movements of line segments in the 2D surface, 2) free or constrained movements of line segments in 3D space, 3) extension or contraction of the length of each line segment.

4.1.3 Pin Arrays Representation

Pin arrays are a shape construction method with transformable pin elements. As mentioned, pin array is one of the most common ways for 2.5D shape displays in the literature of shape-changing displays. Although in the traditional shape display, each actuated pins is not a discrete element and is instead fixed in a table, it is possible to decompose each actuated pin into discrete collective elements (e.g., ShapeClip [Hardy et al., 2015] and ShapeCanvas [Everitt et al., 2016], Lift-Bit [Ratti, 2016]).

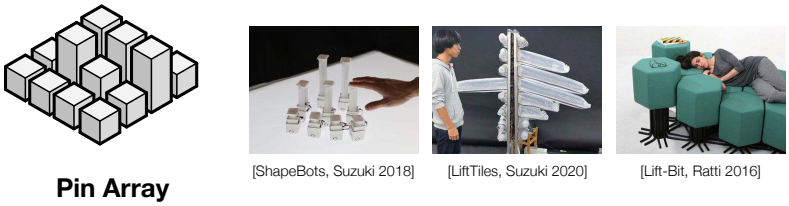


Figure 4.5: Pin array representation. (e.g., ShapeBots [Suzuki et al., 2019b], LiftTiles [Suzuki et al., 2020b], and Lift-Bit [Ratti, 2016].)

The methods to construct and transform shapes with pin arrays are 1) vertical extension of each element, 2) horizontal extension each element, 3) horizontal or vertical movements of each element. Particularly, the third element is unique to the pin arrays with discrete elements, as each element can also change its position in a 2D horizontal or vertical surface (e.g., ShapeBots [Suzuki et al., 2019b], LiftTiles [Suzuki et al., 2020b]).

4.1.4 Single Line Representation

A single line is an approach to constructing a shape by computationally bending a line. For example, a single line can represent a shape with either an iconic contour shape (e.g., phone shape in [Nakagaki et al., 2015]) or with a volumetric shape (e.g., a cube in [Yim and Sitti, 2014]). Griffith shows that it is possible to construct any arbitrary 3D shape with a continuous single line [Griffith, 2004], and recent work demonstrates that a single 3D printed chain can be transformed into various 3D shapes — such as the Stanford bunny — by manually assembling it [Yu et al., 2019].

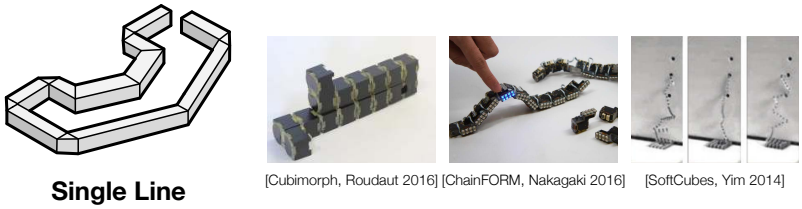


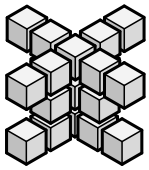
Figure 4.6: Single line representation. (e.g., Cubimorph [Roudaut et al., 2016], ChainFORM [Nakagaki et al., 2016], SoftCubes [Yim and Sitti, 2014])

The methods to construct and transform the shape with a single line are 1) change of the bending angle of each line segment, and 2) change of the length of each line segment. In this representation each segment is discrete, so the overall shape can be reconfigured through the properties of each discrete element (e.g., ChainForm [Nakagaki et al., 2016]). Similar to other representations, each line segment can either be self-actuated [Murata et al., 2002; Nakagaki et al., 2016, 2015; Roudaut et al., 2016] or externally programmed or actuated [Yim and Sitti, 2014; Griffith, 2004].

4.1.5 Voxel Representation

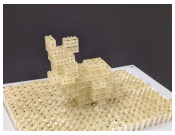
A voxel is a representation of a shape with connected blocks or a lattice structure. The voxel-like structure is one of the most common ways of dynamic shape construction, particularly in the literature of self-reconfigurable modular robots (e.g., M-

Block [Romanishin et al., 2013], Roombots [Sproewitz et al., 2009]). In this representation, each voxel element is physically connected with neighboring elements, so that the overall shape is structurally stable when holding it (e.g., Robotic Assembly of Zooids [Zhao et al., 2017], Dynablock [Suzuki et al., 2018b]).

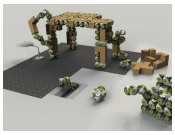


Voxel

[Aerial Assembly, Tibbits 2014]



[Dynablock, Suzuki 2018]



[Roombots, Sproewitz 2009]

Figure 4.7: Voxel representation. (e.g., Aerial Assembly [Tibbits et al., 2014], Dynablock [Suzuki et al., 2018b], Roombots [Sproewitz et al., 2009])

The methods to construct and transform shapes using voxels are 1) movement of each voxel element, 2) change in connection with neighboring elements, and 3) transformation of each voxel element. In self-reconfigurable modular robots, many approaches explore the active voxel elements with self-actuation capability. It is also possible to transform its shape with externally or environmentally actuated force, such as swarm robotic construction [Petersen et al., 2011; Werfel and Nagpal, 2008], swarm robotic arms [Jenett and Cheung, 2017], external robotic arms [Sekijima and Tanaka, 2015; Hiller, 2011], external force [Tibbits et al., 2014; Papadopoulou et al., 2017], and actuation with shape displays [Schoessler et al., 2015; Suzuki et al., 2018b].

4.1.6 Layer Representation

A layer is an approach to construct a shape using a stacked layer. Layer-by-layer is a well-known method to construct an arbitrary shape with additive manufacturing. To my knowledge, I was not able to find any general-purpose shape construction and transformation that uses this method, but it should be possible to achieve a dynamic shape-changing display with transformable layers. For example, Additive Fold-

ing [Yim et al., 2018] demonstrates shape construction with pre-fabricated sheets and connects each layer with a string to create a transformable shape. By dynamically changing the shape of each layer, it is also possible to change the overall shape dynamically (e.g., BendingArches [Winther and Vallgårda, 2016]).

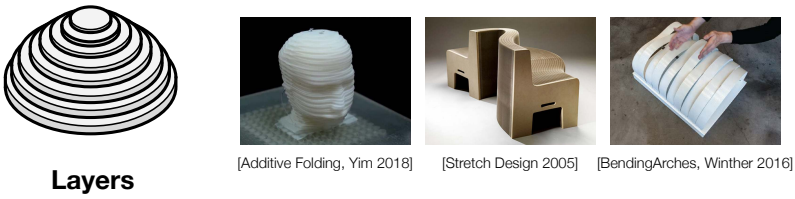


Figure 4.8: Layer representation. (e.g., Additive Folding [Yim et al., 2018], Stretch Design [StretchDesign, 2005], and BendingArches [Winther and Vallgårda, 2016])

The methods to construct and transform the shape with the layered approach are 1) change the shape of each layer, 2) change the thickness of each layer, and 3) change the spacing between each layer. For example, by changing the spacing between each layer, expandable chairs like Flexible Loveby by Stretch Design [StretchDesign, 2005] demonstrate transformable furniture or sculptures. Gronvall et al. [Grönvall et al., 2014] investigates the interaction of such shape-changing benches. Also, Winther et al. [Winther and Vallgårda, 2016] show that shape change in each sliced layer can transform the overall shape.

4.1.7 Surface Representation

A surface is a representation of a shape using connected 2D elements.

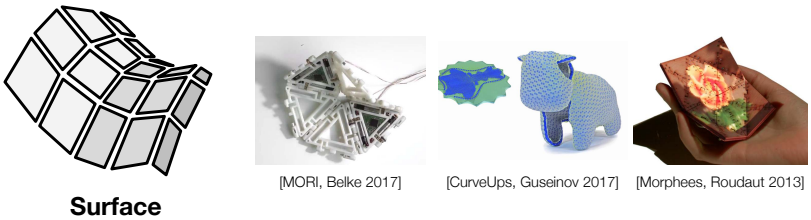


Figure 4.9: Surface representation. (e.g., MORI [Belke and Paik, 2017], CurveUps [Guseinov et al., 2017], and Morphees [Roudaut et al., 2013])

The shape of each element can be a triangle, a rectangle, or any other shape. In computer graphics, it is known as surface mesh. By changing the size or shape of each element, it is possible to transform the overall shape. Also, by changing the angle of each connecting element, a similar transformation can be possible [Guseinov et al., 2017; Roudaut et al., 2013]. For example, Mori [Belke and Paik, 2017] demonstrates that through modular origami robots each of the connections can be bent.

The methods to construct and transform shapes in surface representations are 1) change of the size of each element, 2) change of shape of each element, and 3) change of the bending angle of each element.

4.1.8 Hub and Struts Representation

Finally, the hub and struts construct a shape using connected multiple line segments. This representation is well explored in static geometry, such as a timber frame for building construction or constructive assembly toys like Zome tools.

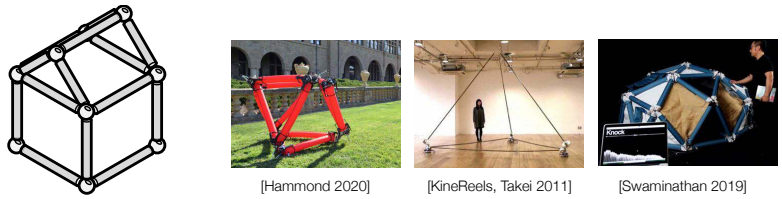


Figure 4.10: Hub and struts representation. (e.g., Untethered Isoperimetric Soft Robot [Usevitch et al., 2020], KineReels [Takei et al., 2011], and Interactive Inflatables [Swaminathan et al., 2019])

For dynamic shape construction, the hub and struts are also explored in responsive architecture [Oosterhuis and Biloria, 2008; Swaminathan et al., 2019; Takei et al., 2012, 2011], walking robots [Bondin et al., 2015], and soft robotics [Hammond et al., 2017; Hawkes et al., 2017; Usevitch et al., 2020]. Although the current implementations are mostly explored for shape transformation of the original structure, it is also possible to construct a shape with discrete reconfigurable ele-

ments [Leder et al., 2019].

The methods to construct and transform the shape using hub and struts are 1) extension of each strut element, 2) movement of the position of each hub element, and 3) connection and disconnection of a hub and a strut. Each element can be either self-actuated [Takei et al., 2012] or externally actuated [Staal, 2015; Yamaoka, 2014].

4.1.9 Analysis and Comparison of Each Representation

In theory, all of the presented representations can create any arbitrary shape given a sufficiently high resolution. But, in practice, each shape representation has its benefits and limitations. Thus it is useful to discuss these trade-offs when considering use cases and applications. Here, I investigate flexibility, graspability, scalability, and interaction capability.

Flexibility

In spatially distributed representations like sparse dots, lines, and pin arrays, each element can independently move and transform without the interference with other elements. This property allows the fast overall transformation with the massively parallel movement of each individual element. In contrast, the interconnected elements like voxel representations may require coordination with neighboring elements when transforming one shape to another, which introduces challenges for rapid and flexible shape transformation.

Graspability

On the other hand, the shape constructed with interconnected elements has its merit. For example, the constructed shape can be dealt with as a single object, thus the user can easily

grasp and hold the resulting object. It provides a more stable structure and efficient in terms of maintaining the same shape, compared to the shape constructed with spatially aligned elements, because these spatial representation needs to maintain the overall shape whenever the user interacts with it. For example, consider a square shape constructed with sparse dots. When the user picks up and moves one element, the other elements also need to follow and move by tracking the user's movement, in order to maintain the same shape. Moreover, the shape constructed with interconnected elements is also used as a static object or tool when the shape does not need to change. Overall, these representations have the benefits of making graspable and stable structures.

Scalability

The scalability is another aspect. In other words, it is related to the number of elements that require to represent the same shape. In general, the densely-arranged representation requires more elements, as opposed to the representation with contour form. For example, compare a cubic shape that is constructed with densely-aligned elements of voxels or sparse dots with the same shape constructed with contour lines of a single line or hub and struts. If each edge of the square requires ten elements, then the densely-aligned representation may require approximately a thousand elements, while the contour lines only require twelve lines. Therefore, if the shape can be easily represented with a line, such as a wireframe, contours, or iconic silhouette, it might be more efficient to use the line-based representations.

Interaction Capability

The different representation can support different interactions. For example, line-based elements like a single line, sparse line,

or hub and struts can support an expressive two hands interaction such as extending or collapsing a line, bending a line or cutting or connecting the line segments. Whereas, voxel representation may not fully support these expressive modifications, as each element may not be as plastic and malleable as the other elements. On the other hand, the voxel representation can support interactive construction like constructive building blocks. By tracking the user's construction, it can be possible to automatically complete the desired shape. Interaction with pin arrays are well explored in the previous work [Leithinger, 2015], but the user can interact with each element by pushing it. Since each element is discrete, it is also possible for users to move and reconfigure the elements to construct a new shape representation in improvisational ways. Surface representation may have a unique interaction capability of folding the entire shape, similar to origami. The combination of manual and automatic folding would be another interesting interaction. The sparse dots also support several unique interactions with both individual (e.g., pick-up and move) as well as collective elements (e.g., gather or move many elements with two hands).

4.1.10 Design Implications

Based on this design exploration, I discuss a couple of design implications that lead to my work.

Exploring New Types of Representation

One implication is that there is still a rich space that the previous works have not explored well. This includes both representation of the shape as well as the methods of construction and transformation. For example, collective shape-changing displays with sparse lines, voxels, hub and struts, surfaces, and layer representations are not well examined in the prior work. Also, some of the work mostly focuses on the shape trans-

formation aspect of the pre-defined shape (e.g., [Takei et al., 2012]). There are still interesting research opportunities to not only transform from one shape to another shape, but also construct various shapes with these elements. Moreover, as we discuss, the exploration of the different shape representations can also allow us to develop a different set of interaction techniques.

To expand the research domain, this thesis demonstrates a range of shape representations, including sparse dots, sparse lines, pin arrays, and voxels. By expanding these domains, we can now further explore what could be potential applications of these shape-changing interfaces. Also, we can explore and compare different interaction techniques with collective elements.

Combining the Transformation Building Blocks

This section reviews not only the types of representation but also methods of transformation of each element. As discussed, the transformation capability of each element can be categorized as these building blocks: 1) change in the horizontal position of elements, 2) change in vertical position of elements, 3) change in orientation of elements, 4) change in length of elements, 5) change in volume of elements, 6) change in connectability of elements, 7) change in a 2D shape of elements.

Although previous work mostly focuses on one method for each project, this implies that by combining these transformations, it is possible to cover multiple representations with single elements. For example, if elements have not only an ability to move (1-2) but also an ability to extend (4), these elements can potentially represent both sparse dots, sparse lines, and pin arrays. If the elements have both an ability to move (1-2) and connect (6), they can also represent both sparse dots and

voxel representations.

The combination of these capabilities for each element can potentially enhance the expressiveness by supporting a range of different representations. Therefore, one interesting, yet underexplored aspect in this design space is which combination of the capabilities provide an interesting opportunity and how we can combine them in one element.

In this thesis, I contribute to exploring this aspect by demonstrating the idea through shape-changing swarm robots (i.e., movement and extension). These examples can provide insight when designing these elements. Also, I discuss the other possible opportunities for future research in the Discussion section.

4.2 Design Space of Collective Elements

4.2.1 Active and Passive Elements

In the previous section, I have reviewed and explored the design space of overall shape representation. In this section, I look into the design space of each individual element that consists of a shape.

In general, there are basically two types of discrete elements that can be used to construct a shape.

- **Active Element:** an element that can change its state (e.g., shape, position, physical property, etc) by itself with internal mechanisms
- **Passive Element:** an element that cannot change its state by itself

In other words, we can see it as a self-actuated element and non self-actuated element.

At first glance, one may think that we need to leverage active, self-actuated elements if we want to continuously form and reform the dynamic shape. In fact, research in robotics has pursued this direction to achieve programmable matter [Goldstein et al., 2005; Goldstein and Mowry, 2004; Toffoli and Margolus, 1991; Yim et al., 2007]. However, when we look into the context of tangible user interfaces or shape-changing user interfaces, we also notice that many previous works do not explicitly use active elements for dynamic shape construction and transformation. For example, in Kinetic Blocks [Schoessler et al., 2015], the elements that construct a shape are just a passive static blocks, thus the constructive blocks cannot dynamically change its state. We can also see each element of kinetic sculpture (e.g., [ART+COM, 2008]) as a passive, non self-actuated element rather than an active element.

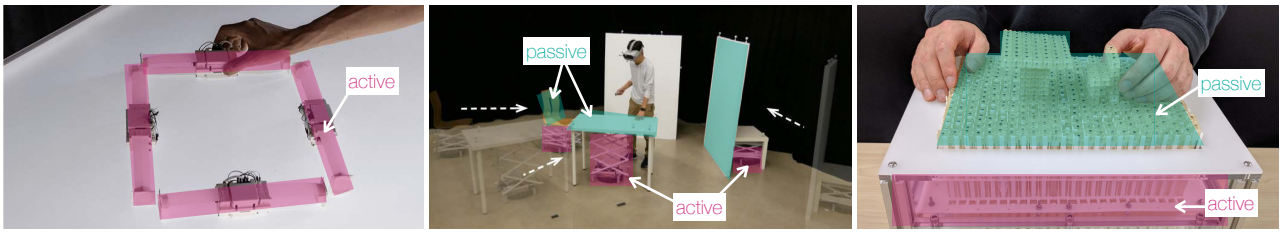


Figure 4.11: Examples of active vs passive elements from ShapeBots [Suzuki et al., 2019b], RoomShift [Suzuki et al., 2020a], and Dynablock [Suzuki et al., 2018b].

Therefore, the fact that whether the element is passive or active seems independent of the dynamicity of the overall shape. In fact, compared to these shape-changing interface examples that leverage passive elements, some of the shape creation with active robots are not as dynamic as these examples, as these robots may sometimes take minutes or hours to complete the shape transformation.

Then, what would be the factor that decides the dynamicity of the shape? And how can we increase the dynamicity?

4.2.2 The Definition of Dynamicity of the Shape

Dynamic vs Inert

To understand this puzzle, I would like to introduce the concept of dynamic vs inert elements. The notion of dynamic or inert is defined as follows:

- **Dynamic:** We call an element is dynamic when the element is ready to change its state (e.g., shape, position, physical property, etc). In other words, the element is currently computationally controllable at a moment.
- **Inert** We call an element is inert when the element is not ready to change its state. In other words, the element is not currently computationally controllable

The Definition of Dynamicity

I argue that the dynamicity of the shape can be defined and measured by *how many numbers of elements in a shape are currently dynamic*. For instance, a sparse dot shape constructed by swarm robots can be highly dynamic because all of the constructing elements are always dynamic (i.e., ready to freely move). Regardless of the elements that are passive (e.g., [Suzuki et al., 2018a]), sparse dots shape constructed by collective elements is highly dynamic.

Figure 4.12: All elements are dynamic vs only one element is dynamic. It illustrates how the number of dynamic elements can affect the dynamicity of the overall shape construction.

On the other hand, consider a shape made of the same swarm robots, but only one robot can move at once — each element can only move after the other element finish moving. In this case, only one element is dynamic, and all of the other elements are inert. As we can see in Figure 4.12, this significantly decreases the dynamicity of overall shape transformation. These configurations can often be seen in robotics applications (e.g., [Rubenstein et al., 2012]), but for the HCI applications, this loses the responsiveness of the interface.

4.2.3 Dynamic vs Inert and Active vs Passive

I also argue that the notion of dynamic vs inert is independent of whether the element is active or passive. As we can see in the above scenario, active, self-actuated robots can also become inert. There are also other situations when the active self-actuated element can become an inert element. For example, self-reconfigurable robots like M-Blocks [Romanishin et al., 2013], all blocks are active as they can change their position by themselves, but if a block is situated at the center of the shape, the element cannot physically move its position.

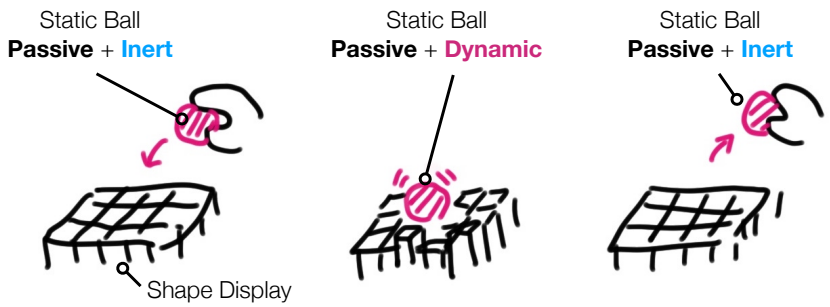


Figure 4.13: Concept of dynamic and inert states

Alternatively, these robots also become inert when their battery runs out. Also, when the user picks up a wheel-based swarm robot, but these elements become inert as it cannot move in 3D space. As we can see, these active elements can become inert because of computational constraints (e.g., a robot cannot

know where it should go until the other robots finish moving), mechanical constraints (e.g., the neighboring elements prevent the robot to move), and system constraints (e.g., the robots runs out the battery).

On the other hand, passive elements can become dynamic. For example, consider a ball placed on top of a shape display, similar to a red ball in inFORM [Follmer et al., 2013]. This ball is an inherently passive object as it does not have any internal actuation mechanism. However, once placed on top of the shape display, the object becomes dynamic, as the position of the ball can be computationally controllable as long as it is on a shape display. But, once the user grasps, it becomes inert again. In the same way, blocks assembled with the shape display [Schoessler et al., 2015] or balls connected to the ceiling motors in kinetic sculpture [ART+COM, 2008] are considered as an example of passive elements that can become dynamic because of the external actuation.

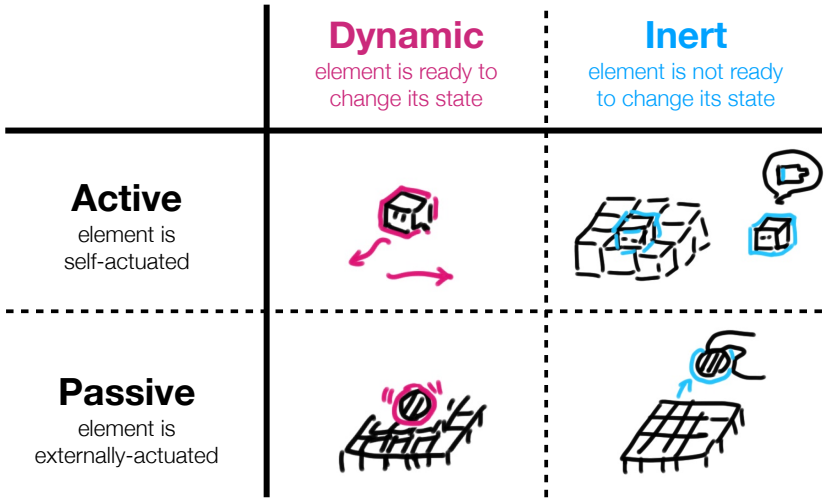


Figure 4.14: Dynamic vs inert and active vs passive.

As we can see from these examples, the notion of dynamic vs inert is independent of the notion of active vs passive, and both active and passive objects can become dynamic. To understand this difference, Figure 4.14 shows the matrix of each axis.

This illustrates when the active or passive elements become dynamic or inert. In this sense, we can see that the concept of active or passive is bound to the inherent *property* of the element, whereas the concept of dynamic or inert is bound to the *state* of the element. And for the dynamic shape construction, the dynamic elements rather play an important role to maintain the overall dynamicity, than the active elements.

4.2.4 Parallel and Collective Actuation of Passive Elements

Three Benefits of Using Passive Elements

This implies the following claim: it is possible to leverage collective passive objects to dynamically construct and transform the shape. This is particularly useful because passive elements provide unique advantages that active elements may not have. Here, I describe the three benefits of using passive elements

- **Scalability:** Active elements can maintain the constructed shape highly dynamic, but it has limited scalability. For example, since each element needs to have sensing, actuation, and power supply, the cost of each element significantly increases. In contrast, passive elements enable simple, inexpensive, easily fabricated components, thus the passive elements should be more inexpensive than the active elements. Therefore, this can significantly improve the scalability of the system.
- **Resolution:** This also contributes to the resolution of the overall constructed shape. For example, although most of the self-actuated robotic approach has a limitation in its overall number (e.g., at most 20-50 elements), using passive elements can increase this limitation, which allows us a higher resolution shape. Also, the size of each element can become small, as the passive element does not require

a space to have motors, batteries, or microprocessors. This allows us to create a finer shape.

- **Robustness:** In addition, the passive elements also contribute to the robustness and stability of the constructed object. These passive elements serve as the stable components of the shape, which can be more robust than the intricate active elements that are often made of lots of electrical and mechanical components. Also, a shape constructed with passive elements can be used as a normal object, thus it is more appropriate when the resulting object does not need to change its shape that often (e.g., building, furniture). In this case, if the shape is constructed with expensive active elements, the cost overall shape becomes unnecessarily high. Finally, if the passive elements are made of soft materials, these elements can also have a compliant feature.

By leveraging these advantages, we can expand the design space and a range of applications of the dynamic shape construction.

Then, how can we leverage the passive elements for dynamic shape construction? As illustrated in Figure 4.12, the parallel and simultaneous actuation of many elements is also key for passive collective construction. To achieve this parallel and simultaneous actuation, this thesis shows two directions: one is parallel assembly and another collective actuation.

Parallel Assembly

Parallel assembly is one of the approaches to dynamically construct a shape with passive collective elements. To understand how it works, let's consider a robotic arm that assembles passive blocks like LEGO blocks or magnetic connectable elements, similar to [Sekijima and Tanaka, 2015].

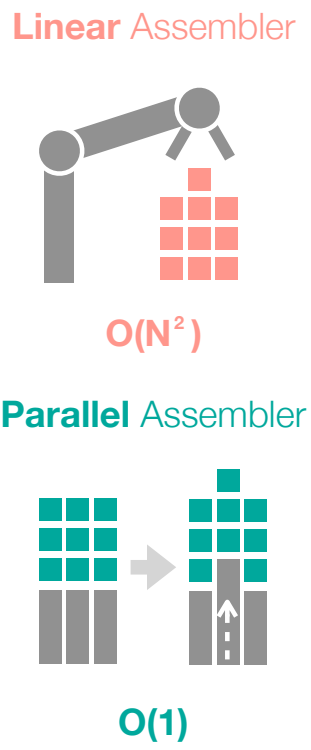


Figure 4.15: Serial vs parallel assembly.

The robotic arm can pick up one object and assemble it every time. Therefore, if the shape needs to be constructed with 100 objects, then it requires to go through this process 100 times. If each process takes considerable time, the overall time to construct a shape can take a while (e.g. if the robot are takes 3.6 seconds per element, it will take 1 hour for a 1000 of them). Thus, the shape constructed by a robotic arm can be seen as less dynamic.

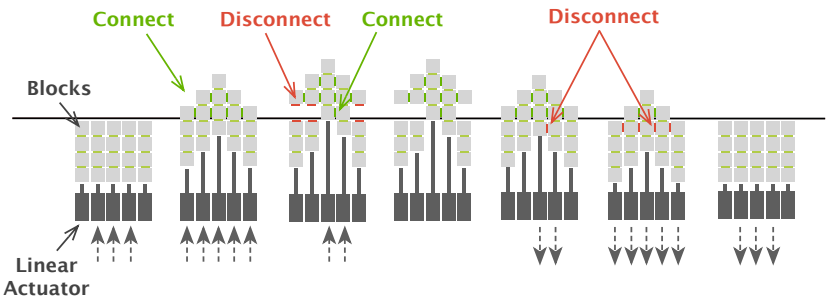


Figure 4.16: Design of parallel assembler.

Instead, parallel assembler aims to construct many objects simultaneously. To this end, I propose a new shape construction architecture using a shape display. The shape display can vertically push passive elements independently and simultaneously. By leveraging this capability, I repurpose this system for a massively parallel assembler. Figure 4.16 shows one approach to make this happen. Similar to the existing 3D printing, the shape display assembles blocks one layer each time, and then stack them layer-by-layer to construct a 3D shape with voxel representation. In this way, the parallel assembler can construct an arbitrary shape in seconds.

One interesting implication of dynamic vs inert elements is that the dynamicity of shape can also change based on the state. For example, in the above case, all of the elements in the initial state are all dynamic, as their states (e.g., position, connectability) are computationally controllable, thus the overall shape has a highly dynamic state. But, once constructed, the

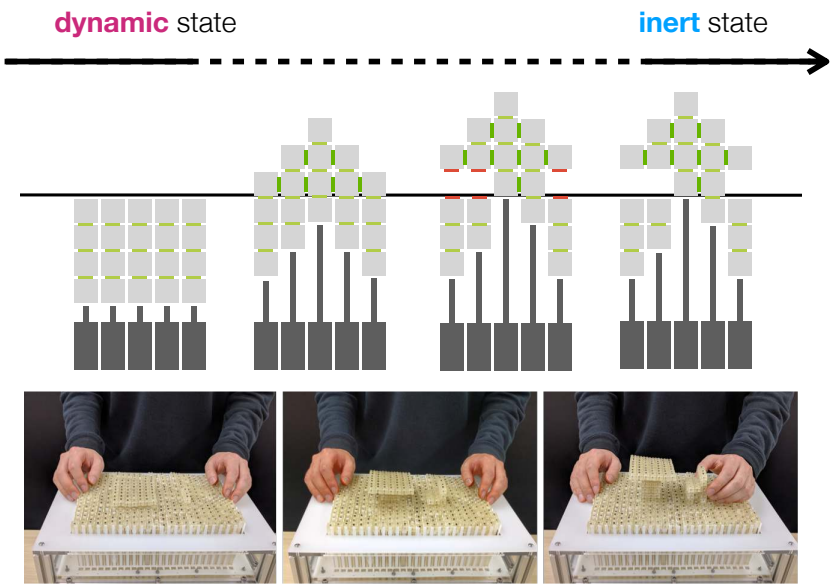


Figure 4.17: The dynamicity of shape can also change in a single system

elements that consist of the shape become inert, as they are no longer controllable with pins. Thus, the overall shape loses its dynamicity. Therefore, the approach of the parallel assembler has a certain limitation in terms of dynamic shape transformation of a constructed object. However, it is still useful to explore, as these passive building blocks promise the relatively high-resolution and scalable approach, compared to the active building blocks.

Collective Actuation

Another approach of parallel and simultaneous actuation is collective actuation. Collective actuation pursues the hybrid approach of active and passive elements — by leveraging the active elements like swarm robots to move, actuate, and construct the other passive elements collectively. To understand this idea, let's consider an alternative scenario of the previously described block assembly with a robotic arm. What if we can instead use a hundred of robotic arms to construct these objects? While each arm can handle only one object, with a hundred of robots, we can now manipulate a hundred blocks

at a time. This significantly increases the overall dynamicity of shape construction. Although there is a certain constraint of actuating all elements (e.g., some robots interfere in the movements), it could make the shape as dynamic as a parallel assembler.

In this way, there is also a rich design space of leveraging a swarm of active elements as an actuator of passive collective elements. Figure 4.18 illustrates some of the possible approaches, where black elements refer to active elements (e.g., swarm robots) and green elements refer to passive elements. For example, these swarm elements can move, conform, transform, assemble, and spatially reconfigure many passive elements collectively and simultaneously. By leveraging these techniques, it is also possible to create a dynamic shape.

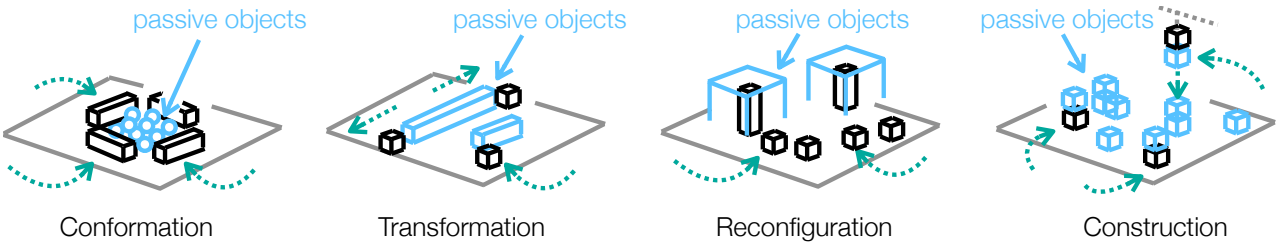


Figure 4.18: Types of actuation for passive collective elements (blue objects represent passive collective elements and black objects represent active collective elements)

Thus, how to combine with active and passive elements within a single system also becomes an important and interesting design consideration. The hybrid approach of using both active and passive elements can open up a new design space that is not previously well-explored. In this thesis, I also demonstrate this idea through swarm robotic actuation and discuss the broader actuation techniques in the discussion section.

Types of Actuation

Finally, in these parallel and collective actuation, I mostly focused on actuation using mechanical force, but there are also a rich variety of ways to actuate passive elements, such as

vibration-based force, magnetic/electro-magnetic force, electrostatic force, aerodynamic force, and acoustic radiation force. [2] These types of force can be also used to actuate passive elements. For example, in the previous work, electro-magnetic force is used in PICO [Patten et al., 2001], ZeroN [Lee et al., 2011a], and Madgets [Weiss et al., 2010], actuated magnetic force is used in Molebot [Lee et al., 2011b] and Living Desktop [Bailly et al., 2016], aerodynamic force is used in floatio [Yui and Hashida, 2016], and acoustic levitation force is used in PixieDust [Ochiai et al., 2014] and LeviPath [Omirou et al., 2015].

[2] The further comparison of these contact and non-contact force methods is also discussed in [Ochiai, 2015]

Ochiai, Y. (2015). *Graphics by Computational Acoustic Fields*. PhD thesis, The University of Tokyo

This thesis also investigates the other non-mechanical force approach for parallel and simultaneous actuation of collective passive elements. One promising method is to use electromechanical actuation, as it enables relatively strong force, while it is easier to computationally control. One drawback of the previous electromagnetic actuation systems, however, was the scalability of the system. Since these systems using an array of mechanical coils, it often introduces the fabrication complexity, which leads to the scalability problem. To address this problem, this thesis also shows a scalable and inexpensive method to actuate collective passive objects with PCB-based electromechanical actuation board. This also promises the small, easy to scale actuation for dynamic shape construction made of externally-actuated passive collective elements.

4.3 Summary

In summary, this chapter discusses the followings:

1. **Shape Representaton:** There are eight different shape representations to make a dynamic physical shape with collective elements

2. **Active vs Passive:** The collective elements are categorized in active or passive, based on the actuation type
3. **Dynamic vs Inert:** The collective elements are also categorized in dynamic or inert, based on the state of the element, and the dynamic or inert elements are independent of whether the element is active or passive
4. **Definition of the Dynamicity:** The dynamicity of the constructed shape can be defined and measured by the number of elements that are currently dynamic.
5. **Shape Construction with Both Passive and Active Elements:** The dynamic shape construction can be achieved with both active (self-actuated) and passive (externally-actuated) elements

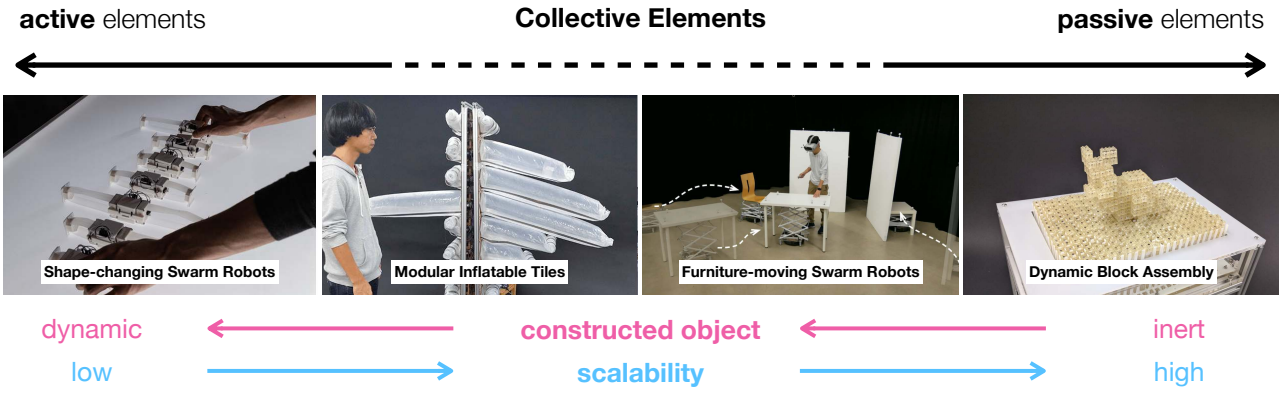


Figure 4.19: The dynamic shape construction can be achieved both active and passive collective elements. By drawing a continuum between 1) the shape constructed by active elements and 2) the shape constructed by passive elements, we can expand the design and application space.

Based on this, I discuss the following design implications that lead to each project.

1. **Exploring the new representation of collective shape construction:** There is still a rich space that the previous works have not explored well. This includes both representation of the shape as well as the methods of construction and transformation. The new representation of shape allow us to develop a different set of interactions.
2. **Combining the transformation methods of active elements**

to improve the expressiveness By expanding and combining each element's shape-changing capability (e.g.,movement, transformation, connection, etc), we can also develop more expressive representation by transforming between different representations.

3. **Constructing shape with parallel and simultaneous actuation enables scalable and high resolution outcomes:** By leveraging actuation systems like an array of electromagnetic coils or an array of actuated pins, we can create a system that can dynamically create a shape with collective passive elements that are actuated simultaneously.
4. **Combining both active and passive elements for collective actuation opens up a new design space:** A swarm of active elements provides an interesting and promising approach as they can be used as dynamic collective elements as well as make the passive collective elements dynamic by collectively and simultaneously actuating them.

Based on this exploration, this thesis contributes to the following three in the next chapters:

- **Part I - Active Collective Elements:** First, I will demonstrate how we can expand the design space through a collective shape construction of active elements.
- **Part II - Passive Collective Elements:** Second, I will show the passive, externally-actuated elements can also make the dynamic shape construction by leveraging parallel and collective actuation. I expand this design space and demonstrate it through new techniques and design architecture.
- **Part III - Interaction with Collective Elements:** Third, I will explore a method and interaction technique to program the collective elements that construct an interactive shape. I demonstrate these techniques as a new way of tangible programming through direct physical manipulation.

Given these investigations, I discuss the design implications of these methods and what we can learn for future investigations. This thesis concludes with the discussion of the potential research directions and opportunities for the future of dynamic and collective shape

Part I

Dynamic Shape Construction with Active Collective Elements

In this part, I will explore dynamic shape construction with **active collective elements**. Active collective elements refer to self-actuated elements that can construct a dynamic shape. By individually changing its physical property, such as position, orientation, length, volume, connectability, and materiality, or actuating external objects, active collective elements collectively and dynamically construct and transform the overall shape for human-computer interaction.

To exemplify how active collective elements can be used for interactive and dynamic physical displays, this chapter introduces two projects: the first one is ShapeBots: shape-changing swarm robots, and the second one is LiftTiles: large-scale pneumatically-actuated transformable tiles.

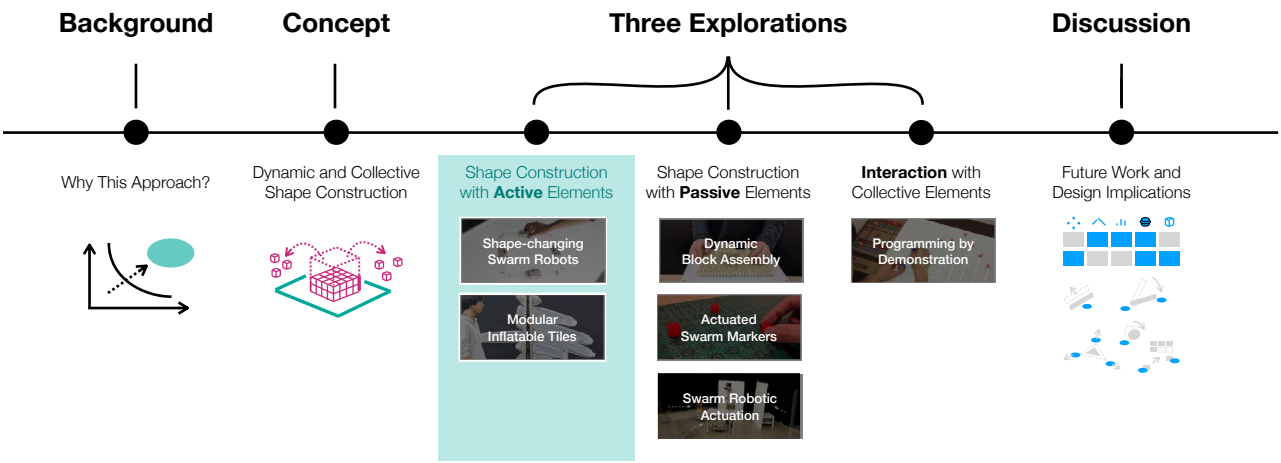


Figure 4.20: Part I: Dynamic shape construction with active elements.

5

Dynamic Shape made of Shape-changing Swarm Robots

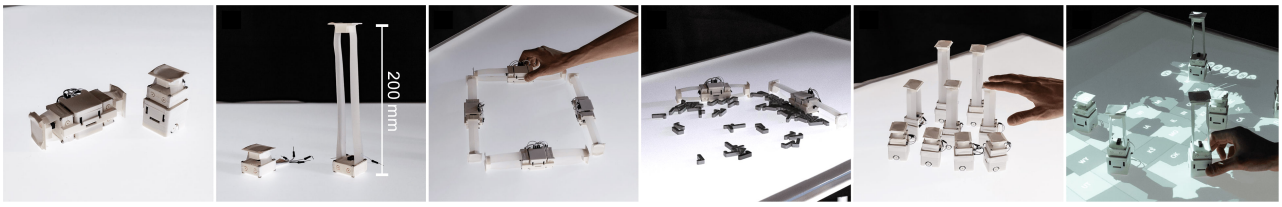


Figure 5.1: ShapeBots exemplifies dynamic shape construction with a swarm of self-transformable robots [Suzuki et al., 2019b].

5.1 Overview

Swarm robots provide one of promising approaches for collective dynamic physical displays. As each element can freely move, these robots can display a shape on 2D surface or in 3D space. For example, drone display is now well known example of how computationally controlled swarm robots can show a shape in the mid air.

Traditionally, a shape constructed by swarm robots is repre-

sented as a spatially distributed points (i.e., sparse dots representation). This project considers what if each swarm robot cannot only move in space, but also transform itself to expand the shape representation. For example, by changing their own shape of the tiny swarm robots into a line, surface, or vertically extendable pins, these swarm robots can also represent different shapes (e.g., sparse lines, single line, pin arrays, etc) which allow to expand the expression of the robots can show.

To explore this idea, this project introduces the concept of shape-changing swarm robots and demonstrate how this both individual and collective shape transformation capability can expand the expressiveness of the swarm displays as well as the range of applications.

The main contributions of this chapter [1] are:

1. A concept of shape-changing swarm robots.
2. ShapeBots implementation with a novel linear actuator that achieves a high extension ratio and small form factor.
3. A set of application scenarios that illustrate how ShapeBots can enhance the display, interactions, and affordances of the current swarm and shape-changing interfaces.
4. Design space exploration of shape-changing swarm user interfaces.

[1] Suzuki, R., Zheng, Clement Kakehi, Y., Yeh, T., Yi-Luen Do, E., Gross, M. D., and Leithinger, D. (2019b). Shapebots: Shape-changing swarm robots. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM

5.2 Shape-changing Swarm Robots

5.2.1 Definition

First, I introduce the concept of shape-changing swarm robots. Shape-changing swarm robots are defined as a type of system that consists of a swarm of self-transformable and collectively movable robots. This thesis specifically focuses on the user

interface aspect of such systems, which we refer to as shape-changing swarm user interfaces.

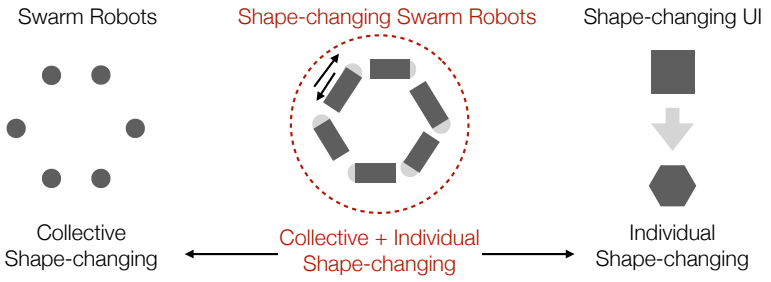


Figure 5.2: Swarm robots leverage collective shape transformation (left), and shape-changing interfaces leverage an individual shape transformation (right). Shape-changing swarm robots leverage both collective and individual shape transformation (center).

5.2.2 Goal

Shape-changing swarm robots are inspired by and built upon existing swarm user interfaces [2]. Swarm user interfaces support interaction through the collective behaviors of many movable robots. By combining such capability with individual shape change, we can enhance the expressiveness, interactions, and affordances of current swarm user interfaces.

[2] Le Goc, M., Kim, L. H., Parsaei, A., Fekete, J.-D., Dragicevic, P., and Follmer, S. (2016). Zooids: Building blocks for swarm user interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 97–109. ACM

For example, self-transformable swarm robots can support representations that are not limited to moving points, but also lines, and other shapes on a 2D surface (Figure 5.1). For example, each little robot could change its width or height to display a geometric shape or represent data embedded in the physical world. By collectively changing their heights, they can also render a dynamic shape-changing surface. In addition to rendering information, it can enhance interactions and affordances of current shape-changing interfaces. For example, these robots can collectively behave to actuate existing objects (e.g., clean up a desk, bringing tools when needed), become a physical constraint (e.g., a shape-changing ruler), provide physical affordances (e.g., create a vertical fence to indicate that a coffee cup is too hot to touch), and serve as a tangible controller (e.g., for a pinball game).

5.2.3 Three Aspects of Shape-changing Swarm Robots

We identified three core aspects of shape-changing swarm robots: 1) locomotion, 2) self-transformation, and 3) collective behaviors of many individual elements.

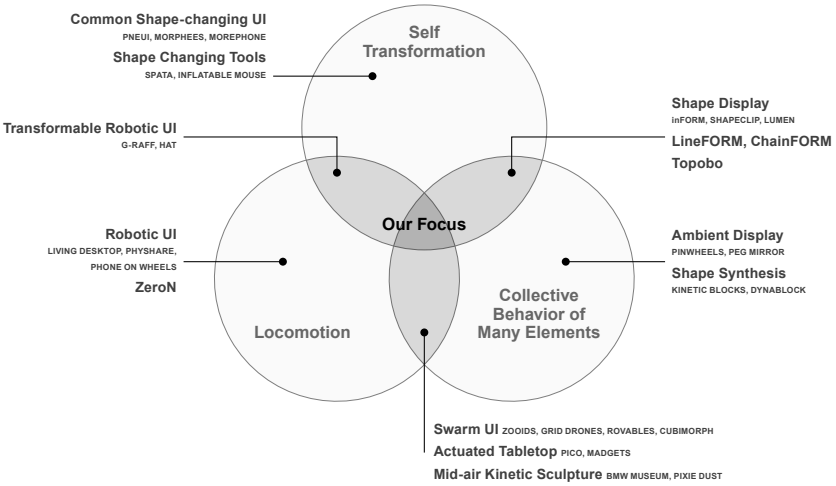


Figure 5.3: Scope and definition of shape-changing swarm user interfaces. The diagram classifies and highlights the difference between existing shape-changing interface systems and shape-changing swarm UIs.

- **Locomotion:** Locomotion is the key to make shape-changing interfaces more ubiquitous, which gives an interface more agency to be at hand on demand. For example, the device can come to the user and provide in-situ help when needed, then disappear when no longer needed. Locomotion also expands the space where the user can interact. For example, shape-changing interfaces that can move on a table, walls, and bodies can free the interaction space from a fixed specific location, as mobile and wearable devices have done.
- **Self-transformation:** Self-transformation refers to shape changes of a single element (e.g., form, texture, volume), as opposed to shape changes through spatial distributions. The shape of a physical object affords use and functionality. Thus, the capability of self- transformation plays an important role to provide rich physical affordances for tangible interactions.

- **Collective Behaviors:** A single shape-changing object is limited in its ability to rep- resent general shapes. Collective behaviors of many elements can overcome this limitation. Many actuated elements can act together to create expressive and general-purpose shape transformation that a single actuated element cannot achieve. Shape displays leverage this aspect to present arbitrary 2.5D shapes on a tabletop surface.

Given these core aspects, we categorized current shape-changing user interface research. Our focus on shape-changing swarm user interfaces lies in the intersection of these three aspects. For example, shape displays leverage both the self-transformation of each actuator and the collective behaviors of many elements. In contrast, swarm user interface and actuated tabletop leverage locomotion and many collective elements. Transformable robotic interfaces leverage self-transformation and locomotion. Shape-changing swarm user interfaces exhibit self-transformation and locomotion, and leverage the collective behavior of many individual elements.

5.3 ShapeBots

To exemplify this concept, we developed ShapeBots, self-transformable swarm robots with modular linear actuators (Figure 5.1). ShapeBots have four key technical components: 1) a miniature reel-based linear actuator, 2) self-transformable swarm robots, 3) a tracking mechanism, and 4) a control system.

5.3.1 Mechanical Design

Miniature Reel-based Linear Actuators

One technical challenge to realize self-transformable swarm robots is the design of a miniature actuator that fits into a small robot and has a large deformation capability. Typical linear actuators, such as a lead screw or a rack and pinion, have only small displacement; they can extend between two to four times in length. One of our main technical contributions is the design of a miniature linear actuator that extends from 2.5 cm to 20 cm (Figure 5.1).

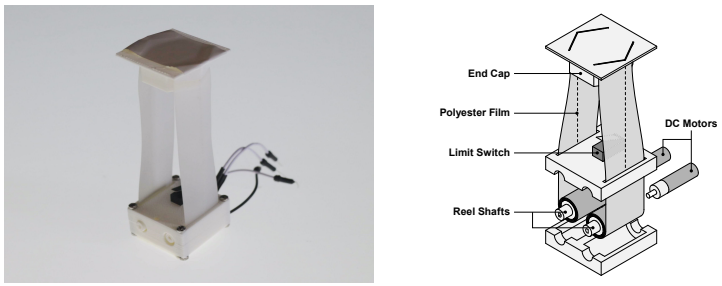


Figure 5.4: Mechanical design of the ShapeBot's linear actuation unit.

Figure 5.4 illustrates the design of our linear actuator. It is inspired by a retractable tape measure, which occupies a small footprint, but extends and holds its shape while resisting loads along certain axes. Our reel-based linear actuator employs small DC motors (TTMotor TGPP06D-700, torque: 900g/cm, diameter: 6 mm, length: 22 mm). The linear actuator comprises two reels of thin sheets (e.g. 0.1mm thick polyester sheet).

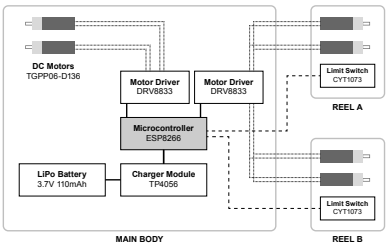


Figure 5.5: Schematics of ShapeBot's electronic components. The main ESP8266 microcontroller operates at 3.3V. Two dual motor drivers drive DC motors for the robot and linear actuators respectively.

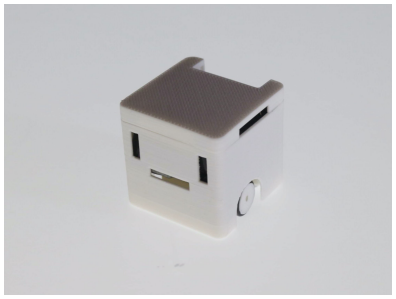
Two DC motors (TTMotor TGPP06D-700) rotate in opposite directions to feed and retract the sheets. Each sheet is creased at the center along its length, and is connected to a cover cap that maintains the fold. This crease increases the structural stability of the linear actuator, similar to how a tape measure remains stable when extended. Thus the linear actuator can push and drag lightweight objects without bending and extend vertically without buckling.

The linear actuator extends and retracts with open-loop control; we estimate extension based on the duration of operat-

ing the motors. When fully retracted, the cap triggers a limit switch (Figure 5.4), then the length of the actuator will be initialized to zero. The average difference between the target and actual length is less than 5 mm (see the technical evaluation section). The polyester sheets are attached to 3D printed shafts with super glue. The 3D printed enclosure measures 3cm x 3cm and is 1.5 cm thick. There is 1 cm offset between the cap and the bottom end for the limit switch. Thus, the initial thickness of the linear actuator is 2.5 cm. Although the length of the reel can exceed 20 cm to achieve a higher extension ratio, such a longer reel would more likely buckle. During prototyping, we observed that friction between sheets and enclosures can cause a jam while extending. Thus, reducing friction is the key to reliable actuation. To reduce friction, we attached a smooth material sheet (e.g., the polyester sheet or peeling sheet of double-sided tape) to the inside of the enclosure.

Swarm Robot

Figure 5.6 illustrates the design of the swarm robot. Each robot is driven by two micro DC motors (TTMotor TGPP06D-136, torque: 550 g/cm, diameter: 6 mm, length: 18 mm). By individually controlling rotation speed and direction, the robot moves forward and backward and turns left and right. Two 3D printed wheels (1 cm diameter) connect directly to the DC motors. An O-ring tire on each wheel increases friction with the ground to avoid slipping.



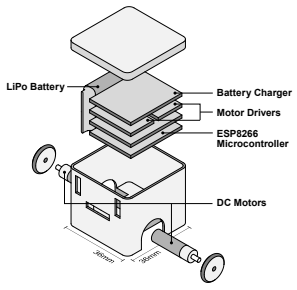


Figure 5.6: Mechanical design of the ShapeBot's swarm robot unit.

Two DC motors soldered to the dual motor driver (DRV8833) are controlled by the main microcontroller (ESP8266). A LiPo battery (3.7V 110mAh) powers both the microcontroller and the motors. Each robot also has an additional DRV8833 motor driver to control the linear actuators; the two motor drivers connect to the microcontroller through a 2-sided custom PCB. All components are enclosed with a 3D printed housing (3.6 cm x 3.6 cm x 3 cm) with three rectangular holes in the front side (Figure 5.6) that house micro USB ports for programming and recharging and the microcontroller reset switch. All 3D printed parts are fabricated with an FDM 3D printer (Cetus 3D MKII) and PLA filament (Polymaker PolyLite 1.75mm True White).

5.3.2 Types of Transformation

Due to the modular and reconfigurable design of the linear actuator unit, ShapeBots can achieve several different types of transformations. The figure demonstrates five types of shape transformations: horizontal, vertical, and curved lines, volumetric change with an expandable Hoberman sphere, and 2D area coverage with an expandable origami structure.

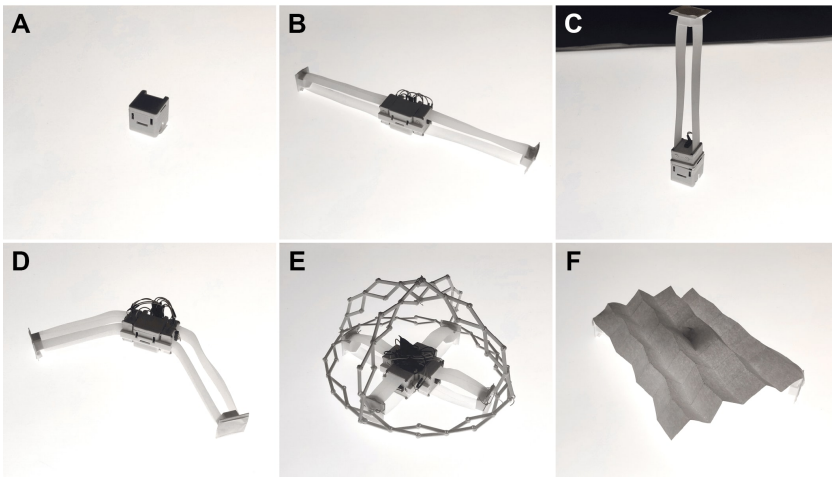


Figure 5.7: Different types of transformation enabled by modular linear actuator units. A) the basic ShapeBot, B) horizontal extension, C) vertical extension, D) bending, E) volume expansion, and F) area expansion.

These configurations support three types of shape change (e.g.,

form, volume, orientation) categorized in Rasmussen et al. [3]. For horizontal extension, each linear actuator unit is fixed with a custom 3D printed holders. For the vertical extension, we used a thick double-sided tape on top of the swarm robot.

[3] Rasmussen, M. K., Pedersen, E. W., Petersen, M. G., and Hornbæk, K. (2012). Shape-changing interfaces: a review of the design space and open research questions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 735–744. ACM

5.3.3 Tracking and Control Mechanism

To track the position and orientation of the swarm robots, we used computer vision and a fiducial marker attached to the bottom of the robot. Precise orientation tracking is critical. For example, to make a triangle with horizontal extended lines, three swarm robots must orient to appropriate directions. The fiducial marker enables easy, precise, and reliable tracking of both position and orientation that is undisturbed when users occlude the sides and top of the robot during the interaction.

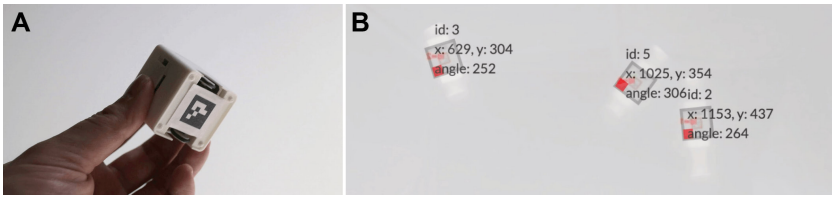


Figure 5.8: A) Fiducial marker (Aruco 4 x 4 pattern, 1.5cm x 1.5cm) is attached to the bottom of each robot. B) OpenCV tracks positions and orientations of markers at 60 FPS.

We used the ArUco fiducial marker [Garrido-Jurado et al., 2014] printed on a sheet of paper and taped to the bottom of the robot. Our prototype used a 1.5 cm x 1.5 cm size marker with a 4 x 4 grid pattern, which can provide up to 50 unique patterns. For tracking software, we used the OpenCV library and ArUco python module. It can track the position of the markers at 60 frames per second. The captured image and position information is streamed to a web user interface through a web socket protocol.

We constructed a transparent table with a 6mm acrylic plate mounted on a custom frame. The web camera (Logitech C930e) beneath the table captures a 115 cm x 74 cm effective area. The camera is connected to the main computer that uses the track-

ing information to control the robots. We cover the transparent plate with a polyester sheet (Mylar drafting film) to diffuse light and reduce hot spots, which cause unreliable tracking. Three lamps illuminate the bottom of the table to increase the contrast of the markers for better readability. For some applications, we also mounted a projector (Epson VS250) 100 cm above the table to present graphical information. As the system tracks the robots from below, the projected image does not affect tracking. Figure 5.9 illustrates our current setup.

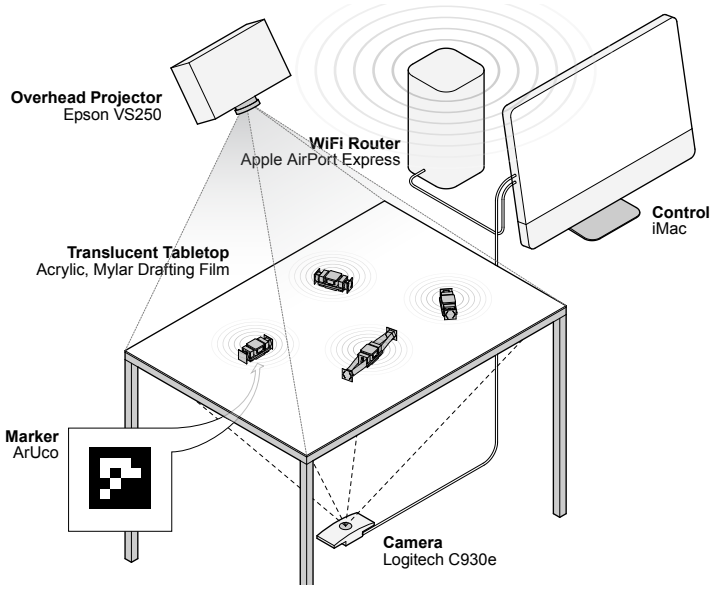


Figure 5.9: Tracking setup. A webcam (Logitech C930e) mounted 90 cm beneath the table and connected to the main computer (iMac) captures 115 cm x 74 cm effective area.

To enable the user to easily specify a target shape, we created a web-based interface where users draw a shape or upload an SVG image. The user draws a set of lines, then the main computer calculates target positions, orientations, and actuator lengths to start sending commands. The user can create a keyframe animation by drawing a sequence of frames.

5.3.4 Interaction Capability

We can use the same mechanism to track user input. The figure illustrates four different types of user interaction that our system supports: place, move, orient, and pick-up.

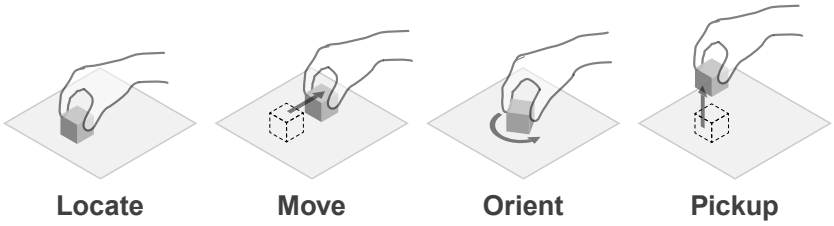


Figure 5.10: Interaction capability of ShapeBots.

The system recognizes as user inputs movement or rotation of a marker that it did not generate. When the system detects a new marker or loses an existing marker, it recognizes that the user is placing or picking up a robot. Robots that the systems are driving are not candidates for user input. Thus, when unintended events occur (e.g., bumping or colliding), the system distinguishes these from user input, as it is driving the robots. A single user can manipulate multiple robots with two hands, or multiple users can interact with the robots. In addition, by leveraging OpenCV's object detection algorithm, the system can detect the position and shape of objects on the table, such as pens, a sheet of paper, coffee cups, and phones.

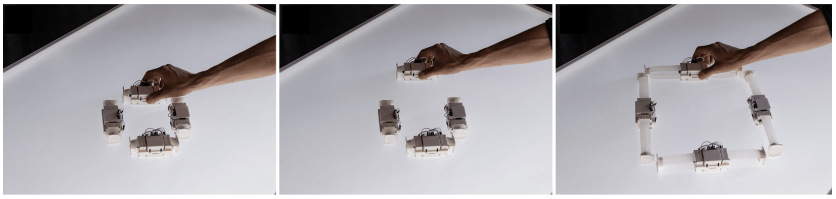


Figure 5.11: Interactive shape change. A small rectangle shape. If the user moves one element, then the robots change positions and lengths to scale the square.

5.4 Application Scenarios

5.4.1 Interactive Display

ShapeBots can also act as an interactive physical display. The figure shows how ShapeBots can render different shapes.

It allows users to preview a shape. For instance, when reading a picture book of animals, children can visualize the fish with

Figure 5.12: Application scenarios in interactive display. Sparse line elements display various shapes, such as hexagon, fish, and text.

ShapeBots at actual size. The figure demonstrates the input and output capabilities as an interactive tangible display. Four robots first move to display a small rectangle. When the user moves a robot, the others change positions and lengths to scale the shape. The user can also move robots to rotate or translate the shape.

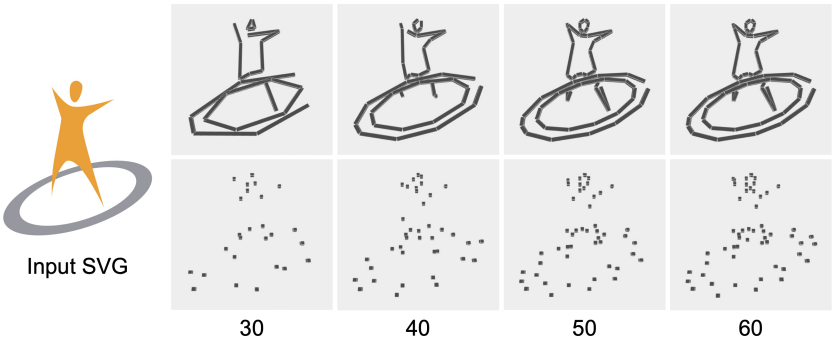


Figure 5.13: Simulation results comparing ShapeBots (top) with swarm robots (bottom). Left to right: original SVG image, rendering simula- tion results with 30, 40, 50, and 60 robots respectively.

The figure highlights the advantage of ShapeBots for rendering contours compared to non self-transformable swarm robots. Using a software simulation, we demonstrate how ShapeBots renders an SVG input at different swarm sizes.

Similarly, ShapeBots can provide a physical preview of a CAD design. The figure shows a user designing a box. ShapeBots physicalizes the actual size of the box. The design and physical rendering are tightly coupled; as the user changes the height of the box in CAD software, the ShapeBots change heights accordingly. The user can change the parameters of the design by moving robots in the physical space, and these changes are reflected in the CAD design.

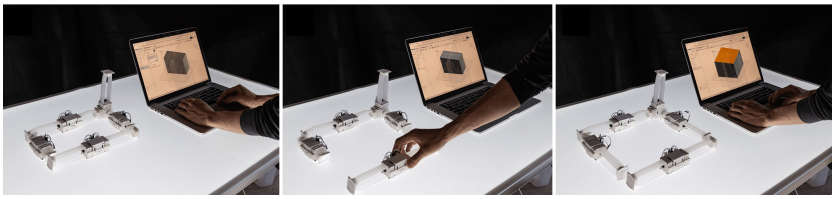


Figure 5.14: Physical preview for the CAD design. ShapeBots provide a physical preview synchronized with the computer screen. When the user manipulates the element, then it updates the digital design of the CAD software.

5.4.2 Interactive Data Physicalization

Interactive display enables many applications. One interesting application area is interactive data physicalization. For example, in Figure 5.15, seven robots transform individually to represent a sine wave. These representations are interactive with user inputs: when the user moves the end robot to the right, the others move to change the wavelength. The user can interactively change the amplitude of the wave by specifying the maximum length.

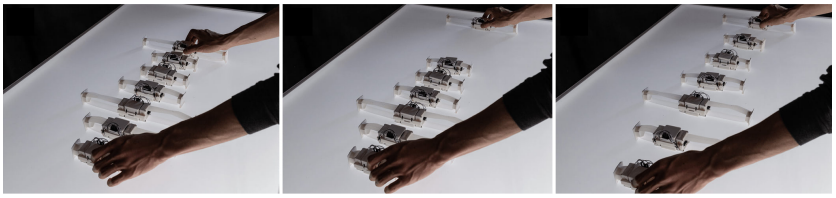


Figure 5.15: An interactive and animated sine wave. Animated sine wave. When the user moves one element, then each robot can collectively move to change the spatial period of the wave.

ShapeBots also support transforming data into different representations such as bar graphs, line charts, and star graphs. The user can place and move robots, which enables embedded data representations [4]. For example, ShapeBots on the USA map physicalize map data; each robot changes its height to show the population of the state it is on. Users can interact with the dataset by placing a new robot or moving a robot to a different state, and the robots update their physical forms to represent the respective population.

Other examples of distributed representations include showing the magnitude and orientation of wind on a weather map, or physicalizing magnetic force fields. This physical data repre-

[4] Willett, W., Jansen, Y., and Dragicevic, P. (2016). Embedded data representations. *IEEE transactions on visualization and computer graphics*, 23(1):461–470

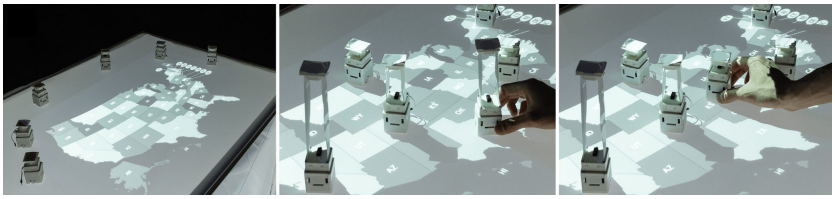


Figure 5.16: Embedded data physicalization on a map. Projected US map. When the user selects the dataset, the ShapeBots move to position and visualize data with their heights. When moved, the robots change their heights accordinly.

sentation could be particularly useful for people with visual impairments [Guinness et al., 2018; Suzuki et al., 2017].

5.4.3 Distributed Physical Affordances

By leveraging the capability of locomotion and height change of each robot, ShapeBots can create a dynamic fence to hide or encompass existing objects for affordances. For example, the Figure illustrates this scenario.

Figure 5.17: Distributed dynamic physical affordances. A user pours hot coffee, then ShapeBots create a vertical fence to prevent the user from grabbing the cup. Once the coffee has cooled, the ShapeBots disperse.

When the user pours hot coffee into a cup, the robots surround the cup and change their heights to create a vertical fence. The vertical fence visually and physically provides the affordance to indicate that the coffee is too hot and not ready to drink. Once it is ready, the robots start dispersing and allow the user to grab it. These scenarios illustrate how the distributed shape-changing robots can provide a new type of affordance, which we call distributed dynamic physical affordances.

By leveraging the ability to actuate objects and act as physical constraints. As an example, Figure shows two robots extending their linear actuators to wipe debris off a table, clearing a workspace for the user.

Figure 5.18: Clean up robots. A desk is filled with debris. Two robots starts moving and wiping the debris. Once the robots finish cleaning up, the user can start using the workspace.

5.4.4 Adaptable Tools and In-situ Assistants

In another scenario, the ShapeBots brings tools to the user. For instance, when the user needs a pen, a robot extends its actuators and pushes the pen to the user. These robots can also be used as tools. In the same example, when the user needs to draw a line with a certain length, the user specifies the length, then the robot extends its length to serve as a ruler. The user can also bend the linear actuator or using multiple robots to draw a curved line or other shapes.

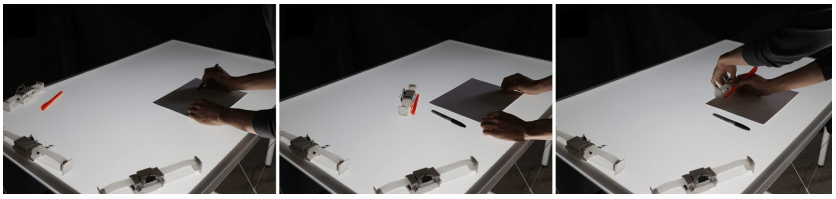


Figure 5.19: Shape-changing Tools. A user is working on a desk. When the user needs a pen, ShapeBots can bring it. ShapeBots can be also used as a tool like ruler.

Finally, ShapeBots can be employed as a tangible gaming platform. The figure illustrates two users playing a table football game using two extended robots. The user controls a robot acting as the pinball arms whose position and angle are synchronized to a controller robot. Users hit or block a ball to target a goal, similar to table football.

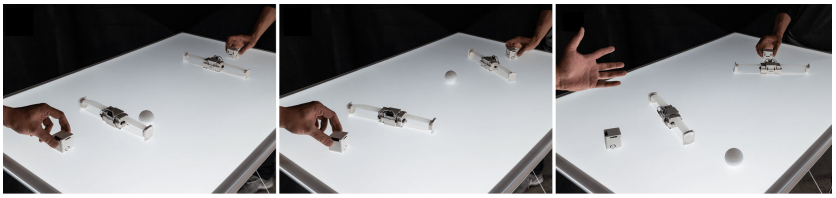


Figure 5.20: Tangible game controllers. Two user start playing a table football game. Each user uses one ShapeBot as a controller and another ShapeBot to hit the ball. The user can play with these Shape- Bots like table football.

5.5 Discussion and Design Space

We think there is the broader design space of shape-changing swarm user interfaces. To point out how future work can address open research areas, we will discuss some of the key design factors and considerations. The table identifies dimensions of the design space of shape-changing swarm user interfaces (Figure 5.22). The highlighted regions represent where ShapeBots fit within the design space.

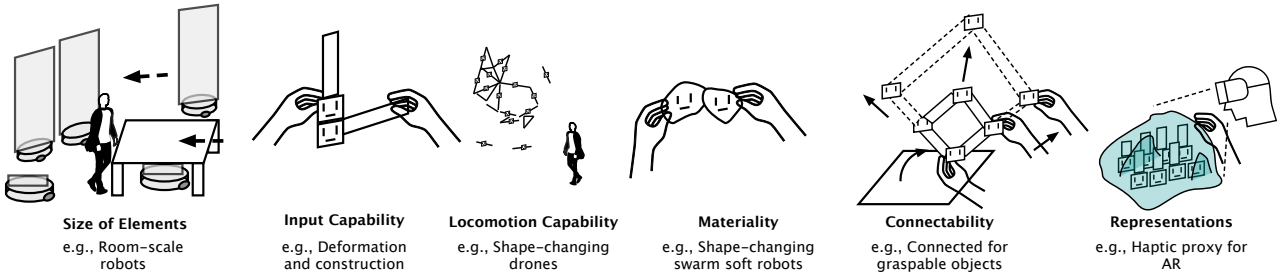


Figure 5.21: Design space of shape-changing swarm user interfaces illustrating future research opportunities.

5.5.1 Size of Elements

The size of the swarm elements is another dimension. This paper focuses on small (3 - 4 cm) robots, but other sizes would open up new application domains. For example, room-size shape-changing swarm robots could produce dynamic furniture (e.g., transforming chairs and tables) or modify the spatial layout of a room through movable and expandable walls. Today, most large robots are seen in factories and warehouses, but as they enter everyday environments (e.g., cleaning or food delivery robots), we see opportunities for shape-changing robots to enable people to interact with and reconfigure their environments.

5.5.2 Input

Our current prototype supports four types of user inputs: place, move, orient, and pick up. Supporting other types of

inputs can enhance richer interaction possibilities (Figure 24B). For example, if the user can also manually extend lines or deform the shape of robots, new types of applications could be possible, such as using these robots as a physical tape measure like SPATA [Weichel et al., 2015].

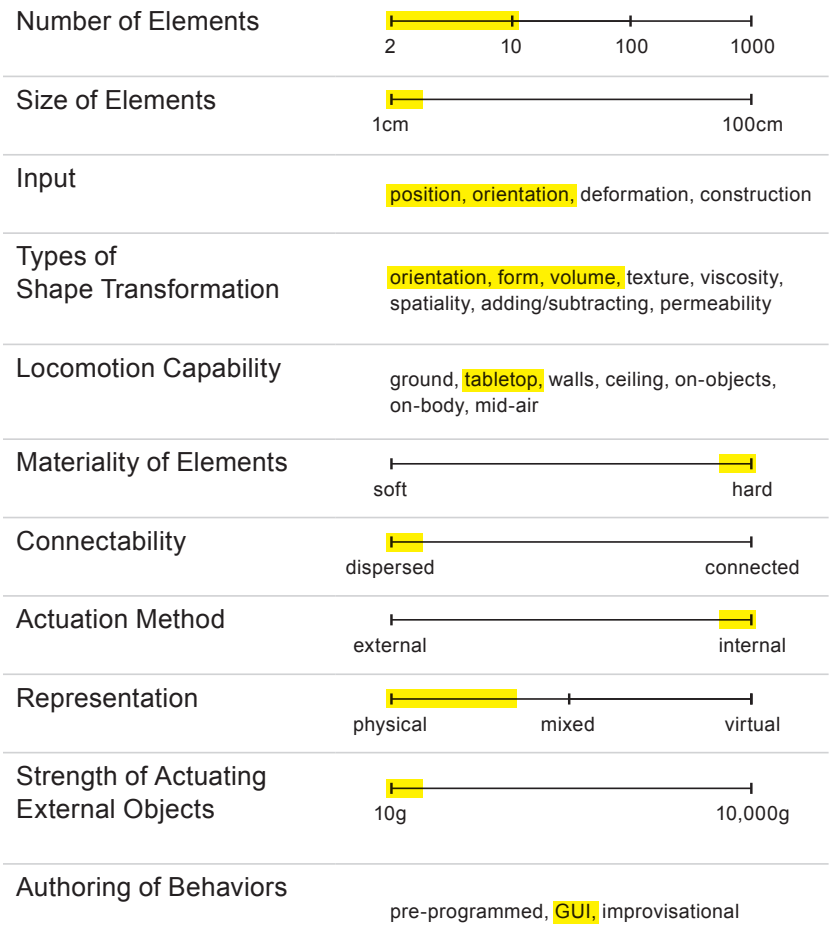


Figure 5.22: Design space of shape-changing swarm user interfaces.

5.5.3 Locomotion Capability

Camera-based tracking limits the locomotion capability of Shapebots to a tabletop surface. However, future shape-changing swarm robots could also cover walls, ceilings, objects, the user's body and hover in mid-air. Swarm robots on walls, windows, and building facades could serve as an expressive tan-

gible public information display. A swarm of ceiling crawling self-transforming robots could pick up objects and move them from one place to another through extendable arms. Using a sucking mechanism demonstrated in Skinbots [Dementyev et al., 2017], robots could move on-body or on-object. Mid-air drone swarms, with added shape-changing capabilities, could form more expressive mid-air displays like vector graphics or a 3D mesh structure

5.5.4 Materiality of Elements

Most swarm robots, including our current prototype, are made of rigid materials. However, soft swarm robots made of malleable materials could exhibit other expressive self- transformation capabilities such as changes in volume and stiffness. From the interaction perspective, it would be interesting if users could construct an object using these soft robots like pieces of clay, and the constructed soft object can transform itself into another shape.

5.5.5 Connectability

In the ShapeBots prototype, the swarm elements do not physically connect, but the ability of swarm robots to connect would enable graspable 3D shapes. For example, the connection between lines enables the user to pick up a rendered object while the object can dynamically change its shape or scale in the user's hand. With a sufficient number of lines, such objects can represent arbitrary shapes, similar to LineFORM [Nakagaki et al., 2015] and ChainFORM [Nakagaki et al., 2016].

5.5.6 Representation

The physical content representation of shape-changing swarm robots can be combined with other modalities. As we demonstrated, with projection mapping, graphics can present infor-

mation that is difficult to convey solely through physical representations. Alternatively, graphics can be shown through internal LED lights (e.g., represent groups in data plots through color). An interesting research direction would leverage the physicality of robots to provide haptic feedback for virtual or augmented reality. For example, we demonstrated a scenario where the robots show a physical preview of CAD design. With mixed reality the user can overlay information on top of the robots like Sublimate [Leithinger et al., 2013] or MixFab [Weichel et al., 2014].

6

Dynamic Shape made of Modular Inflatable Tiles

Figure 6.1: LiftTiles exemplifies the dynamic shape enabled by collective transformation of the modular inflatable tiles [Suzuki et al., 2020b, 2019a].

6.1 Overview

In this chapter, I will explore how the active collective elements can be scaled up for larger shape construction, including body-scale, room-scale, or building-scale. This chapter specifically focuses on the shape representation of pin arrays (see Chapter 4) and how collective discrete elements can dynamically construct various shapes by extending its length.

Traditionally, existing modular shape displays research [Everitt et al., 2016; Hardy et al., 2015] often focuses on interactions at the scale of the human hand. However, larger-scale shape construction and transformation promises lots of exciting applications, as discussed in [Green, 2016; Sturdee et al., 2015].

One of the challenges to scale up the modular shape displays is its difficulty of implementation. To construct and deploy such large-scale shape displays requires substantial and therefore prohibitive cost, time, and effort. These barriers restrict the research community to prototype and explore applications beyond the scale of the human hand.

In this chapter, I introduce LiftTilesr, modular inflatable actuators as building blocks for prototyping *room-scale* shape-changing displays. By leveraging discrete, compact, and highly extendable actuators, the user can construct a large-scale shape display. Moreover, our modular and reconfigurable design allows researchers and designers to quickly construct different geometries and to explore various applications. This chapter describes the implementation of LiftTiles and possible application scenarios.

The main contributions of this chapter [1] [2] are:

1. Room-scale modular shape displays with reconfigurable inflatable actuator.
2. Design of a highly extendable linear actuator that is low-cost, lightweight, compact, robust, and easy-to-fabricate.
3. A range of applications that demonstrate the potential of the system for room-scale shape transformation.

[1] Suzuki, R., Nakayama, R., Liu, D., Kakehi, Y., Gross, M. D., and Leithinger, D. (2020b). Lifttiles: Constructive building blocks for prototyping room-scale shape-changing interfaces. In *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction*. ACM

[2] Suzuki, R., Nakayama, R., Liu, D., Kakehi, Y., Gross, M. D., and Leithinger, D. (2019a). Lifttiles: Modular and reconfigurable room-scale shape displays through retractable inflatable actuators. In *Adjunct Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM

6.2 Large-scale Modular Shape Displays

6.2.1 Design and Technical Considerations

Actuation techniques that have been utilized for prior shape displays include servos, nitinol, hydaulic and pneumatic actuators. To achieve the room-scale shape change, we define several design goals and resulting technical requirements for a room-scale shape display.

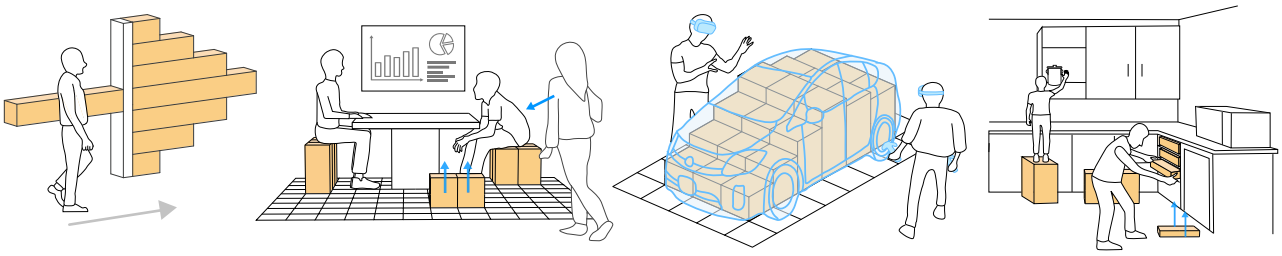


Figure 6.2: Concept sketch of application scenarios with room-scale modular shape displays.

Scalablilty

The scalability is an important requirement which is related to size, cost, fabrication complexity, and the number of elements. Commercially available mechanical actuators tend to be expensive when the system requires larger size, stronger force, and higher load resistance. Thus, we considered inflatable actuation for low-cost and scalable actuation methods.

Robustness

Strength and robustness of the actuator is another design requirement which is related to the factors of force, load tolerance, safety, and speed. The strong force might be one of the unique requirement for the room-scale shape displays. Unlike the table-top scale, the room-scale shape displays needs to bear the heavy and large objects. We explored a number of designs and prototypes, and found that a larger size can provide higher stability. Thus, we designed our actuators, with the goal

of achieving a large footprint and high load tolerance, while maintaining the ease fabrication for scalability.

Deployability

Finally, deployability is the last design requirement which is related to installation complexity, thickness, extension ratio, weight, and power consumption. One common design strategy is modular and reconfigurable design. A simple and compact unit allows the user to easily install and customize, based on their needs and situation. However, if each element can be too large and heavy, it becomes an obstacle for storage and portability. Through our prototype process, we aim to minimize the thickness and weight, so that the user can easily pick and place, just like placing a tile on the floor.

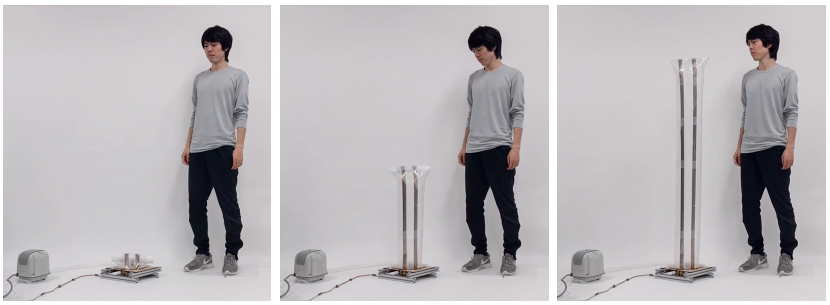


Figure 6.3: The actuator can extend from 15 cm to 150cm when inflated and re- tract when deflated

6.3 LiftTiles

In this section, I introduce LiftTiles, a modular and reconfigurable room-scale shape display that leverages pneumatic actuation. I will describe the following three aspects:

1. **Retractable and inflatable actuators:** a compact yet highly extendable actuator using inflatable and retractable constant force springs.

2. **Modular and reconfigurable design:** modular design allows actuators to be rearranged into different topologies and orientations.
3. **Design space exploration:** the exploration of several design factors of LiftTiles, such as size, shape, position, orientation, and color, which informs the configuration and applications.

6.3.1 Mechanical Design

Our inflatable actuator is a new type of extendable linear actuator that uses an inflatable structure similar to a party horn. Each inflatable actuator consists of a flexible plastic tube and two constant force springs, which are rolled at their resting positions. When pumping air into the tube, the actuator extends as the internal air pressure increases and the end of the tube unwinds. When releasing air through a release valve, the inflatable tube retracts due to the force of the embedded spring returning to its resting position.

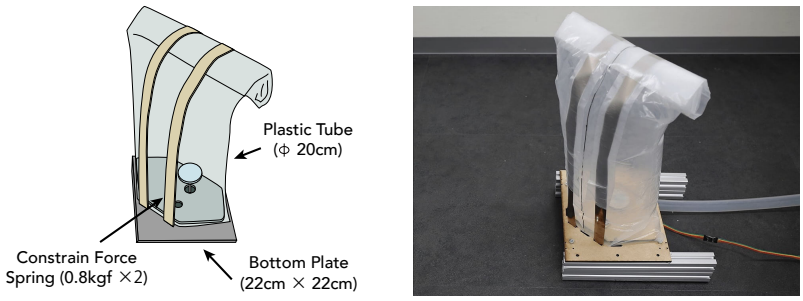


Figure 6.4: Each actuator consists of a plastic tube and constant force springs.

Our actuator design is inspired by the prior reel-based pneumatic actuator proposed by Hammond et al. [3]. During our prototyping process, we first implemented and tested that design to determine if we could scale it to large-scale actuation. However, the force of the spiral torsion springs used in [Hammond et al., 2017] was too weak to retract a large sized pneumatic tube (e.g., 20cm in diameter). In addition, the retractable force of the spiral springs is not constant and changes accord-

[3] Hammond, Z. M., Usevitch, N. S., Hawkes, E. W., and Follmer, S. (2017). Pneumatic reel actuator: Design, modeling, and implementation. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 626–633. IEEE

ing to the length of the inflatable tube (e.g., weak spring force at shorter extension), Therefore, we explored a modified designs and fabrication for larger-scaled pneumatic actuators. We found that instead of a spiral torsion spring at the end of the actuator, using a constant force springs throughout its length allows more stability and stronger retraction for larger-scale actuation, as the constant force spring can provide the constant retraction, independent with the current height. Thus, we decided to further explore the design and fabrication technique based on the constant force spring.

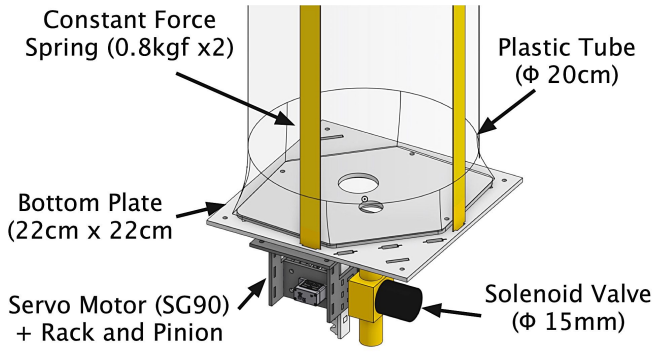


Figure 6.5: Each actuator consists of a plastic tube and constant force springs.

Figure 6.5 illustrates an overview of our actuator design. The basic components of each actuator are 1) an inflatable tube, 2) a base plate, and 3) two valves to supply and release air.

1) Inflatable Tube

The inflatable tube is made of a plastic tube and two constant force springs. In our prototype, we used a polyethylene vinyl tube (0.15mm thickness), which is 27 cm wide when deflated, and is 20 cm in diameter when inflated. We first cut each plastic tube at 1.5m, insert the internal bottom plate, and then heat seal the both ends for air-proofing. Two constant force springs (Dongguan Yongsheng Metal) are attached to the side of the inflatable tube. Each constant force spring is 1.5 m in length, 2 cm in width, and a 0.8 kgf load, which leads the 1.6 kgf retractable force in total for each tube. One end (flat side) of the

spring is fixed to the bottom plate and the other end (rolled side) is fixed to the top end of the tube by extending it. Then, we stick the spring to the plastic tube with a tape. The tube can be rolled with the force of the spring.

2) Base Plate

The inflatable tube is attached to the base plate that was fabricated from a laser cutter. To correctly position the tube we use two plates, one of which is inserted in the inside of the inflatable tube, and the other is outside of the tube, which can sandwich the bottom of the tube to appropriately fix the position as well as to prevent air leakage. The two plates and the tube are fixed to the bottom plate with four screws, and each screw has an O-ring in both the inside and outside of the tube to prevent any air leakage with pressure. To supply and release air, there are two holes on both plates. The hole are 20 mm and 45 mm in diameter for the air supply and release respectively, which are connected to the two valves described below.

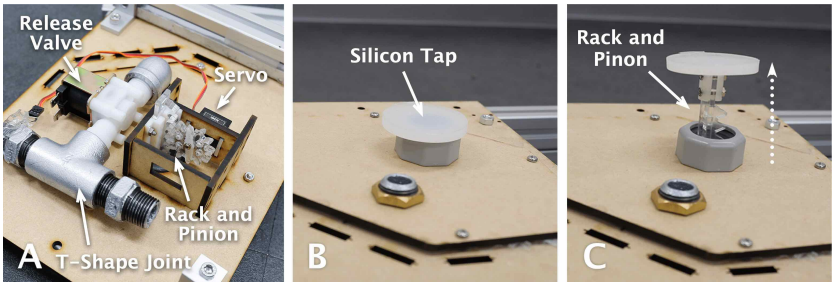


Figure 6.6: The solenoid valve and the servo motor attached on the base plate. The servo motor, rack, and pinion mechanism actuate and move up the silicon tap to open and close the release valve.

3) Air Supply and Release Valves

The air supply hole is directly connected to a solenoid valve (Ebowan Plastic Solenoid Valve DC 12V), which is 20mm in diameter for the outer hole and 15 mm for the internal hole respectively. When running current through the valve, it can open up and supply air through the supply hole. For the release valve, we fabricated a custom silicon tap. The silicon

tap is 50 mm in diameter at the top and is a T-shape from the side, which can withstand the high pressure from the top (Figure 6.6). The silicon tap can open and close the release valve through vertical actuation. A rack and pinion gear mounted on a servo motor (TowerPro SG90) actuates this silicon tap; the servo raises this silicon tap to release air (Figure 6.6C) and pull down the tap to close the valve (Figure 6.6B). The servo motor, rack, and pinion are fixed to the base plate with the laser-cut outer structure (Figure 6.6A). Two aluminum frames are attached to the bottom plate, so that it can provide a space between the plate and the ground to store the solenoid valves, rack, and pinion mechanism.

The overall size of each actuator has a 22 cm x 22 cm footprint and is 15 cm in its height (7 cm for a rolled tube and 8 cm for the bottom plate and the valves), and 1.8kg in its weight. When we mount a telescopic enclosure (described below), it has total a 30 cm x 30 cm footprint and is 18 cm in its height for each actuator. In our prototype, the total cost of each actuator is less than 8 USD, including the solenoid valve (3 USD), contact force springs (0.5 USD), the servo motor (2 USD), and other material cost such as vinyl tubes. We fabricated 25 inflatable actuators in total for our prototype.

6.3.2 Actuation Mechanism

Figure 6.7 illustrates the overall mechanism of pneumatic control for the height of each actuator. At the initial state, each tube is stored in a roll and both valves are closed.

By opening the supply valve and pumping air into the pneumatic tube, the supplied air can gradually extend the tube from the bottom while the top of the tube is still rolled. When closing the supply valve and stopping the air from pumping, the internal air can maintain the current height. The internal

air pressure can withstand the external force from the vertical direction.

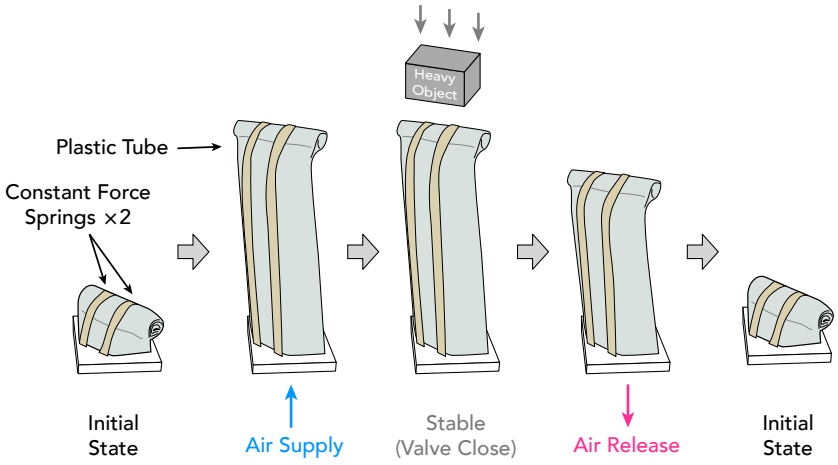


Figure 6.7: Mechanism of our inflatable actuator and pneumatic control system for actuator arrays.

When we open the release valve by raising the silicon tap with the servo motor, it releases the air and retracts the tube with the retractable force of the attached spring. Thus, by opening and closing the supply and release valves, the system can control the height of each actuator.

6.3.3 Fabrication Process

Here, we describe the fabrication process of our retractable and inflatable actuator. First, we cut the vinyl tube to an appropriate length (1.5m) and then place it on the bottom plate. Next, we install the solenoid valve and silicon tap with the custom designed rack and pinion gear as well as the servo motor to the bottom plate. Then, we seal the tube and pump air with the air compressor or air pump to inflate it. Next, we fix one side of the contact force springs at the bottom and then peel and extend them until reaching the top of the tube. Then, we attach the springs to the side of the tube with tape. Finally, when releasing air, the constant force springs can automatically roll the tube. In our setting, the assembly time approximately takes 15-30 minutes.

Telescopic Enclosure

For the vertical actuation, the retractable and inflatable actuator has a non-flat shape at the top, which makes an unstable structure when a user steps, stands, or sits on it. Therefor, we designed and fabricated an extendable enclosure based on a telescopic structure. Each telescopic enclosure consists of ten different-sized square tubes made of corrugated plastic sheets (Figure 6.8).

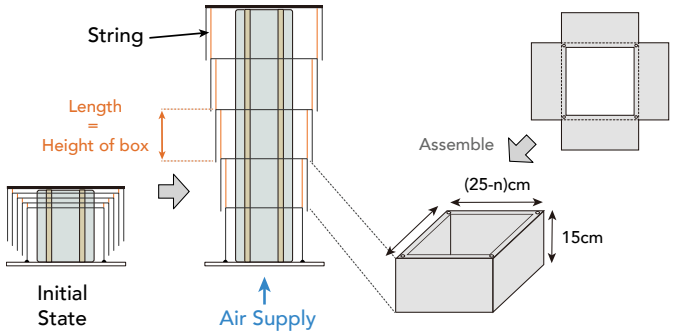


Figure 6.8: Telescopic structure for a collapsible enclosure.

We laser cut the plastic sheets (3mm thickness) and fold them into the tube, then stack these tubes and connect them to each other with mobilon rubber bands (Nishinbo MB8064TA100G, 6mm thickness, 80mm folding diameter) to ensure they are not separated when extended. Then, we fix the telescopic enclosure to the base plate with cable ties. This design allows the telescopic enclosure to extend along with the internal inflatable tube. We tested the load bearing of this telescopic enclosure and the internal inflatable actuator by placing weights in 2kg increments on top of them. The actuator could withstand 10kg, after which the telescopic enclosure would start to bend sideways, which made it difficult to appropriately measure the maximum load bearing. However, during use we did observe that multiple actuators stacked next to each other would resist bending and withstand higher loads.

6.3.4 Modular and Reconfigurable Design

Each individual actuator is modular and can be connected with the neighboring actuators. At the end of the solenoid air intake valve of each actuator, it is connected to a T-shaped plumbing joint. Adjacent actuators are pneumatically connected through a silicon tube between the T-shape joints. This way, an array of actuators is connected to a shared pressurized line.

Figure 6.9: Modular design and different actuator arrangements. Each module is connected with a flexible pneumatic tube.

An air compressor (Yasunaga Air Pump, AP-30P) pressurizes the shared line. This air pump provides up to 12.0kPa of pressure and supplies 30L of air volume per minute. Due to the shared line, when the number of simultaneously moving actuators increases, the pressure supplied to each actuator decreases. In this case, using multiple air compressors can alleviate the issue. In our setup, we used a single air pump for each set of ten actuators.

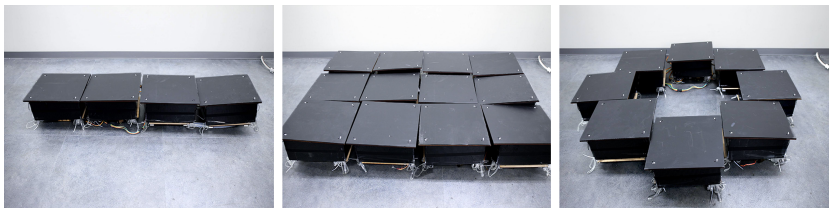


Figure 6.10: Modular design and different actuator arrangements. Each module is connected with a flexible pneumatic tube.

6.3.5 Pneumatic Control

As we described in Figure 6.7, the height of each actuator can be controlled through changing the state of a solenoid valve and a tap actuated with the servo motor.

To control each valve and servo motor, we used a 16 chan-

nel 12V relay module (SainSmart) and a 16 channel PWM Servo Motor Driver (PCA9685) respectively. The servo motor driver and relay module are controlled from the Arduino Mega Micro-controller.

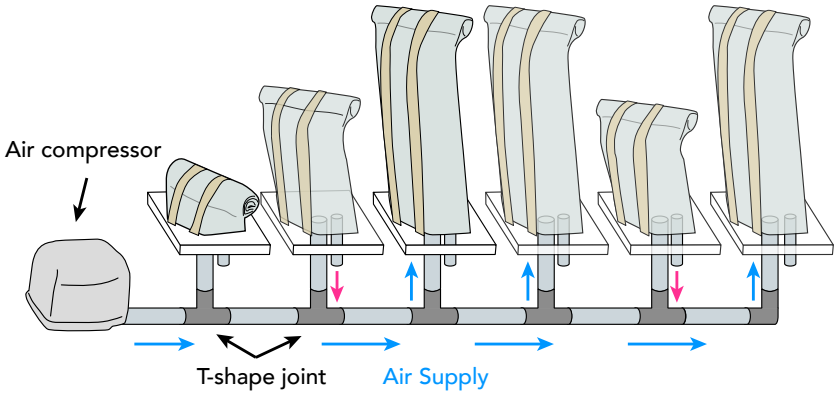


Figure 6.11: Pneumatic control system for actuator arrays.

A Realsense SR300 depth camera mounted on the ceiling captures the user's position for interaction. Based on the sensed data, the software selectively controls the valve and the tap of each actuator. Our client software written with OpenFrameworks visualizes the current height and control of each actuator, and communicates with the master Arduino through the serial command interface.

6.4 Application Scenarios

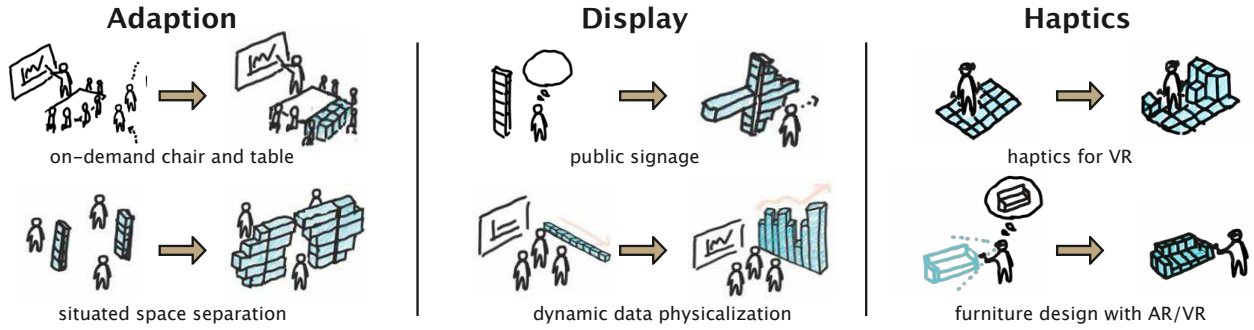


Figure 6.12: Example applications with LiftTiles.

Based on the insights from the expert review, we explore a

number of possible application scenarios for reconfigurable room-scale shape displays and propose the following three high-level interaction spaces. 1) *Adaption:* adapting to the user's needs and situation, 2) *Haptics*: providing feedback synchronized with visual information. 3) *Display:* communicating with the user by displaying information or data, and Based on these categories, we explore some possible application scenarios for each interaction space.

6.4.1 Adaptive Environments

Reconfigurable Architectural Space

The first application area is the ability of architectural spaces to reconfigure and adapt to the users' needs and context. We envision an architectural multi-use space that consists of LiftTiles floors and walls to render furniture and dividers on demand. For example, a room-scale shape display can transform the floor into a chair, a table, or a bed depending on an individual person's height, abilities and how they are using the space at any given time. This way, limited urban living spaces could transition between bedroom, a co-working office, gaming room, and dining room during different times of the day and days of the week.

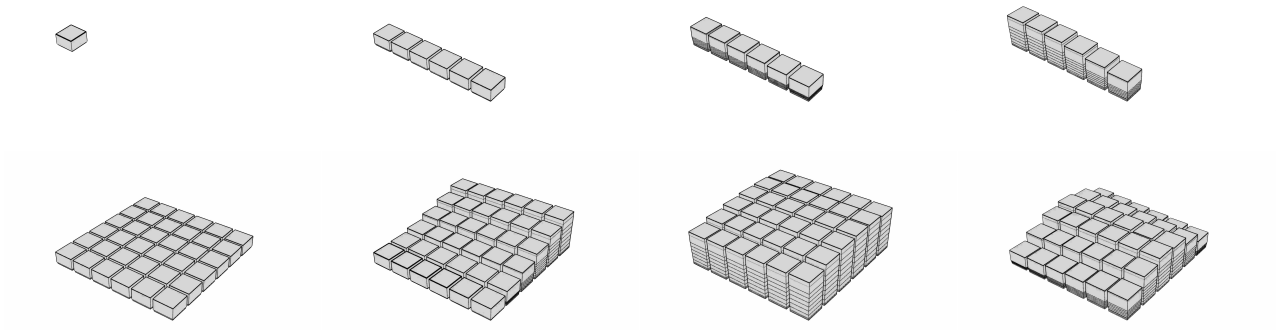


Figure 6.13: Modular and reconfigurable shape-changing tiles.

An interesting question in this application is how the room

interacts with users and objects while transforming between the different configurations. By sensing the persons presence, the room would avoid rendering elements where they currently stand, or even create barriers like a safety fence around a transforming section. Also, LiftTiles could be combined with custom built static objects. As an example, the system may render the legs of tables and chairs, onto which a lightweight tabletop and seating cushions are placed. These static objects could be used to issue commands to the system: When a user holds a seating cushion at a certain position and issues a command, LiftTiles renders legs at that specific location and height, and when removing the cushion, these legs automatically disappear.

Figure 6.14: Adaptive space separation. In a public working space, LiftTiles can create a temporal meeting space by separating the space.

Besides the user picking up and moving objects in the room, LiftTiles could also transform it's surface to slide or tumble objects into their desired position, similar to how prior work moves and assembles objects on smaller shape displays [Schoessler et al., 2015].

Deployable Pop-Up Structure

As the LiftTiles actuators are compact compared to traditional furniture and walls, they can be deployed in temporary use cases like festivals, trade show, concert stages, and disaster relief quarters. In this application, the actuators are shipped in a compact box and laid out inside an empty room, similar to

tiles. After connecting them to a compressor and control computer, the operator selects a use case and they render a layout of chairs, beds, tables, and partitions. This arrangement could be pre-programmed or it could be defined by the operator on site by specifying what shapes to extrude. After the event, the actuators contract into compact tiles and can be stored for future use.

6.4.2 Dynamic Haptic Environments

Haptics for VR

LiftTiles can improve virtual or augmented reality experiences by rendering room-scale haptic objects. For example, when the user puts on a headset and starts a game, the room can transform its shape to synchronize the physical landscape to the virtual one. The actuators can mimic furniture, walls, and bumps so that the user can feel the haptic sensation through walking and touching.

Figure 6.15: Room-scale dynamic haptics for VR.

This approach could also be combined with the idea of human actuation [Cheng et al., 2015], where human actors place objects to provide large-scale force feedback for a user in VR. For example, when the actors place the module at the instructed location, the module can automatically change its height to create a haptic proxy object like a table or a wall. Thus, by leveraging the reconfigurability and instructions for human volunteers that continuously rearrange modules in an open

area, the system can provide room-scale haptic sensations with a much smaller number of modules.

Full-scale Design Mock-up for AR

In domains like car design or interior design, it is important to see the object as a full-scale mock-up during their design process. For example, despite the dominance of CAD tools, car designers still create full-scale clay models for design modifications and reviews. To aid this design process, LiftTiles can provide an instant real-scale object mock-up, so that they can see, touch, and modify the car design. By combining LiftTiles with augmented reality headsets, realistic renderings can be overlaid on top of the physical mockup.

6.4.3 Information Display

Public Signage

In addition to adapting for functional use as furniture, LiftTiles can also communicate with the user by displaying information through shapes. For instance, LiftTiles acts as a public signage display by extending an array of horizontal actuators from a pillar. When a user asks for directions, the pillar transforms to render an arrow shape pointing towards the target location. By stacking multiple horizontal displays along a wall, they can render more expressive information and animation. When extending all of the actuators, these display sections cover the wall, when retracting them selectively, they can form a pictograph through negative space.

Dynamic Data Physicalization

LiftTiles can render data physicalizations, such as bar charts, 3D map data, and body-scale data sculptures. Such public large-scale data physicalizations can engage people and at-

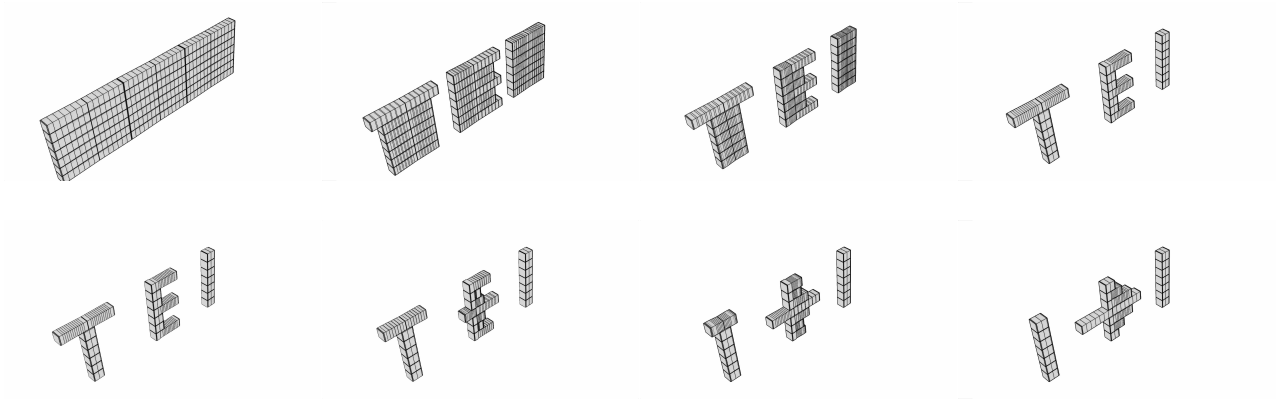


Figure 6.16: Shape-changing walls for information display.

tract attention. In addition, LiftTiles can provide interactive spatial data physicalizations. An example of such an application would be a data discovery room in a science museum. A ceiling-mounted projector projects a timeline of the last 2,000 years onto the floor. Children pick up a LiftTiles module and place it on a time period, then the actuator extends to represent the average body height of the same age at the given time period. This way, 10 years old girl can see and feel how children 100 years ago were of a different average height. Such body-scale data physicalizations encourage spatial thinking and discovery to relate to data in meaningful new ways [Jansen, 2014; Jansen et al., 2015].

6.5 Discussion

6.5.1 Expert Review

To understand the potential benefits and limitations of LiftTiles, we conducted individual expert reviews with three professionals: two architects and one interior designer. All of them had at least five years of working experiences in architecture or design offices. All three participants were male. We were particularly interested in the insights from their professional

point of view, and to identify potential applications for a tool like LiftTiles in their future practice. For each interview, we introduced the current prototype and demonstrated how it works. We then asked the participants open-ended questions about possible use cases and practicality. Each interview took approximately 60 minutes.

Needs vs Cost Effectiveness

When we asked about the deployability of LiftTiles for actual architectural spaces, while P1 stressed the need for reconfigurability, he was also concerned about its cost-effectiveness compared to existing reconfigurable furniture. *P1: "As a user and an architect, I think there is a strong need for transforming spaces. However, I think that it would not be cost-effective when actually trying to deploy it. For example, even if there is a desire to want something like a reconfigurable desk in the living room, simply placing a traditional desk is quicker and cheaper."* While all of the participants were attracted to the concept and stated it was interesting (P1) and possibly useful (P3), from their point of view the current prototype of LiftTiles would need further development to meet the cost and performance criteria before actually deploying it in the real world.

Permanent vs On-demand

All participants agreed that LiftTiles would be better suited as a temporary, rather then a permanent architectural building element. *P1: "I think that there will be a fundamental problem when actually using it as a permanent building. These elements may not meet the common requirements for permanent buildings, such as precision and working lifetime." P3: "Consider the static or deployable building elements, this definitely fits the latter category."* On the other hand, the participants did see a strong potential for deployable structures like temporary shelters and for use cases

where the users needs would dynamically change.

P1 gave a narrative story about his past experience *P1: "For example, you can put it in an evacuation shelter, and something that can ensure privacy in a simple way. I also think air can work as insulation for heating." P3: "A tent is usually not flexible enough, so this might provide a more flexible structure."*

P1 also mentioned that *"there are some projects that repurpose vacant houses for an on-demand place like shopping or community activities. Reforming the entire internal structure is usually costly, but these transformable structures can be easily deployed and changed according to space and needs, which seem interesting".* P2 suggested that LiftTiles would be interesting in a hotel lobby, where this type of a chair would attract guests. Other on-demand scenarios proposed included pop-up structures for events, such as for a circus (P1) or a concert stage (P3).

Full-scale Object Previews

Outside of the architectural context, one participant, identified is an interior designer, saw a strong potential for full-scale design and haptics. *P3: "When we design furniture, say a sofa, we usually need an actual physical object to check the real-scale size. We usually make this by cutting blue foam, but it's really tedious. This display seems able to make it for us, which is really helpful"* He also mentioned that this type of mock-up could be used by designers in other domains that work with full-scale models, such as a cars, boat layouts, or interior design.

Another interesting application for full-scale object previews was suggested for dynamic film studio sets: *P2: "It can be used for a set of film making. For example, by changing the color to green, these can be a transformable physical chroma key background, which helps actors perform."*

Interactive Playground

P2 saw potential for interactive applications that utilize the dynamic nature of LiftTiles. *P2: "I'm very interested in an interactive playground where children can play on the top, so that they can interact with this, for example, a child pushes one actuator, then the other actuator can pop out."* P3 also suggested dynamic visualizations, such as large-scale Google maps installation which could render terrain.

6.5.2 Limitations and Future Work

In our current prototype, there are a number of limitations that need to be addressed in the future work.

Open Loop Control

Currently, we are controlling the height of each actuator by estimating the time needed to inflate it with air. The stability would be much improved if a height (e.g., IR photo reflector) or pressure sensor provides feedback for precise, closed loop control of actuator height. We also noticed that sometimes two similar tubes are extending at a different pace, even pumping air from the same pump. By leveraging a closed feedback system that measures the current height and controls the valve, consistent speed and the control would be much increased.

Lack of Interactivity

Related to the above limitation, this paper focuses primarily on the actuation mechanism, and less on sensing input, which limits the interactivity of the current prototype. While we used a RealSense depth camera for position tracking of actuators and users, it only provided limited fidelity and therefore our demonstrated applications are based on pre-programmed shape transformations. We anticipate that embedded sensing for touch and pressure would greatly increase the interactive

capabilities in a future iteration of LiftTiles. Also, as we saw in the expert review, there are many interactive applications that can be connected with data (e.g., rendering a 3D map). In terms of interactive applications, we are also interested in a dynamic appearance by changing the color with LED. While we did not prototype these features, we think these color change can also provide some affordances. More input modality and interactive applications are promising for future work.

Load Tolerance

While the pneumatic actuator can withstand the heavy object, we noticed that our outer enclosure is not stable and robust enough, as we discussed in the system section (see Telescopic Enclosure). We put this enclosure to stabilize the top, but it is the weakest part in terms of structural robustness. This is because the telescopic enclosure was manually fabricated with the plastic cardboard. We believe more robust structure of the extendable enclosure can increase stability.

Speed of Transformation

One current limitation is the slow speed of the transformation. While there are some potential solutions, for example, using a stronger air compressor and larger valves, the speed of transformation at the large inflatables is generally slower than mechanical actuation. Most of the application scenarios we explored may not require the fast speed of transformation, but for some scenarios such as dynamic haptics for VR, the current configuration may not be fast enough. Multiple stronger air compressors can solve these issues, but the noise of the compressor will be another problem. We anticipate another actuation methods may be suitable for fast and highly dynamic shape transformation.

Modular and Mobile Shape Displays

Currently, each module needs to be connected to the central computer. Each control signal and power supply came from the central microcontroller and power sources. We are interested in having each actuator can have its own microcontroller and power supply, so that it can increase the modularity and flexibility. Moreover, we are interested in adding the locomotion functionality for each module, inspired by [Iwata et al., 2005]. The mobility allows an autonomous and distributed large-scale shape display which can deploy, extend, and store by themselves. We envision these modular and locomotive actuation units can further transform our physical space to dynamic and interactive environments for Human-Architecture Interaction.

Part II

Dynamic Shape Construction with Passive Collective Elements

In this part, I will explore dynamic shape construction with **passive collective elements**. Passive collective elements refer to elements that are externally actuated through environmental force or mechanical contact. Although a shape constructed by passive elements do not have an ability to transform by itself, these passive elements have a significant advantage of scalability of constructing elements — it is easier to scale up the number of elements. Also, the size of the elements can be small, as each element does not require internal power source or actuation mechanism.

One of the challenging of the passive collective shape construction is how we can actuate these passive elements for dynamic construction and transformation. This chapter explore this aspect through three projects: the first one is FluxMarker: externally-actuated swarm magnetic elements with PCB-based electromagnetic arrays, the second one is Dynablock: dynamic block assembly with passive magnetically connectable custom-built blocks, and the third one is RoomShift: dynamic spatial reconstruction with swarm robotic actuation.

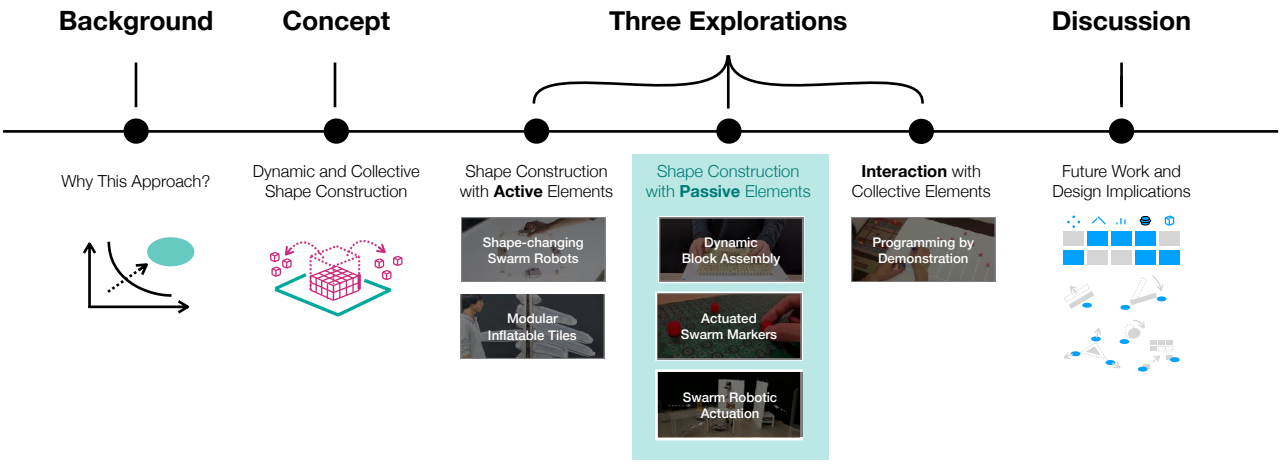


Figure 6.17: Part II: Dynamic shape construction with passive elements.

7

Dynamic Shape made of Externally Actuated Swarm Markers

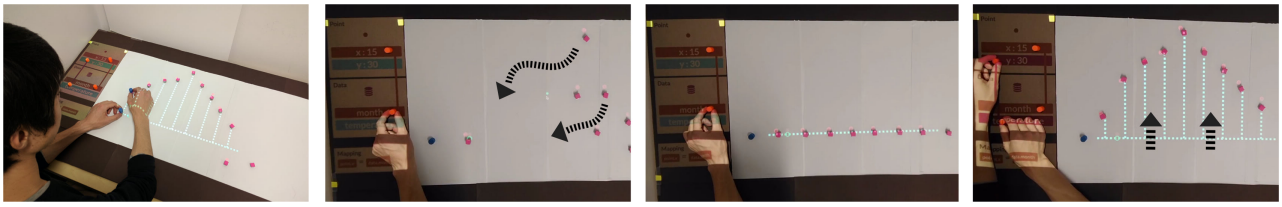


Figure 7.1: Reactile exemplifies dynamic shape construction with a swarm of externally-actuated magnetic elements [Suzuki et al., 2018a, 2017].

7.1 Overview

In this chapter, I demonstrate how we can construct a dynamic shape with spatially aligned points of externally actuated passive elements. Constructing a shape with sparse dots representation is often demonstrated through self-movable swarm robots that can freely move on a tabletop surface. Active collective elements, like the swarm robots, usually have limitations in size and scalability. For example, the size of each element is

bound to the size of motors, batteries, and an internal microcontroller, which is often difficult to be smaller than a centimeter. Also, the cost of self-actuated elements would be higher than the passive elements as each element needs highly sophisticated electronics and mechanical systems, this also introduces the problem when scaling up.

This project demonstrates how we can leverage a swarm of passive objects, for example a swarm of magnets, as an externally-actuated collective elements to construct a dynamic shape for displaying information and user interaction. To demonstrate this concept, I developed Reactile, an electromagnetic actuation board that actuates a swarm of small magnets to collectively move and display shapes on a tabletop surface.

The main contributions of this chapter [1] [2] are

1. Design of low-cost, scalable PCB-manufactured electromagnetic coil arrays for external actuation mechanism.
2. Hardware and software system that can track and actuate multiple passive magnetic markers in the X-Y grid position on a surface.
3. Exploration and a user evaluation of accessibility applications for blind users.

[1] Suzuki, R., Kato, J., Gross, M. D., and Yeh, T. (2018a). Reactile: Programming swarm user interfaces through direct physical manipulation. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 199. ACM

[2] Suzuki, R., Stangl, A., Gross, M. D., and Yeh, T. (2017). Fluxmarker: Enhancing tactile graphics with dynamic tactile markers. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 190–199. ACM

7.2 Externally-actuated Passive Objects

7.2.1 Possible actuation types

Many different actuation techniques can enable the actuation of passive objects with external force. We reviewed a variety of actuation approaches that have been proposed in different areas such as tangible user interface, robotics, and accessibility. Possible actuation methods include mechanical actuators (e.g.,

DC motors, servo motors, stepper motors), piezoelectric actuators (e.g., piezo-elastomer, piezo-electric linear motor, ultrasonic motor), electrostatic actuation, pneumatic and hydraulic actuation, acoustic levitation, magnetic actuation, electromagnetic actuation.

Among these options, electromagnetic coil generates strong actuation force that is enough to actuate passive magnets. PCB manufactured electromagnetic actuation scales relatively well because many coils can be aligned on a PCB. For example, in our design an 8x8 array of coils can be aligned on a 10cm x 10cm PCB, costing only 0.50 USD. While the cost of printed circuit board increasees with the size of the board, the cost increase is trivial, and the cost of transistors to drive a high current for electromagnetic actuation is also inexpensive compared to mechanical or piezo-electric components.

In addition to cost and scalability, we value the simplicity of fabrication and control mechanism which allows the larger accessibility community to quickly adapt, replicate, and test. As mentioned above, pneumatic and hydraulic actuation methods are also promising approaches. The complexity of design and fabrication of hydraulic actuation can be alleviated with advanced 3D printing technology [MacCurdy et al., 2016], but it is still difficult to design complex fluidic circuits that can control the multiple pixels individually. In short, the fabrication and control mechanisms of such pnuematic actuated devices are a challenge. In contrast, using electromagnetic coils leverages commercially available PCB manufacturing for fabrication, and a standard circuit design for the control mechanism. Thus, we chose to develop an electromagnetic actuation technique that meets all the considerations we identified previously, while allowing the simple control and fabrication process.

7.3 Reactile

To demonstrate this concepts, we developed Reactile, a PCB-based electromagnetic actuation board that actuates a swarm of passive magnetic markers, which can represent a sparse dots shape on a 2D surface.

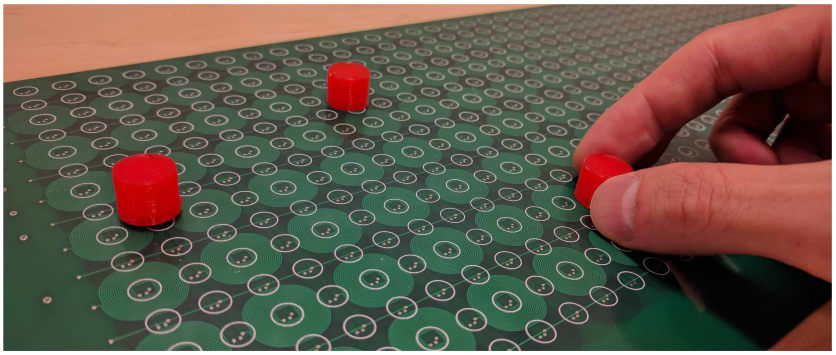


Figure 7.2: Reactile uses a field of electro-magnetic coils fabricated with a standard PCB manufacturing. Each board has 16 x 40 coils and the final prototype uses five boards to cover 80 cm x 40 cm area with 3,200 coils. This board can actuate passive magnetic markers shown as red objects with 10 mm diameter.

To enable tracking magnetic markers as well as user interaction, Reactile uses a set of distinctively colored markers using a mounted standard RGB camera and computer vision techniques. With the computationally controlled actuation and computer vision tracking, Reactile allows the closed-loop control of the externally-actuated swarm display. In the following, we describe the hardware and software design and implementation of Reactile system.

7.3.1 PCB-Based Electromagnetic Actuation

In Reactile, a user interface consists of a swarm of passive magnetic markers which move on a 2D workspace driven by electromagnetic forces. Reactile uses a grid of electromagnetic coils to actuate these magnetic markers. Running current through the circuit coils generates a local magnetic field so that each coil can attract a single magnet located within its area.

Each coil is aligned with a certain offset in both horizontal

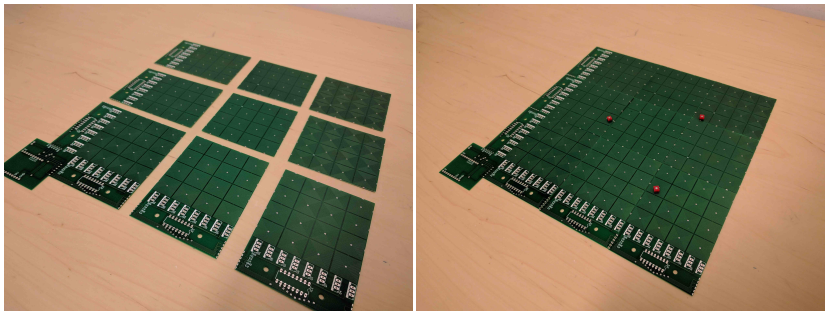


Figure 7.3: Modular boards can be aligned together side to side as tiles, allowing the overall size of the coil array to be large.

and vertical direction with an effective area overlap, which allows the coil to attract the magnet located in the adjacent coil. We design electromagnetic coil arrays to be fabricated with a standard printed circuit board (PCB) manufacturing (Figure 7.2). This reduces the cost and fabrication complexity, making it easy for the actuation area to scale up.

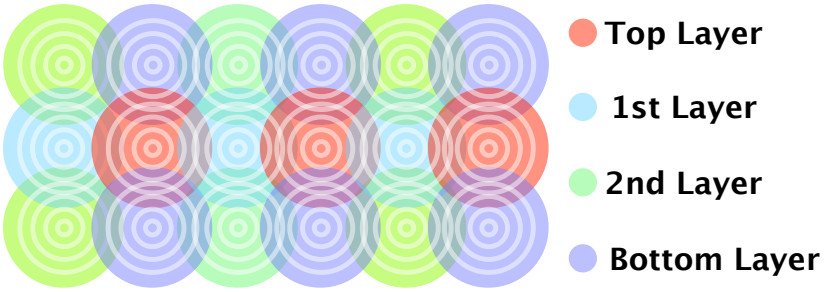


Figure 7.4: A simplified schematic of our coil design of a 4-layer PCB. Each layer has a set of coils aligned with a certain offset in both horizontal and vertical directions. Each coil is 15 mm diameter and has 2.5 mm overlap between nearby coils.

7.3.2 Coil Design

Figure 7.4 shows the simplified schematic of the coil design. Our PCB design is a 4-layer board, and each layer contains a set of coils, each of which has an identical circular shape with a 15 mm diameter and a 2.5 mm overlap between nearby coils. Each coil has 15 turns with 0.203 mm (8 mils) spacing between lines, and the distance between centers of two coils is approximately 10 mm, which makes a 10 mm grid for attractive points. Due to the maximum size of the PCB facility we used, a single board has 40 x 16 coils which approximately covers a 40 cm x 16 cm area. We design the actuation board to be scalable,

so that we can extend the effective area without any design changes. The final prototype covers an 80 cm x 40 cm area with 80 x 40 coils by aligning five identical boards horizontally. The fabrication of each board costs approximately $80 USD, including manufacturing of PCB and electronic components.

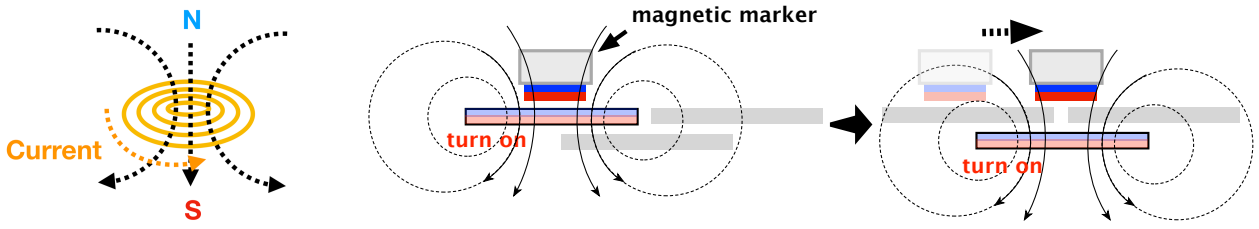


Figure 7.5: An actuation mechanism of Reactile. Running current through the coils generates a local magnetic field to attract magnetic markers located within its area.

7.3.3 Passive Magnetic Marker

Each marker consists of an N48 neodymium disc magnet and a 3D printed cap. As shown in Figure 7.5, the magnet is attracted with a local magnetic field generated by nearby coils. The basic requirement for a magnet is that its size is large enough to overlap with nearby coils (Figure 7.5). Thus, the minimum size of magnets depends on the size of the coil and offsets. In our prototype, the minimum size of the magnet is 6 mm diameter, and we used magnets with 10 mm diameter.

All electromagnetic coils generate the same direction of a magnetic field to attract magnetic markers, similar to [Strasnick et al., 2017]. Thus, each magnet is directed in the same direction (e.g., the north pole faces up and south pole faces down). As all the magnets face the same direction, they are prevented from attracting and connecting with others. The magnetic markers repel each other if the distance between two markers becomes closer than a certain distance. The minimum distance between magnets depends on the diameter and strength of the magnets, and in our prototype, this minimum distance is approximately 30 mm.

7.3.4 Circuit Design

To produce a local magnetic field, we switch on the current for each coil. As our board has 80 x 40 coils, it requires 3,200 switches to control each coil. To reduce the required switches, we adopt a multiplexing technique for efficient current control.

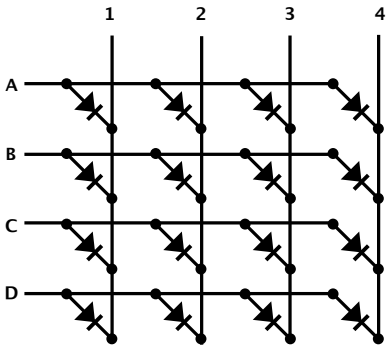


Figure 7.6: Multiplex coil matrix, similar to a LED matrix display.

Similar to LED displays, this approach only requires 80 + 40 switches to control 3,200 coils. On the other hand, this allows us to control only one row at a time; By switching the current, it can move multiple markers with a relatively high refresh rate. In our settings, the system switches the current in 100 ms for each marker. For example, if there are 10 markers in different row, it takes approximately 1 second (=100 ms x 10) to move them independently.

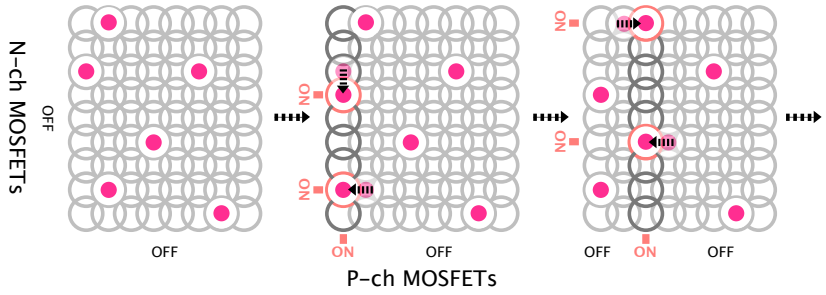


Figure 7.7: A control mechanism with push and pull pair of P-ch and N-ch MOSFETs. While only one column (or row) can be turned on at each time, switching with fast refresh rate (10Hz in our settings) allows to move multiple magnets nearly simultaneously.

To switch the current on/off for each row and column, we use the push and pull pair of P-channel and N-channel power MOSFET transistors. To run the current through a coil, the gate voltage of P-ch and N-ch MOSFETs should be set as LOW and HIGH respectively. For example, to turn on the coil at column 10 and row 8, we set P10 as LOW and the rest of columns (P-ch) as HIGH, and N8 as HIGH and the rest of the rows (N-ch) LOW.

The gate voltage of each MOSFET is controlled by daisy-chained shift registers. The five boards share the same data, latch, and clock pins of the shift register, so that only six pins

are required to control the 80 x 40 coils in the entire board. The shift register is controlled by an Arduino microcontroller, which communicates with a host computer through I2C communication.

We used 74HC595 for 8-bit shift registers, MSS2P3 for diodes, and AO3401 and AO3400 for P-ch and N-ch MOSFETs respectively. All electric components are surface-mount parts and are attached to the bottom layer; therefore, the top layer is flat to allow the markers to move freely. The source voltage for P-ch MOSFET comes from a 5.5V external power supply, and the average and peak current for each coil were 0.4A and 1.2A respectively.

7.3.5 Marker Tracking and Control

To track the markers' positions, we use a standard RGB camera and computer vision techniques. The software first extracts an image of the workspace by detecting white color and finding the contours in the image. Then, we approximate contours with polygonal curves to obtain the positions of the four edges of the rectangle workspace. After extracting four edges of the rectangle workspace, the system warps the input image with a geometric transformation to eliminate distortion and fits the image to the rectangular workspace.

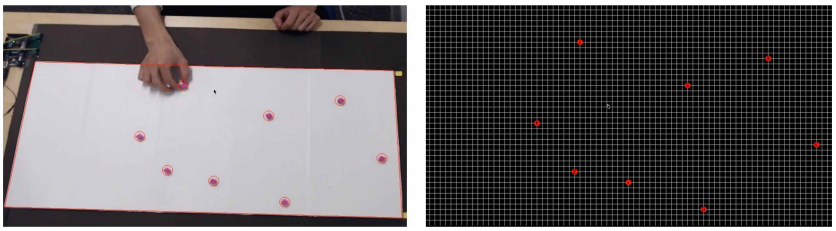


Figure 7.8: We use computer vision to detect a rectangle workspace and positions of the markers. The system uses detected position information within 80 x 40 grid for path planning and controlling marker movements.

To make it easy to track swarm markers, we color them in high contrast colors. To track markers in an image, we first convert the image's color scale to hue, saturation, and value (HSV) and

detect a specific color with a lower and upper threshold for each value. Then, the input image is converted to a binary-colored image where the detected color is white and the rest is black. The detected colored marker position is then calculated as a relative position within the workspace by dividing its horizontal position by 80 and vertical position by 40. We use this technique to detect the standard red markers as well as the other special markers including constraint markers (blue) and selection markers (orange). Figure 7.8 left illustrates the input image captured by the camera and detected workspace highlighted with a red-lined rectangle. Figure 7.8 right shows the position of each marker projected onto an 80 x 40 grid based on the warped workspace. We used OpenCV for computer vision and Logitech C920 for the RGB camera, which is mounted 100 cm above the table.

7.4 Application Scenarios for Accessibility Assistant

One interesting application of this system is an accessiblity assistant for blind people. For people with visual impairments, tactile graphics are an important means to learn and explore information. However, raised line tactile graphics created with traditional materials are static. While available refreshable displays can dynamically change the content, they are still too expensive for many users, and are limited in size. Threfore, we explore how this inexpensive scalable method can renders dynamic content on top of static tactile graphics with movable tactile markers.

These dynamic tactile markers can be easily reconfigured and used to annotate static raised line tactile graphics, including maps, graphs, and diagrams. We explored four possible ap-

plications: location finding or navigating on tactile maps, data analysis, and physicalization, feature identification for tactile graphics, and drawing support. We then evaluate our prototype with six participants with visual impairments.

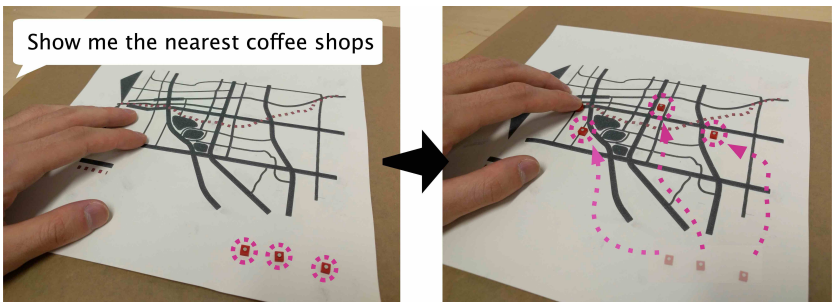


Figure 7.9: Externally-actuated passive markers enables blind users to touch the information. For example, combining with tactile maps, these markers can point out spatial locations on top of the static tactile map.

7.4.1 Location Finding and Feature Identification

Tactile maps provide blind people a means to explore geographical information. For example, a tactile map of the campus will display a layout of buildings and braille labels associated with each building. However, finding a particular location is often a tedious task; unlike a sighted person's ability to scan a map and quickly identify a specific location, blind users usually explore the map sequentially and must orient themselves to the whole graphic before finding a specific location. Moreover, although the information is often labeled with braille, reading braille takes time and is inaccessible for those who cannot read braille. Audio feedback can help to orient users to the name or feature of the current location, however, this technique makes it difficult to orient oneself to specific locations on the page.

Dynamic tactile markers can help to identify a spatial location quickly. For example, responding to "Where is the nearest coffee shops?" in a local area map or "Where is the Black Sea?" in a geographical world map, the dynamic tactile markers can move around on the tactile map, and the blind user can use their hands to quickly skim the map to identify the location of

the marker. They can quickly find the marker position relative to their current location or an outline of surrounding areas on an existing tactile map. In this way, they do not lose contact with spatial reference points or the spatial memory they have developed. In addition, responding to the query, "How can I get to this place?", other markers can instantly draw the tactile path by aligning dots on the map. Once the user is satisfied by finding the location or route, the dynamic tactile markers can be reset, and cleared from the tactile graphic.

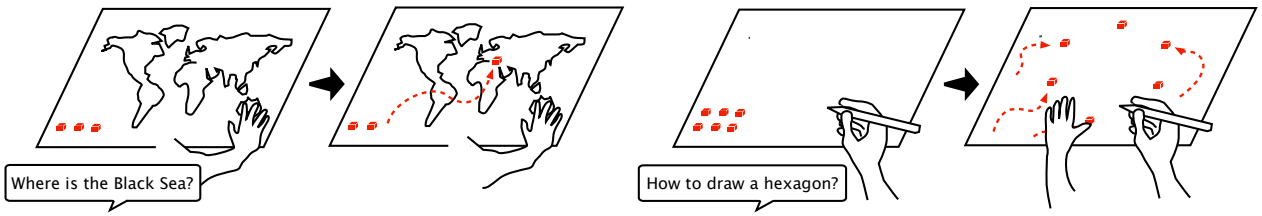


Figure 7.10: Location finding and drawing assistant.

Similar to location finding, the dynamic tactile marker can be used to locate a specific feature on a tactile map based on a user's question. For example, a student with visual impairments is given a brain model to use in her biology class. She can ask "which region of the brain has a memory function?", the dynamic tactile marker can point out the domain of a hippocampus by positioning the marker within that region of the organ. This is a similar, but different interaction from existing interactive tactile graphics, which explains the feature of each region triggered by the user's pointing, while the dynamic tactile marker can point out the location triggered by the user's question. In another scenario, a student is in a lecture, and the professor is presenting a graphical representation of a cell via PowerPoint, and uses a laser pointer to identify the cell nucleus for the sighted students. The student with visual impairment, who has a tactile version of the graphic, can ask the dynamic tactile marker to move to the corresponding location.

7.4.2 Data Analysis and Physicalization

Data analysis is one of the most challenging tasks for people with visual impairments. As the visualized data is not accessible for blind users, they often find it difficult to interact with the data. Dynamic tactile markers can help blind users make sense of data through data physicalization [3].

[3] Jansen, Y. (2014). *Physical and tangible information visualization*. PhD thesis

One advantage of using dynamic tactile markers is the ability to update the data for a different context. For example, a blind user who wants to analyze the temperature of a city over time might want to know the pattern throughout the year, maximum temperature, and minimum temperature of the city. Twelve dynamic tactile markers can position themselves to display a plot graph to represent the temperature data of each month. By touching the data point and referring to the scale, which can be given by a static embossed paper, the user can find the maximum and minimum temperature of the city. While understanding the pattern of the data can be challenging with audio representation alone, with dynamic tactile markers, she can also comprehend the pattern of the graph by recognizing spatial positions. If she wants to analyze the temperature data in a different city, she can just ask "render the data point" with the city name. Then, the dynamic tactile markers can be repositioned to render the requested data point.

7.4.3 Guided Drawing Assistant

In addition to supporting an interpretation of content or analyzing the data, the dynamic tactile marker can also support students to create their own tactile graphic representations. Many students with visual impairment have limited exposure to drawing or making their own representations of information due in part to the lack of educational practices and materials. The dynamic tactile marker can help blind users to make their

own tactile representations by guiding them with reference points of the drawing.

For example, when a blind user is trying to draw a hexagon, six dynamic tactile markers would appear, marking the reference points of each corner of the shape. The user can touch the markers to position themselves with the non-dominate hand, and guide them to draw the line to the next point (Figure 7.10). Or, the tactile markers can form a nearly solid edge that the user could mark alongside. This guided drawing can be particularly useful when creating their own tactile graphics when used in conjunction with inexpensive physical tactile drawing boards such as the Sensational Art Board, the inTact Sketchboard and 3D printing Doodle Pens.

7.4.4 User Evaluation

We evaluated our prototype with six people with visual impairments to investigate the plausibility of the application scenarios identified during our formative work. We found that all participants were able to use the markers to identify specific features on the tactile graphics faster than when they did not have a reference point, albeit they wanted to have the markers move along paths to guide them between landmarks.

Participants

Six people with visual impairments participated in the user study (3 male, 3 female); three participants were also part of the earlier formative study. P1 (male) P2 (male) P3 (female) identified as being totally blind. One female participant identified as being legally blind with a little bit of light perception (P4). One male (P5) and one female (P6) participant identified as having a visual impairment, but had functional vision through the use of assistive technologies.

Method

We conducted a 45-minute session with each participant. During each session we presented an overview of the research, introduced the prototype and described how it worked in conjunction with the graphics, and then showed the participant two embossed tactile graphics from the local university's accessible media lab so that they would have basic familiarity with the graphics.

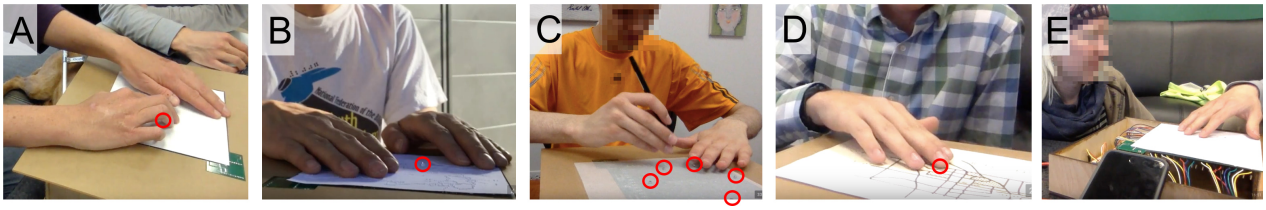


Figure 7.11: User evaluation with six blind users. Participants used our system to identify a point of interest, explore a sectional view of a human brain, draw a shape, and navigate a street map, where red circles indicate the positions of one or more markers.

The graphics included (A) an embossed tactile map of Eastern Europe and Russia and (B) an embossed tactile graphic representing a sectional view of a human brain. At the beginning of the session we provided the participants with the context in which these graphics might be used and provided time for them to explore the graphics. We then asked the participants to (C) draw a hexagon on a piece of trace paper, in order to observe their familiarity with drawing without any aids. Figure 7.11 shows examples of user study sessions.

They were also interested in using the markers to create raised lines around specific tactile elements so that they could feel the boundaries and the contained tactile information. Our participants also noted the possibility for the system to annotate graphics in real-time, which would help understand their data sets, interpret tactile graphics at the same time as teachers present the same information visually during lectures, and with building in-situ ways to navigate. Finally, our participants confirmed that the markers would help people learn to draw, in particular, young students.

7.4.5 Findings and Discussion

In order to assess the application, we asked participants to use the tactile map, tactile graphic, and drawing paper to preform a task with the tool. Each participant performed those tasks in slightly different ways, and provided unique feedback and new ideas about the effectiveness of the tool.

Spatial Navigation: When viewing the tactile map, P1 rapidly scanned the display area with two hands and found the boundaries of the countries represented on the map without guidance; he said that he loves geography and is good at geometry. He immediately started looking for the Black Sea, at which point we used the system to help him locate the sea. He found the marker within seconds and noticed that it was positioned in the middle of the sea. P1 compared his experience with this tool as being similar to working with a teacher of the visually impaired (TVI), who might manually place a simple magnet or sticker on the map to mark a location. He suggests that we use the system to guide someone to follow a path in order to discover a landmark, in this case *"the Yangtze River"* if this map also included China. At the end of the user study he said *"The best application I could see this being used for is to have the marker move with the user following along, so that the teacher could trace a path out for me in real-time."*

P3 also identified real-time mapping as an important application of the tool. She suggested *"If you could have a tactile map, where then you could locate two buildings [using the markers], and then figure out pathways between those markers [which represent the buildings], you could then start populating the map with landmarks using these markers."* P5 and P6, both low vision, explored the tactile graphic visually and did not have any ideas for how this tool would support them with navigation.

Feature Identification and Locating: In addition to using the markers to identify specific landmarks or geolocations, P3 wanted the markers to form into a raised line around specific regions of the tactile graphic to make the boundaries more amplified. When viewing the tactile map with the marker, it was located in the middle of an empty space. She asked, *"I was wondering, is the marker in the middle [of the country]?"* She then suggested that the dynamic markers would be more beneficial if they could outline the boundary of the country or entity one was trying to find.

When using the dynamic markers to explore the map, P2 started brainstorming other possible applications. He provided the scenario of using Google to find restaurants with four stars, and the then using the system to automatically populating the location of the restaurants to narrow his decisions about where to go.

When using the markers to find Kazakhstan on the tactile map, P4 indicated that the markers provide a sense of independence. *"It works better than having another person poking at the spot. Even if you know where their finger is, and you start taking time to explore around, they might think you are lost–which you are not–and try to show you around."* She also mentioned that if an instructor was talking about a specific location on a graphic or map, it would be easier to keep up if the marker was in the corresponding position on the display. *"This would be useful if it was synced up with a lecture and graphics, or even if it was synced with an instructors laser pointer; if it was tracking what was up on the board, and I could follow along, that would be amazing."*

P6 elaborated on this concept, indicating that as somebody with low vision, she has a hard time following lectures that have slides. *"I usually ask the presenter to give audio cues when they are changing slides so I can follow along with the slides in front*

of me, but they usually forget to do that. Or if they have annotated graphics and they forget to describe that...then this tool would be very helpful. If this could be used to help track those animations on a print out of their slides. I think this would be great for low vision."

Data Analysis and Visualization: P3 mentioned wanting to be able to rapidly zoom into a specific area of a graphic, and see the details in higher resolution. She envisioned that the user could specify a region of a graphic, and the system could dynamically represent the zoomed in region in higher resolution next to the original view. P2 suggested that the system could be used to display incoming financial data from wall street to show how the market fluctuates.

P4, an astrophysicist, remarked that while the system is not yet useful for her work, she enthusiastically recommended connecting the system to Excel so that she could create diagrams of her data in real-time using multiple markers. *"If they [the markers] were more stable and you had an ordinary piece of tactile graphic paper, you could add the markers on top of that and make a line graph. That would be very useful. That would absolutely be helpful to me in my professional career. On of the things that happens when I am writing a paper is preparing my data in Excel, I can't get any feedback from that graph. If I had something like this plugged into my computer, I could see if it graph matches my numbers. I could actually check my own work before publishing. That is a big deal!"*

Drawing or Tracing Guide: All participants found the task of following the dynamic marker to draw a hexagon to be slow and troublesome as they already had strategies for how to draw the shape. For example, when drawing a hexagon freehand, P4 first drew a square and then used her spatial understanding to add additional sides. She remarked that it was not different from following a real person and it did not

provide the tactile feedback that a drawing board would. P2 remarked that the system would be difficult to use to draw organic shapes due to the orthogonal layout of the coil grid. He also remarked that it would be more useful if the markers could be closer together. However, P1 pointed out that it could provide a sense of independence for people who want to practice drawing on their own; P3 thought that the system would be a good resource for young children learning to draw.

Technical Evaluation

The user study provided us an opportunity to evaluate the technical characteristics of the system. Here we discuss users' comments with respect to support, perceptibility, and scalability:

Support: The design of the physical prototype enabled the user to directly place embossed tactile graphics on the PCB display. During the user study, we found that the markers moved well along embossed paper graphics, but Swell paper was too thick for the current level of magnetization to maintain contact with the display. Throughout the user study the participants remarked that the markers felt loose and that they moved too easily when touched; they wanted them to have a stronger magnetic connection to the PCB display. P4 remarked *" If I was taking a test and was in a hurry looking around, and moved the marker, it would slow me down." We noted that in some cases this made participants hesitant to freely explore the graphics."*

Perceptibility: P4 and P5 explicitly said something about the markers. P4 noted that the marker stands out in comparison to the rest of the page. P5 wanted to make sure that they would not cover any important graphic content. P1, P3, and P4 each remarked on the heat of the magnets and underlying PCB display. When looking for the marker P4 said, *"In some ways*

I think the heat is a good indicator of where the marker is going to be...because the marker is a very specific spot, but the heat is a region." The heat she was referring to is the result of current flowing through the coil, which turns out to be a useful side-effect for P4. In contrast, P2 found the heat less favorable and noted that if we increase resolution of the markers, the heating problem would need to be addressed.

Scalability: All participants were impressed by the low cost of the prototype and the prospect of a larger display size. P1 mentioned that if the resolution were lower (the markers more spread out), a mechanism would be needed to help users find their starting reference point. Without this it would be laborious to find the marker. P3 was satisfied with the current size of the display since she could explore the whole display in a short amount of time.

8

Dynamic Shape Construction with Parallel Block Assembly

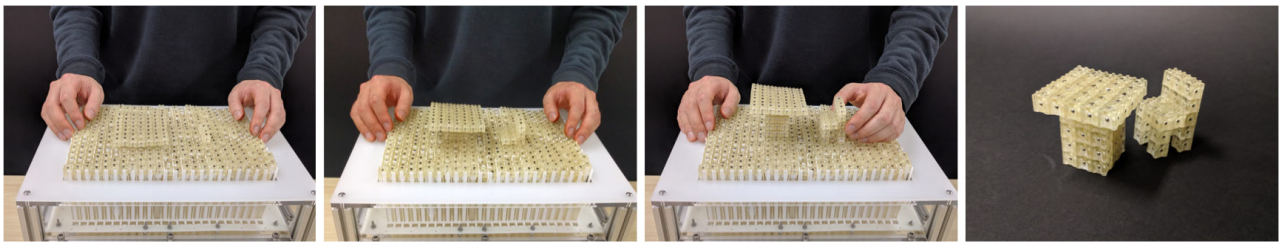


Figure 8.1: Dynablock is a rapid and reconstructable shape formation system, comprised of a large number of small physical elements [Suzuki et al., 2018b].

8.1 Overview

In the previous chapter, I showed that collective actuated elements can construct 2D shape with spatially distributed markers. Then, how can we go beyond 2D shapes towards 3D shapes? This chapter discusses a method to construct dynamic voxel shape with collective passive elements.

The voxel representation of dynamic shape has many advan-

tages over the other presented representations (e.g., sparse dots, lines, and pin arrays). For example, this allows the user to grasp a constructed object and interact with it. This is difficult with previously discussed shape representations, as the constructing elements are not physically connected each other, thus the user cannot grasp an object as a whole. Also, the voxel representation can also represent any three-dimensional geometries, such as overhanging structures, which cannot be represented with pin array representation.

Traditionally, creating the dynamic voxel shape is one of the central themes in programmable matter research. For example, self-reconfigurable modular robotic research take an approach to using modular robots as a voxel to create various shapes. In this approach, self-actuated robots dynamically move and connect with other robots to reconfigure the overall structure. However, they are typically suffered in the resolution of the constructed shape, as the size of each element usually becomes large due to the required electro-mechanical components, as well as the number of available elements are also limited due to the high components and fabrication cost.

In this chapter, I explore different approach for dynamic shape construction for voxel representation [1]. I investigate how we can leverage externally-actuated passive blocks as elements to construct a dynamic shape. One of the challenges when using passive elements is that how we can actuate many passive elements simultaneously because, as we discussed, the parallelism of the actuation can significantly affect the the dynamicity of the shape construction. To this end, I propose a new shape construction architecture, called Dynamic 3D Printing. Dynamic 3D printing is a new way to rapidly assemble passive blocks by leveraging a parallel assembler. Similar to the existing 3D printing, dynamic 3D printing assembles blocks one layer each time, and then stack them layer-by-layer to construct

[1] Suzuki, R., Yamaoka, J., Leithinger, D., Yeh, T., Gross, M. D., Kawahara, Y., and Kakehi, Y. (2018b). Dynablock: Dynamic 3d printing for instant and reconstructable shape formation. In *The 31st Annual ACM Symposium on User Interface Software and Technology*, pages 99–111. ACM

an arbitrary 3D shape with voxel representation.

The important point is that this approach can assemble one layer at the same time, so that this can drastically decrease the construction time of one layer from $O(N^2)$ to $O(1)$, compared to a sequential assembling, where N represents a number of block elements for width and height of each layer (Figure 8.2).

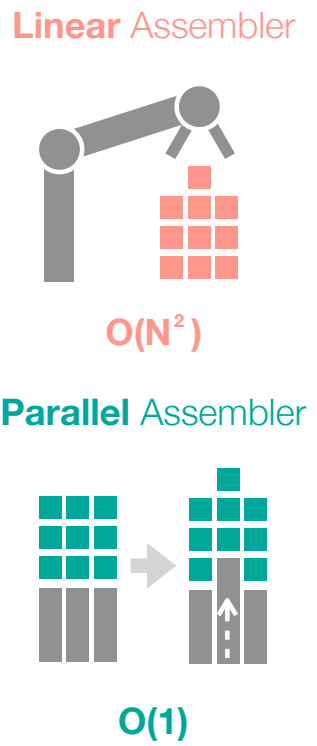


Figure 8.2: Serial vs parallel assembly. If the system can assemble one layer at once, the assembly time will significantly decrease.

We call it dynamic 3D printing because the shape creation process is similar to common 3D printing techniques such as stereolithography 3D printing, in which the shape is created by layer-by-layer from the bottom. However, in contrast to traditional 3D printing, dynamic 3D printing is fast — it can construct objects in seconds —- and the elements can be reused by deconstructing and reconstructing the object. Due to the resolution and stability of the currently constructable objects, dynamic 3D printing does not aim to replace the existing 3D printers. Rather, we envision the future where a 3D printer would become an interactive medium, rather than merely a fabrication device. For example, such a 3D printer could be used in a Virtual Reality or Augmented Reality application to dynamically form a tangible object or controller to provide haptic feedback and engage users physically. For children, it could dynamically form a physical educational manipulative, such as a molecular or architectural model, to learn and explore topics, for example in a science museum. Designers could use it to render a physical product to present to clients and interactively change the product's design through direct manipulation. In this vision, Dynamic 3D printing is an environment in which the user thinks, designs, explores, and communicates through dynamic and interactive physical representation.

To demonstrate this idea, we developed Dynablock, a hardware and software prototype of dynamic 3D printing. Shapes

made with Dynablock consist of 9 mm blocks, which connect to neighboring blocks with embedded permanent magnets. Dynablock's hardware employs a 24 x 16 pin-based shape display as a parallel assembler. 3,072 (= 24 x 16 x 8 layers) blocks are stacked atop the shape display, and each motorized pin pushes up blocks to assemble the object layer by layer. When blocks are inside the assembler, separators keep their horizontal magnets disconnected. As the blocks are pushed upwards and out of the assembler, they connect with their neighbors magnetically to form an object. Due to weaker vertical magnetic connections, blocks can disconnect vertically during this process to form overhangs. Therefore, Dynablock can assemble arbitrary and graspable 3D shapes with overhangs, rather than the 2.5D shapes of existing shape displays [Follmer et al., 2013]. Given a shape of 3,000 elements, Dynablock drastically reduces the assembly time to seconds, which enables interactive applications that conventional 3D printing techniques do not support. Moreover, the generated objects can be disassembled into individual elements for our system to reuse to assemble the next shape. In this chapter, I will describe the design architecture of dynamic 3D printing and implementation of Dynablock.

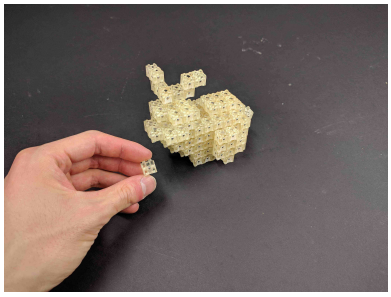


Figure 8.3: Dynablock, a proof-of-concept prototype of dynamic 3D printing.

8.2 Dynamic 3D Printing

8.2.1 Definition

First, we define Dynamic 3D Printing as a class of systems that have the following properties:

- **Immediate**: The system can form a physical shape in seconds.
- **Reconstructable**: Rendered shapes can be disassembled and reconstructed by hand or with the system, and the blocks

are reusable.

- **Arbitrary Shapes**: It can create arbitrary three dimensional shapes.
- **Graspable**: The output shapes and structure are graspable and solid.

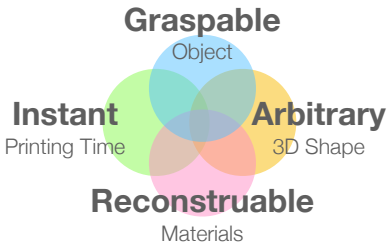


Figure 8.4: The definition of dynamic 3D printing

Dynamic 3D printing differs from existing 3D printing in speed and reconstructability: Dynamic 3D printing forms shapes in seconds, rather than minutes. In addition, because individual elements can be disconnected, the shape can be easily disassembled into its basic building blocks once the object is no longer needed.

8.2.2 Challenges

To meet the requirements of Dynamic 3D Printing, several challenges in design and engineering must be overcome:

- **Resolution:** To render physical objects realistically, Dynamic 3D Printing must support relatively high resolution. For example, for objects that can be held in the hand, each physical element must be millimeter scale, and ideally one order of magnitude smaller.
- **Scalability:** Resolution also dictates the number of elements that are used to form the object. For example, if an element is 1 mm in size, then on the order of one million (= 100 x 100 x 100) elements are needed to build a 10 cm^3 structure. Therefore each element must be inexpensive and easy to manufacture.
- **Speed:** Rendering objects with thousands or tens of thousands of elements places stringent constraints on speed. If the time needed to render an object scales linearly with the number of elements then it will be difficult to support uninterrupted and seamless interaction.

- **Stability:** Objects produced by the Dynamic 3D Printer must be sufficiently stable and robust for users to grasp them. Thus, the connection between the elements should be strong.

8.3 Designing a System for Dynamic 3D Printing

In this section, we outline a design for a hardware system to enable Dynamic 3D Printing. We describe the two key components to achieve the requirements: 1) A parallel assembling method and 2) a fast connection and disconnection mechanism. We outline the design considerations and possible methods for both features.

8.3.1 Parallel Assembler

Dynamic 3D printing deploys a large number of small discrete material elements, which are assembled to form arbitrary shaped macro-scale objects. Individual elements are passive, which requires an external actuator to perform the assembly. A straightforward way to assemble these material elements is the pick-and-place [Maeda et al., 2016; Sekijima and Tanaka, 2015] method or depositing single elements [Hiller and Lipson, 2009] (e.g., using a 3DOF robot arm), similar to an FDM 3D printer. However, in this method assembly time scales with the cube of the dimensions and assembling many elements would take a long time. Alternatively, we explore a parallel assembly method that can create an entire layer of an object at once. We considered several designs for parallel assembly.

The first design uses a pin-based display to push elements into place and then connect them. As illustrated in Figure 8.5, the assembler consists of an N x N grid of motorized pins and linear actuators. The elements, which are the same size as

the pins, are stacked on top of the pins (Figure 8.5 A). When stacked, the elements are connected in vertical direction, while disconnected with nearby elements in horizontal direction. Similar to existing pin-based shape displays [Follmer et al., 2013], the assembler can incrementally generate 2.5D shapes by individually moving pins to push elements to the surface.

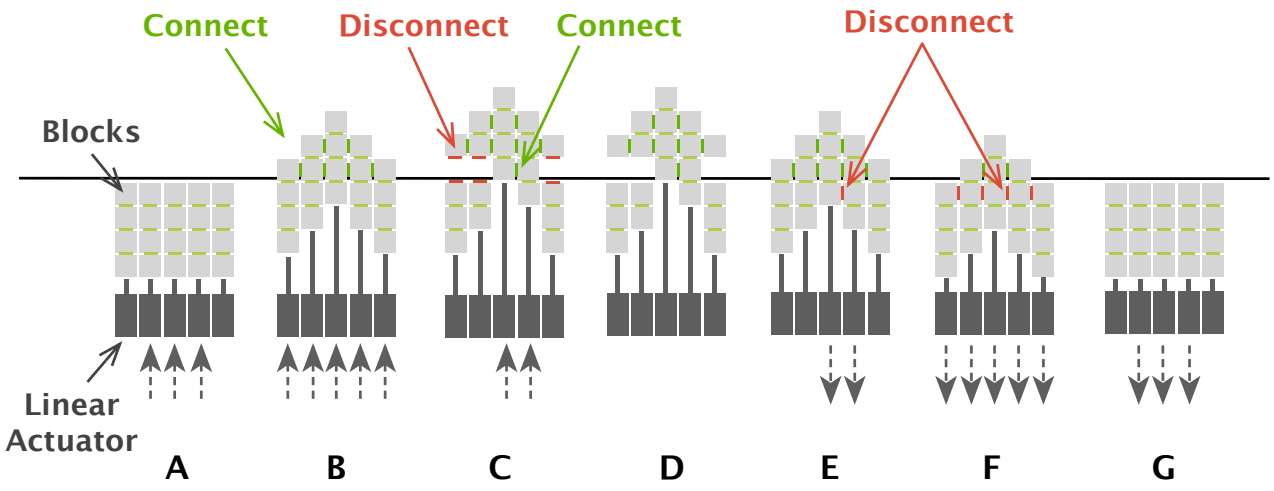


Figure 8.5: Illustration of parallel assembly using a pin-based display.

Once the elements are pushed onto the surface, each element connects to neighboring elements to form one layer of the shape. As the elements in this layer are now connected horizontally, the next layer can be formed by the same process while the previously formed layer lies on top of the next layer (Figure 8.5 B). To go from 2.5D to 3D, the elements can be disconnected in the vertical direction (Figure 8.5 C). Thus, if the pin simply pushes each layer, it can construct an overhang or inner hole structure without needing support structures. By repeating this process, the desired three-dimensional shape can be formed.

It can also disassemble a rendered shape with a similar process. By selectively moving pins down or manually pushing the object from the top, each layer will be buried to its initial position. When clearing the object, the horizontal connector is

disconnected so that each element can move down (Figure 8.5 E-F). Then, the system can reconstruct a new shape from the beginning, or possibly reconfigure an existing shape through partial clearing and reconstruction.

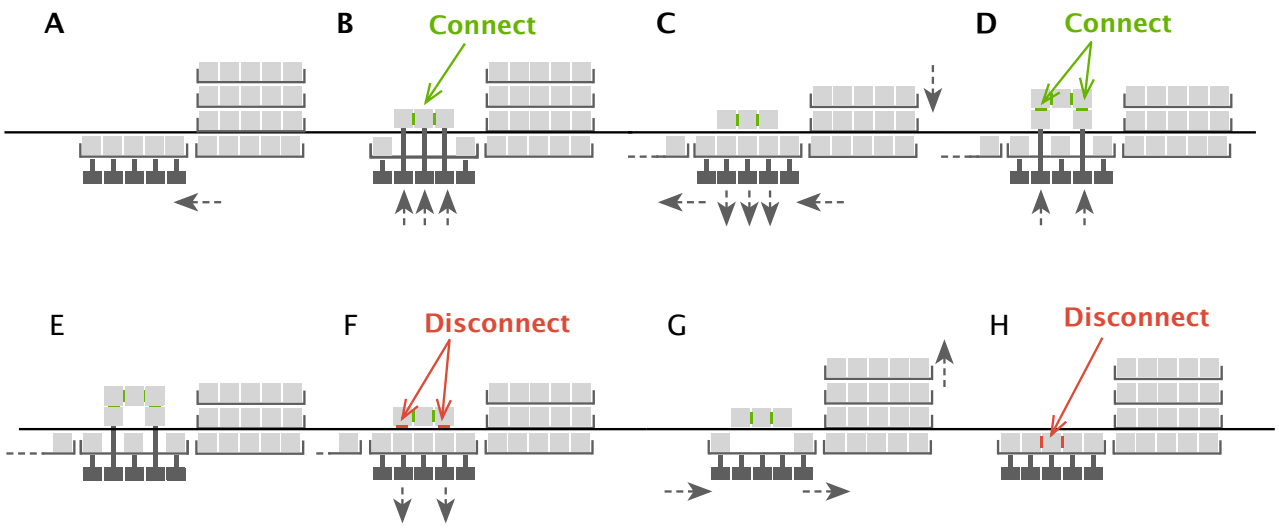


Figure 8.6: Illustration of parallel assembly using a horizontal feeder.

Another design for a parallel assembler is to use a horizontal feeder. As illustrated in Figure 8.6, the system is composed of a binary linear actuator in the vertical direction and material feeder in the horizontal direction. The system can actuate individual pins, while each pin can only go back and forth. The system prepares a stack of layers and feeds them to the build platform, then similarly assembles one layer at a time, pushing each layer up with a linear actuator. Similar to the first design, each layer is connected horizontally to prevent from falling apart while feeding the next layer. By sequentially feeding and forming each layer, the system can assemble a 3D object.

Each proposed design has both advantages and disadvantages. The design using a pin-based display enables fast and interactive rendering of 3D shapes, regardless of resolution. The construction time depends only on the speed of the motorized pins and the number of layers, assuming the connection speed

is negligible. However, it can only support elements of a single material as there is no way to change the element during production. In addition, miniaturizing shape displays is also an engineering challenge. Compared to the pin-based design, the second design is a simpler in mechanism, thus can be more easily scaled, as it requires only a binary linear actuator for vertical displacement. This design also could support multi-material elements by switching the material for each layer or computationally compound multiple elements for each feeder. On the other hand, the assembly time could increase linearly as the number of vertical layers becomes larger, because even if the connection can be switched instantly, there is still a bottleneck of the horizontal feeder having to travel the entire length of one dimension for each layer.

We prototyped both designs, but in this paper we focus on the first design with a pin-based display due to its fast assembly time.

8.3.2 Connection and Disconnection Mechanism

The next key design component is the connection and disconnection mechanism. A switchable connector is the key to allow the material elements to be reusable for reconstructable shape formation. An appropriate design and selection of the connection mechanism is important for several reasons. First, the speed of switching between connection and disconnection significantly affects the entire assembly time because the formation of each layer depends on the switching time. For example, if switching between connection and disconnection takes 10 seconds, constructing each layer takes more than 10 seconds, and therefore it would take N x 10 seconds to build N-layer objects, which is too slow for real-time interaction. Moreover, the connection mechanism would have the greatest impact on the cost and complexity of manufacturing the elements. Thus,

the connector design must be carefully considered with regard to speed and manufacturing complexity.

Type	Connection	Disconnection	Time	Manufacturing
Mechanical Latching	Push / Rotate	Pull / Rotate	1 - 10s	Simple
Permanent Magnet	None	Push / Rotate	0.1 - 1s	Simple
Electromagnet	Run Current	Turn off Current	0 - 0.1s	Complex
Electrostatic	Apply Voltage	Turn off Voltage	0 - 0.1s	Complex
Electro-permanent magnet	Run Pulse Current	Pulse Current	0 - 0.1s	Complex
Thermal bonding	Heat and Cool	Heat	1 - 30s	Simple
Photochromic bonding	Expose Visible Light	Expose UV Light / Heat	1 - 10s	Simple
Dry Adhesion	Surface Contact	Reduce the Contact Area	1 - 10s	Simple

Table 8.1: A list of switchable connectors.

A variety of switchable connectors have been proposed in the literature of modular self-reconfigurable robots. We summarize some of these approaches in Table 8.1.

Mechanical Latching

Mechanical latching is the simplest and most common way for reversible connection (e.g., LEGO blocks). While existing systems in modular robots usually achieve mechanical latching with internal motors and actuators [Nilsson, 2002], past work in digital materials has explored micro-scale mechanical latching by press fitting [Cheung and Gershenfeld, 2013]. As mechanical latching can be achieved with simple mechanical force, elements can be simple to fabricate. However, depending on the design the external assembler can be complicated and switching the connection may be slow.

Magnetic, Electromagnetic, Electrostatic Connectors

Magnetic force is another option. The simplest connection uses a permanent magnet to connect and uses external force to push or rotate the magnet to disconnect. This approach has been explored in several systems [Schoessler et al., 2015; Zhao et al., 2017]. Electromagnetic connection can be faster as it can switch

states by running current, and it can be fabricated with a standard PCB manufacturing [Pece et al., 2017; Strasnick et al., 2017; Suzuki et al., 2017]. However, one notable disadvantage of using electromagnets is power consumption: The electromagnet requires continuous current to hold the magnetic force. On the other hand, electrostatic and electro-permanent magnetic connection can maintain the connection. For example, an electro-permanent magnet can be switched to the connection state with pulse current without requiring continuous current [Knaian, 2010]. Although these connection mechanisms are appealing due to their speed and size, for millimeter scale (e.g., 1 mm [Karagozler et al., 2009] to 10 mm [Gilpin et al., 2010]), manufacturing complexity presents difficulty for large numbers of elements.

Thermal Bonding, Photochromic Bonding, and Dry Adhesion

Thermal and photochromic bonding are other reversible connection methods. These bonding mechanisms leverage phase change of materials between liquid and solid to bond elements. Similar to soldering, thermal bonding uses heating to change the phase of a material from liquid to solid, and cooling to solidify the bond. For fast phase changing, it is common to use a low-temperature melting metal such as Galium or Field's metal, which melts at 40-80C degree [Ladd et al., 2013; Neubert et al., 2014]. Thermal bonding is used in recent work on liquid metal 3D printing [Ladd et al., 2013; Wang and Liu, 2014]. Existing systems use a heater (e.g., resistive heating [Neubert et al., 2014]), but cooling the metal at room temperature takes time. An alternative phase-changing connection is photochromic bonding, which leverages UV or visible light to change the phase of materials such as azobenzene [Akiyama and Yoshida, 2012] or liquid crystal materials [Saito et al., 2016]. We also expect that reversible dry adhesion, which can connect elements with Van der Waals force, could be another

approach [Lee et al., 2007; Northen et al., 2008]. Although these methods are promising, they have not been substantially explored for connecting modular robots or digital assembly.

We prototyped 10 mm blocks with three different connectors (permanent magnet, electro-permanent magnet, and thermal bonding using Field's metal). We decided to further explore and implement a design using permanent magnetic connectors due to the simple manufacturing and faster speed of connection and disconnection of this approach.

8.4 Dynablock

To demonstrate the concept of dynamic 3D printing, we developed Dynablock. Shapes made with Dynablock consist of 9 mm blocks, which connect to neighboring blocks with embedded permanent magnets. Dynablock's hardware employs a 24 x 16 pin-based shape display as a parallel assembler. 3,072 (= 24 x 16 x 8 layers) blocks are stacked atop the shape display, and each motorized pin pushes up blocks to assemble the object layer by layer. When blocks are inside the assembler, separators keep their horizontal magnets disconnected. As the blocks are pushed upwards and out of the assembler, they connect with their neighbors magnetically to form an object. Due to weaker vertical magnetic connections, blocks can disconnect vertically during this process to form overhangs. Therefore, Dynablock can assemble arbitrary and graspable 3D shapes with overhangs, rather than the 2.5D shapes of existing shape displays [Follmer et al., 2013]. Given a shape of 3,000 elements, Dynablock drastically reduces the assembly time to seconds, which enables interactive applications that conventional 3D printing techniques do not support. Moreover, the generated objects can be disassembled into individual elements for our

system to reuse to assemble the next shape.

8.4.1 Block Design

Each element of Dynablock is a 9.4 mm 3D printed block. We chose a simple magnetic connection for easy and fast manufacturing. It is also inexpensive, which allows for scalability. Systems, like ours, that employ fixed magnets for connection must address the problem of polarity: in order to attract and connect, the two mating faces must always have opposite polarity.

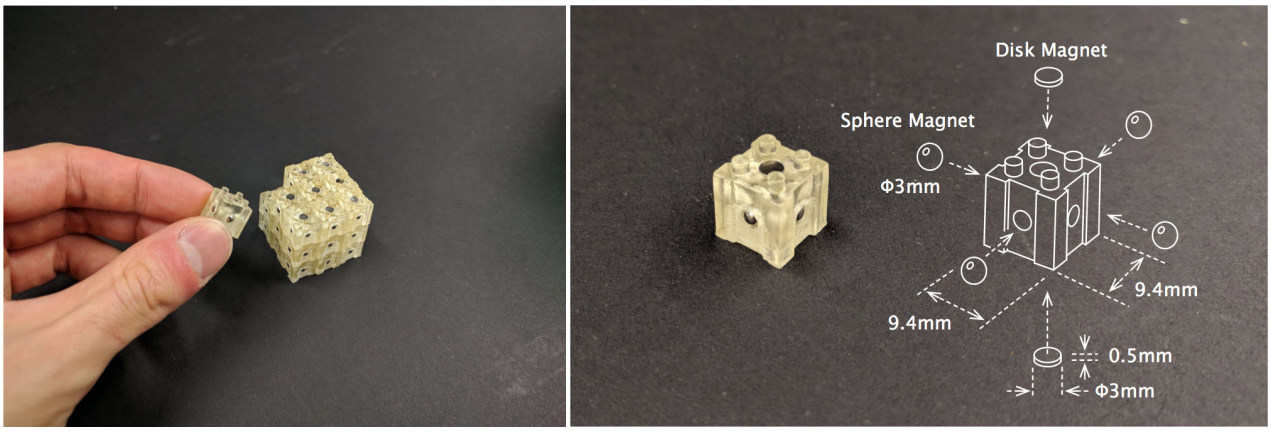


Figure 8.7: Design of a single block. Each element of Dynablock is a 9.4 mm 3D printed block. For horizontal connection, we used ϕ 3 mm sphere magnets, and for vertical connection, we used ϕ 3 mm disk magnets.

Inspired by [Schoessler et al., 2015], we use an omni-directional magnetic connector for horizontal connection to address this problem. Each horizontal face has a spherical pocket containing an N35 spherical magnet with a 3 mm diameter. The diameter of the pocket is slightly larger (3.3 mm) than the magnet inside, allowing the magnet to rotate freely within the pocket, while the (2.5 mm) hole on the face is small enough to prevent the magnet from escaping. Because each spherical magnet can rotate freely, when two faces are brought together, their two magnetic connectors can rotate and align their polarities. Each horizontal face also has a slit with 0.5 mm in depth and 4 mm in width, which receives a spacer to separate blocks during assembly. To support vertical connection, we embed a thin (ϕ3

mm x 0.5 mm thickness) disc-shaped magnet in both top and bottom faces. Each block has four studs (ϕ1 mm x 1 mm thickness) on the top and mating cylindrical holes (ϕ1.4 mm x 1.2 mm thickness) on the bottom. These studs prevent horizontal rotation between vertically stacked blocks.

8.4.2 Mechanism for Horizontal Connection and Disconnection

Next, we describe the mechanism for connection and disconnection using the shape-display. The block elements of Dynablock are stacked on top of the pin arrays. As described in the design section, each pin can push the vertically connected stacked blocks.

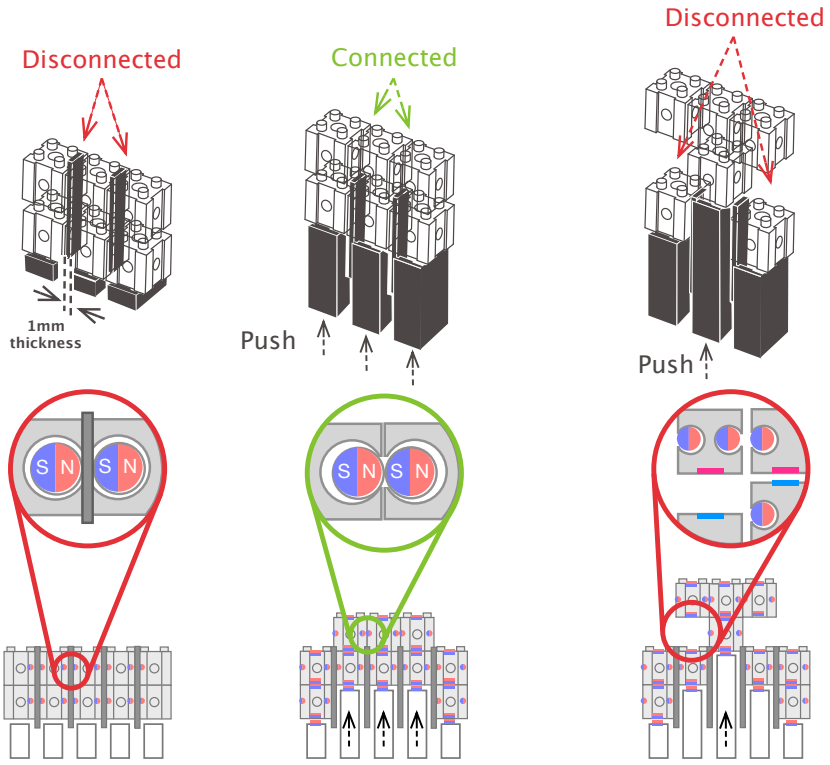


Figure 8.8: Horizontal connection is achieved by a pair of rotatable sphere magnets, and vertical disconnection is achieved by different attraction force of the weaker magnet.

Figure 8.8 illustrates the mechanical design. As described above, each block has a 0.5 mm deep slit, which receives a 1 mm thick spacer attached to the bottom of the plate that serves

as an obstacle to horizontal connection. Although there is still magnetic attraction between blocks, it is too weak to connect the blocks. This horizontal separation mechanism allows each pin to individually push the stacked blocks without interfering with nearby stacks.

For stable connection and disconnection, a careful design must be considered. For example, if the spacer and slit are too thin, then the distance between two magnets could be too short to maintain strong magnetic attraction. On the other hand, if the spacer and slit are too thick, then it can be difficult to attract and connect with nearby magnets once the spacer is removed. We found that the 1 mm thick spacer and 0.5 mm thick slit are thick enough to reduce the magnetic attraction, while thin enough to allow stable connection when the spacer is removed.

Figure 8.9 shows a photo of the system viewed from the side, along with an illustration of the horizontal magnet connections. In the stacked state, all five magnets on the top layer are disconnected due to the white 1 mm spacer. Therefore, an actuated pin can individually push up the blocks on top of it without affecting neighboring blocks. The photo in figure 8.9 depicts the state after the three center blocks are pushed up with the pin. At the top of the block holder, there is no longer a spacer to prevent the magnets from forming a connection, so the three blocks connect horizontally. Note that the separating spacer only exists at the center to fit within each slit, so that the blocks can be densely packed within the block holder and don't shift when pushed up.

8.4.3 Mechanism for Vertical Connection and Disconnection

By default, the blocks connect vertically when stacked in the block holder. Thus, to create an overhang or internal hole

structure, blocks must be disconnected in the vertical dimension.

For vertical disconnection, we take advantage of the difference in the attraction force between slightly different magnets on horizontal and vertical surfaces. We use N35 3 mm sphere magnets for horizontal connection, and N45 3 mm with 0.5 mm thick disc magnets for vertical connection. The attraction force of the vertical connections is smaller than that of the horizontal connections. The pull force and surface field of the vertical connection (N45 3 mm x 0.5 mm disk magnet) are 0.10 kgf and 1,650 gauss, while the horizontal connection (N35 3mm sphere magnet) achieves 0.16 kgf and 8,060 gauss respectively. This allows the vertically connected blocks to be detached while maintaining the horizontal connection.

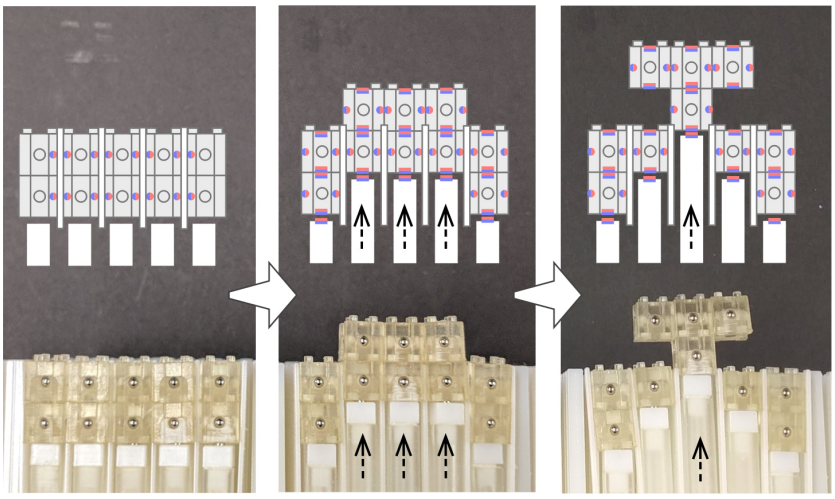


Figure 8.9: Side view of Dynablock: The parallel assembler creates a shape with an overhang from magnetically connected blocks.

Figure 8.8 shows a side view of this mechanism. On the left, the two top blocks are horizontally connected while also vertically connected to the blocks in the layer below. When pushing up only one block, the horizontal connection is stronger than the vertical connection. This makes it possible to create an overhanging structure without needing a support structure.

Note that the vertical disconnection is only invoked to detach blocks from the underlying layer, but in an assembled object the vertical connection between blocks is maintained. This allows the rendered object to be stable without breaking when grasped.

To disassemble the rendered object, the system drops each pin to its initial position. Then, the object is pushed down into the block holder, breaking the horizontal connections between the blocks. Once the blocks are in their initial positions, the system can reconstruct and render a new shape.

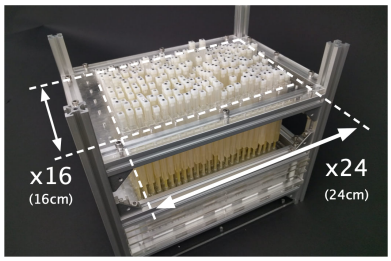


Figure 8.10: Dynablock's parallel assembler implemented using a shape display.

8.4.4 Shape Display as a Parallel Assembler

For the parallel assembler, we built a pin-based shape display. Taking inspiration from FEELEX [Iwata et al., 2001] and shapeShift [Siu et al., 2018], we used a geared DC motor and a motorized lead screw for linear actuation. The blocks are held within the 3D printed block holder, as shown in Figure 8.11. In each cell, the block holder has a 4 mm wide, 1 mm thick spacer to separate neighboring blocks. This spacer prevents them from connecting horizontally.

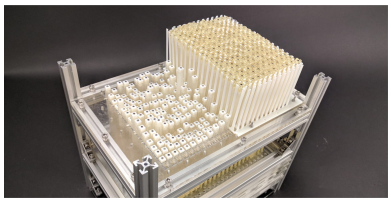


Figure 8.11: The shape display with the block holder on the right half.

The assembler consists of a 24 x 16 array of motor-driven pins. Each pin moves up and down, driven by a small DC motor (TTMotors TGPP06-D700) and a 3D printed lead screw (2 mm pitch, 4 starts, 120 mm in length). TGPP06-D700 is 6 mm in diameter and 29 mm in length and can rotate 47 rpm with 1:700 gear ratio. The 2 mm 4 starts lead screw can travel 12 mm per second without load, and each motor consumes approximately 60 mA. The pins are 3D printed with a nut at the bottom to travel along the lead screw. Each pin is 120 mm long and has a 7mm square cross section with a 5 mm diameter hole from top to bottom, and an N45 disk magnet (ϕ 3mm x 2.4 mm thickness) is attached at the top. Guide grids at the top prevent pins

from rotating and ensure that pins travel vertically.

The 24 x 16 guide grids have 7.5 mm square holes with 10.16 mm pitch and are cut from a 5 mm acrylic plate. We fabricated the pins, the lead screws, and blocks with an inkjet 3D printer (Keyence Agilista 3200) with water soluble support material. In total, we fabricated 384 (= 24 x 16) pins and lead screws, and 3,072 (= 24 x 16 x 8 layers) blocks. To create the magnetic blocks, we embedded spherical magnets in each block by hand and inserted disk magnets using a bench vice.

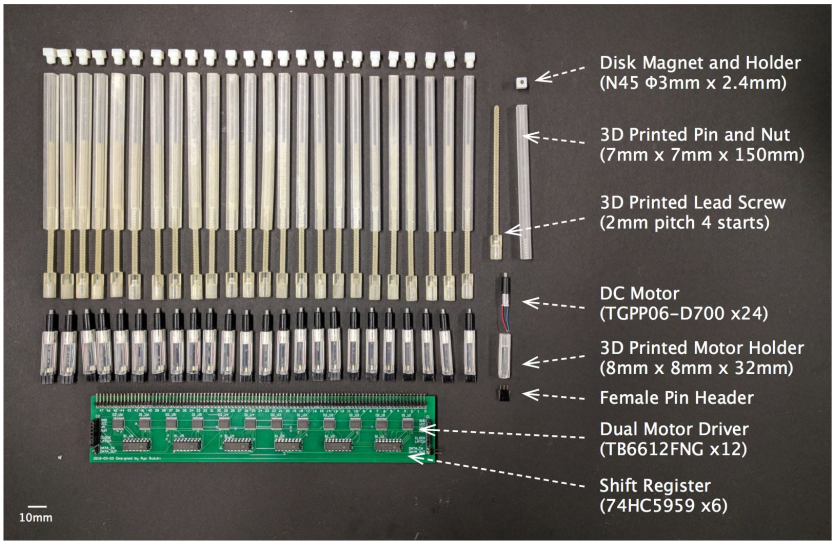


Figure 8.12: Components of our implementation of Dynablock's parallel assembler.

In order to connect motors to a printed circuit board (PCB), we designed and fabricated a custom motor holder. A 3D printed motor holder is attached to each DC motor, soldered with a 2.54 mm pitch female pin header at the bottom for simple fabrication and assembly. Each motor holder is fixed with a 3mm acrylic plate with an 8mm hole. The motor holder has clutches to fix the motor's position and prevent it from rotating. The motor is connected to a printed circuit board and aligned vertically through an L-shaped male pin header, aligned horizontally with a 10.16 mm (= 2.54 mm x 4) pitch. Motor holders are fabricated with the Form 2 3D printers using standard resin.

Each controller PCB comprises twelve dual motor drivers (TB6612FNG) and six shift registers (74HC5959). Each motor driver can switch the direction of two motors using an H-bridge and each shift register can individually control three of the dual motor drivers (i.e., six motors). Thus, for each row, one PCB can independently control the direction of 24 motors. The speed of each motor can be also controlled through pulse-width modulation (PWM). The shift register is controlled through a daisy-chained serial-in parallel-out signal, sharing the latch and clock among all PCBs. Thus, all the shift registers are controlled with three digital pins from an Arduino Uno. The VCC of the controller is connected to 5V and the external voltage for DC motors is connected to a DC 5V power supply . These low-cost components support scalable production (in our prototype, a DC motor, a motor driver, and a shift register costs US $3.10, $0.90, and $0.30 respectively).

8.4.5 Software

We built an interactive voxel editor and simulator (Figure 8.13). The voxel editor allows the interactive design of objects that can be formed by the Dynablock. The user imports an 3D design as an STL file which our software converts to voxels at an appropriate resolution.

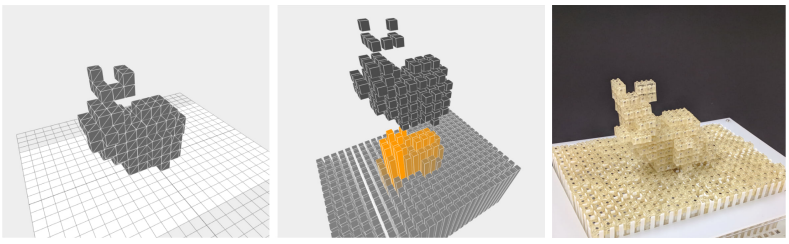


Figure 8.13: Interactive voxel editor and simulation software.

The user also can interactively edit the shape of imported object or create a shape from scratch. Once the user confirms the shape to be rendered, then the simulator shows and tracks the current position of the array of 24 x 16 actuated pins.

The user interface of the software is built using Three.js and React.js, and communicates with the Arduino controller through a Node.js server. The Node.js server logs commands sent to the Arduino, such as the running time and the direction of each motor. The log is stored as JSON data and tracks the duration of pushing each pin, and based on these data we can compute the pin positions. The Node.js server feeds the information to the front-end to simulate and visualize the current positions of the pins in a browser through websocket. To track the user interaction and vertical displacement, we mounted a Kinect depth camera above the system. Figure 8.14 illustrates the entire architecture of software and hardware of Dynablock.

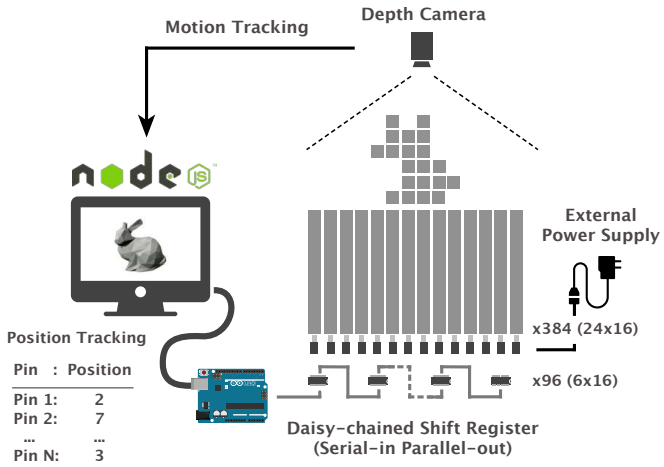


Figure 8.14: System architecture of our dynamic 3D printing prototype.

8.5 Application Scenarios

In this section, we illustrate several possible application scenarios with dynamic 3D printers.

8.5.1 Direct Interactive Fabrication

Dynamic 3D printing would enable a new design workflow for digital fabrication. One notable advantage of dynamic 3D printing is the capability of connecting and disconnecting

building blocks through direct manipulation. The user can also define variables or abstract attributes for parametric design through direct and gestural interaction [Leithinger et al., 2011; Suzuki et al., 2018a]. By leveraging this capability, the user could interactively design and fabricate in a physical space, similar to the man-machine dialogue proposed by Frazer et al. and later tangible CAD interfaces [De Araùjo et al., 2012; Frazer, 1995].

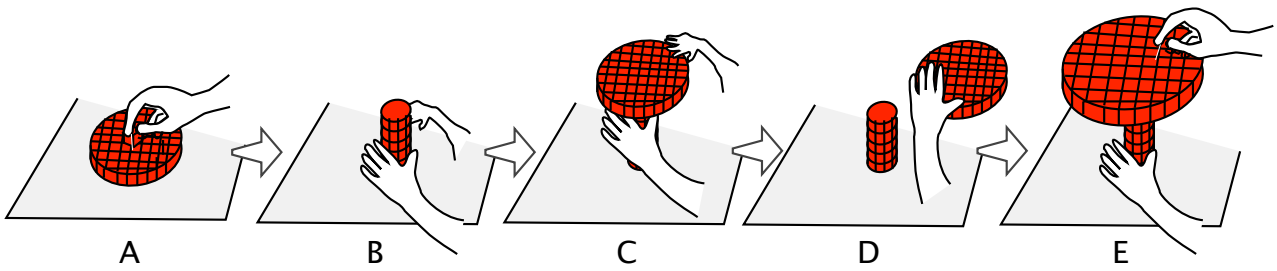


Figure 8.15: Application in direct interactive fabrication.

For example, when designing a table, a user can first create a large disk for the table top (Figure 8.15 A), then create a cylinder for the table's leg (Figure 8.15 B). The user can easily attach the table top and the leg (Figure 8.15 C). If the table top is too small, the user can detach and disassemble it (Figure 8.15 D), then design and generate a larger version (Figure 8.15 E). This new workflow can introduce direct touch interaction to the current interactive fabrication workflow [Willis et al., 2011c]. Moreover, similar to faBrickation [Mueller et al., 2014], the user can attach 3D printed parts into the object rendered by Dynablock to achieve a higher resolution or embed different materials.

8.5.2 Dynamic Physicalizable Textbook

Today, an augmented reality system can show an interactive 3D image for a textbook. For example, a natural history textbook can show an animated Brontosaurus, a chemistry textbook can show a water molecule, and an architecture textbook can show

Mies van der Rohe's Barcelona Pavilion. What if a textbook could generate physical 3D shapes with which a student can interactively explore a concept?

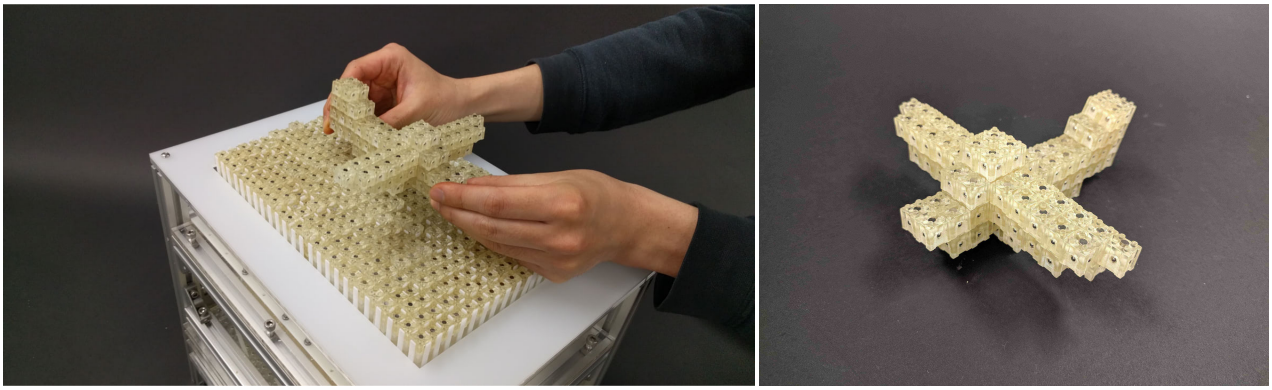


Figure 8.16: Airplane model rendered with dynamic physicalizable text- book.

For example, imagine a student reading a design textbook explaining the aerodynamics of an airplane. The student pushes a "render" button, and the book immediately generates a physical aircraft model. The actual physical object can leverage direct manipulation (Figure 8.16). For example, the system can synchronize a projector and a depth camera to visualize a computational fluid dynamics simulation and project the air flow around the model. In contrast to using only visual information, a student can explore complex ideas by directly manipulating tangible objects. We can also leverage a Hall effect sensor array like GaussSense [Liang et al., 2012] to detect the position and the orientation of the assembled object on a flat surface. This way, the physical object can be used as a tangible controller for various education and design applications.

8.5.3 On-demand Haptic Proxy Objects for VR

While virtual reality is emerging in various applications, a key research topic is how to provide rich and high resolution haptic feedback synchronized with visual information [Siu et al., 2018; Zhao et al., 2017]. For example, when playing a game in virtual reality, a controller may be used to represent

different objects depending on the situation: a steering wheel, a fishing rod, or a guitar. In such cases, dynamic 3D printing could create on-demand haptic proxy objects for VR. Inspired by the the robotic assembly of haptic proxy objects [Zhao et al., 2017], we envision that a dynamic 3D printer enables a user to immediately create a custom physical object and use it as a haptic proxy for the visual image presented in the head-mounted display. While this object would not dynamically change its shape in real-time, the user could form different physical controllers to match the visual appearance.

8.6 Discussion

8.6.1 Resolution

While our current implementation uses 9 mm blocks, the size of blocks depends on the size of embedded magnets. Our first prototype uses 3 mm sphere magnets, but the block size can be reduced by using smaller magnets. For example, we prototyped 3 mm size blocks with commercially available N50 1mm sphere neodymium magnets. However, to reduce the element size, the parallel assembler must also be scaled down, which requires overcoming the engineering problem to miniaturize the assembler's various electrical and mechanical parts. The geared motor used in our prototype has a 6 mm diameter, so in our current design, the pitch between two blocks cannot be smaller than 6 mm. To further reduce the pitch size, we might use push-pull flexible linkages, similar to the inFORM system [Follmer et al., 2013]. Alternatively, prior work suggests that it is possible to implement a more closely packed, higher resolution pin-based display using a fusible alloy clutch array [Peters, 2011]. Thus, we believe our design of Dynablock can be scaled to a 3 mm block resolution without fundamental design changes.

8.6.2 Speed

While our current prototype can form a shape in seconds, the shape has at most only eight layers. However, if the element size becomes smaller, the required number of layers would also increase. Then, does the assembly time also increase linearly with the resolution? We note that the assembly time would only depend on the travel speed of actuated pins (i.e., how long it takes to go up to eight layers), assuming the connection and disconnection time is negligible. For example, if the time to push each layer is longer than the time to switch the connection, then the completion time of one layer depends solely on the actuation speed. Assuming the assembly time of one layer is constant and independent of the horizontal resolution, we expect the speed of formation in dynamic 3D printing would be fast enough even in higher resolution. Regarding the actuation speed, faster motors exist with the same 6 mm diameter (1:24, 1:136). Although using these fast geared motors may decrease the compilation time to a speed comparable to current shape displays, there is a trade-off between travel speed and torque as the faster-geared motors may be too weak to lift the magnetic blocks. Thus, faster motors may require a different connection method or design.

8.6.3 Connector

As we discussed, the connection mechanism plays an important role in assembly speed. If it takes a long time for elements to connect or disconnect with neighbors, the assembly will be too slow. In this prototype, we chose permanent magnets and mechanical disconnection for simplicity and ease of construction. While we note this method works at the millimeter scale, further reduction in size to sub-millimeter or even micrometer scale may require a different connection mechanism. There are several promising alternatives. For example, one possible

connection mechanism is dry adhesion leveraging the Van der Waals force in a hair or gecko-like structure. Another possible solution is to use MEMS or ASIC based electro-permanent magnets. We are interested in exploring these alternative connection methods.

9

Dynamic Space Construction enabled by Swarm Robotic Actuation

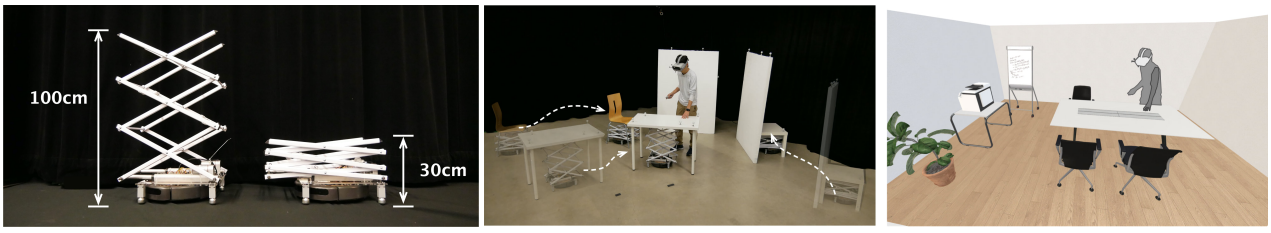


Figure 9.1: RoomShift exemplifies dynamic space construction and reconfiguration with collective actuation enabled by furniture-moving swarm robots [Suzuki et al., 2020a].

9.1 Overview

Dynamic shape construction with collective elements is not only for construction of an object at a scale of human hands or bodies. But, also a space or an environment also consists of collective elements, such as tools, objects, furniture, and built environments. At different scale, the shape constructed by collective elements also becomes an element for larger scale. For example, an small piece of block can be an element of an object

on top of a table. The object on top of the table is an element that consists the space on the table. The table is an element that consists a room. The room is an element that consists a house. The house is an element that consists a city. In this way, based on the viewpoint, we can think of collective elements at different scale. As we discussed in the Background section, the space and the environment also play an important role in user interaction. Therefore, it is important to consider how we can also dynamically construct and reconfigure a space for human-computer interaction.

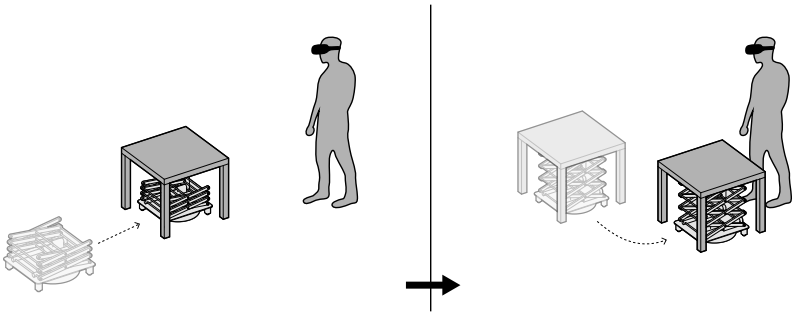


Figure 9.2: RoomShift robots move beneath a piece of furniture to lift, move and place it.

In this chapter, I focus on collective elements at room-scale, and how we can dynamically construct and reconfigure the room, by actuating the elements that consists the room. At this scale, the elements become objects, furniture, and built environments such as wall or table. I will look into how we can actuate these elements more dynamically.

To this end, I explore the dynamic spatial reconstruction with furniture-moving swarm robots, called RoomShift [1]. RoomShift consists of nine shape-changing robots based on Roombas with mechanical scissor lifts. These robots drive beneath a piece of furniture to lift, move and place it. RoomShift exemplifies **collective assistant through ambient spatial reconfiguration** — collectively moving existing furniture to assist everyday tasks as well as providing haptic feedback for virtual reality.

[1] Suzuki, R., Hedayati, H., Bohn, J., Zheng, C., Do, E. Y.-L., Szafir, D., and Leithinger, D. (2020a). Roomshift: Room-scale dynamic haptics for vr with furniture-moving swarm robots. In *Proceedings of the CHI '20*, pages 199:1–199:13

9.2 RoomShift

RoomShift is a small swarm of furniture-moving swarm robots. Inspired by shelf-moving warehouse robots (e.g., Kiva robots [2]), we develop a swarm of shape-changing robots that can move a range of existing furniture. Each robot has a mechanical lift that can extend from 30 cm to 100 cm to pick up, carry, and place objects such as chairs, tables, and walls.

[2] Wurman, P. R., D'Andrea, R., and Mountz, M. (2008). Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI magazine*, 29(1):9–9

9.2.1 Mechanical Design

RoomShift comprises nine shape-changing swarm robots based on the Roomba Create 2 [Dekan et al., 2013]. For the mechanical lift structure, we repurposed an off-the-shelf expandable laundry rack (Room Essentials Compact Drying Rack) and attached two linear actuators (Homend DC12V 8 inch Stroke Linear Actuator, which extends from 32 cm to 52 cm) at the base of the rack. The linear actuators are fixed to the endpoints of the scissor structure with 8 mm steel rods, so that when the actuator contracts, the mounted scissor structure extends vertically (from 30 cm to 100 cm).

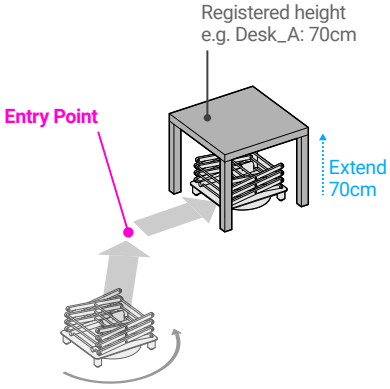


Figure 9.3: The robot first goes to an entry point to move underneath the furniture.

The scissor structure moves at a speed of 1.3 cm / sec. To mount the scissor structure, we fixed a 6mm acrylic bottom plate (35 cm x 35 cm) and four omni-directional casters (Dorhea Ball Transfer Bearing Unit) to relieve the Roomba of most of the weight that the robot carries. Each robot moves at 20 cm / sec. Figure 9.5 illustrates the mechanical design of each RoomShift robot.

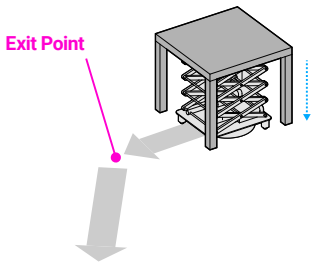


Figure 9.4: When the robot leaves, the system navigates the robot to the entry point to avoid the collision with the legs of furniture.

We considered and tested several actuation mechanisms such as a pneumatically-actuated inflatable structure [Hammond et al., 2017; Suzuki et al., 2020b; Teng et al., 2019], a deployable structure using coilable masts [Jensen and Pellegrino, 2001; Joosten, 2007], and a mechanical reel-based actuation [Takei

et al., 2011]. Pneumatic actuation is problematic for our mobile setup as it requires a tube connected to a pump or pressure tank to supply air. The deployable structure and mechanical reel-based actuation affords a much higher extension ratio, but is limited in its robustness and load-bearing capability.

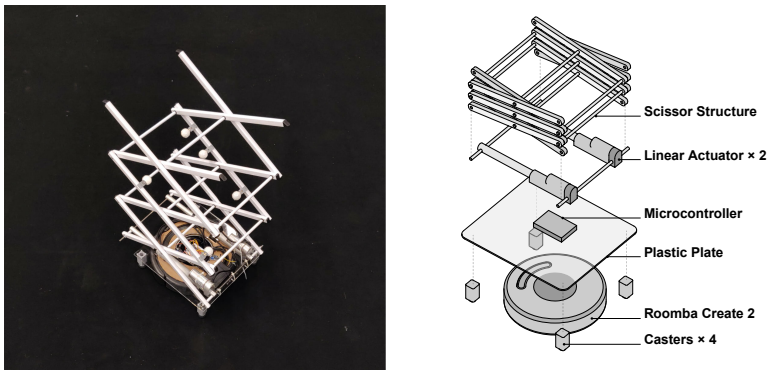


Figure 9.5: Mechanical design of the robot and the scissor structure.

The mechanical scissor structure is appropriate for our purpose because it is inexpensive (compact drying rack: $ 15, linear actuators: $ 35 x 2) and lightweight (2kg). Existing warehouse robots such as Kiva [Guizzo, 2008; Wurman et al., 2008] have a limited expandable capability as they are designed for one specific shelf, whereas our mechanical scissor lift can move various objects by leveraging its highly expandable structure (4 times expansion ratio). The current actuator height (30 - 100 cm) was chosen to cover a wide range of standard chairs and tables, which measure 30 - 76cm and 48 - 96 cm respectively [3]. The maximum height of the scissor structure itself can be also extended by adding more elements like combining two scissor structures to double the maximum height, with the trade-off with the less stable structure.

[3] Woodworking, H. (2019). Standard furniture dimensions

9.2.2 Object Actuation

The scissor structure is robust and can withstand up to 20 kg. Thus, our robots can lift and carry heavier objects than an unmodified Roomba. One advantage of the RoomShift approach

is that the robot itself does not need to support the human weight, for example when a user sits on a chair; once the robot places the chair, it serves as static furniture. This significantly reduces the possibility of a mechanical breakdown.

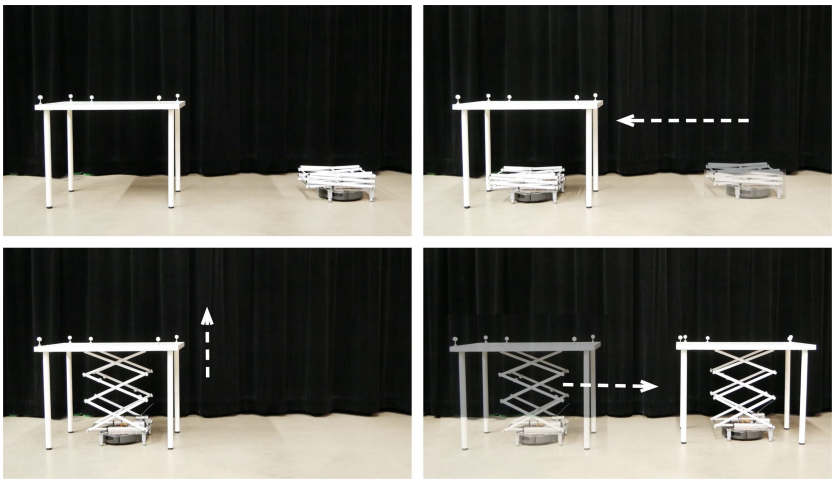


Figure 9.6: A RoomShift robot drives underneath a desk, lifts it by extending the scissor structure, and moves it.

Although the maximum load for the Roomba is 9 kg, the corner-mounted casters distribute and carry heavier loads. Thus, our robots can lift and carry heavier objects than an unmodified Roomba. The maximum weight the robot can lift and carry is 22 kg. When we put a heavier object than 23 kg, we observed the scissor structure started to break. The strength of the scissor structure suffices to lift lightweight chairs and tables, such as the IKEA honeycomb furniture used in our prototypes.

The weight of the furniture we have tested (depicted in Figure 9.7) ranges from 3.5 to 11.2 kg. For heavier objects, multiple robots can also coordinate to lift a piece together if there is sufficient space under the furniture. Also, with a more robust scissor structure, we can carry heavier objects, as we observed the Roomba base itself (with the corner-mounted casters) can carry up to 30 kg load.

This approach also increases flexibility because different types

of furniture can be actuated with the height-adjustable scissor lift. For example, Figure 9.7 illustrates various static props that the RoomShift robot can actuate. These objects include furniture such as a desk, a long table, different chairs, and a side table. Note that due to the robot's minimum collapsed size, objects must have at least 30 cm clearance below them, and enough horizontal space to fit the robot.

Figure 9.7: Different types of furniture moved by the system, including various standard chairs, desks, racks, and tables. A designer can also create custom props for specific applications, for instance, the styrofoam wall mounted to a side table seen in the corner.

9.2.3 Tracking System

To accurately control the RoomShift robot, we require precise motion tracking that can cover the play area in which a user walks. We use an optical tracking system with twenty IR cameras that can track objects in a 10 m x 10 m space.

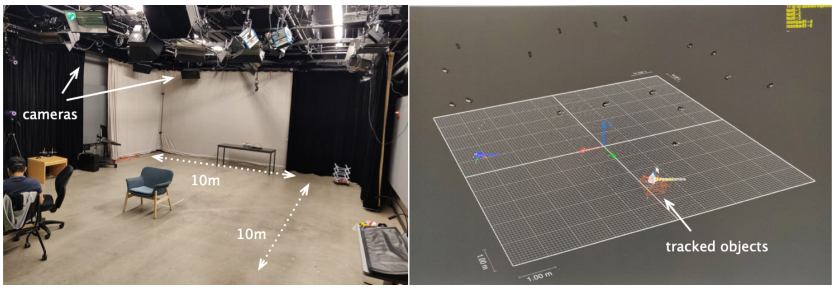


Figure 9.8: Photo of tracked space and screenshot of tracking software.

Figure 9.8 depicts the space and mounted cameras on the ceiling and tracking software. The system tracks six degrees of freedom (DOF) position of the objects with retro-reflective

spherical markers at 60 FPS frame rate. To track each robot, we attached five 30 mm spherical retro-reflective markers to the bars of the scissor structure (Figure 9.9). We attached markers to a pair of parallel bars, so that the markers' relative positions can maintain regardless of the height of the scissor lift. Through the height orientation of the marker, we also estimate the height of the scissor structure.

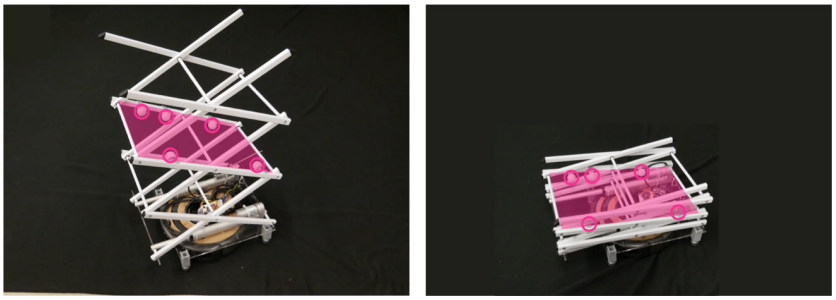


Figure 9.9: Retro-reflective markers mounted to parallel lift bars, high- lighted in pink.

All physical props in the system have retroreflective markers attached to capture and track their positions and orientations, and plan the paths for the robots to pick them up and avoid collisions. They also enable the system to track the robots while moving objects: when a robot markers are hidden beneath an object it is carrying, the system can still reliably track the robot using the object markers as a proxy for the hidden robot.

9.2.4 System Design

Each robot is controlled with ESP8266 microcontroller chip, powered by the Roomba through a voltage regulator. An additional external battery powers the linear actuators for the scissor structure.

The microcontroller receives commands over WiFi and controls the left and right wheels of the Roomba using a PWM signal. The microcontroller also operates two linear actuators using a dual motor driver (TB6612FNG). The Roomba's internal bat-

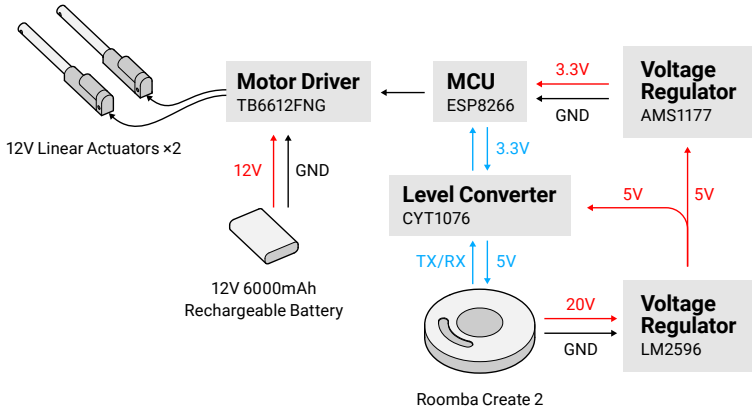


Figure 9.10: Hardware schematic of the robot. The power source of the microcontroller is the Roomba's internal battery which supplies 14-20V. The logic level converter converts the voltage for serial communication between the microcontroller (3.3V) and Roomba (5V).

tery is insufficient to supply the current for the linear actuators (600-800mA for average, 1.5-2A for peak current), so we use an external portable rechargeable battery (12V 6000mAh) to power the actuators.

To control the robot, we use a simple path planning algorithm. The input of our path planning algorithm is 1) the current positions of the robots, 2) the positions of obstacles (e.g., furniture and users), and 3) the target locations at time t. Given these inputs, the algorithm outputs the goal of each robot at the next time step (t + δt). The system continuously moves the robots until they reach their target locations. The main server continuously tracks the robot positions, calculates their wheel speed with PID control and sends commands at 30 Hz over WiFi.

9.3 Application Scenarios

9.3.1 Dynamic Haptic Environments for VR

One of the promising application areas is to provide haptics for virtual reality. Through a dynamic haptic environment, users can not only see, but also touch and interact with the whole virtual scene with their bodies — they can walk, sit on, and lean against objects in the VR environment. RoomShift enables

a room-scale dynamic haptic environment for virtual reality by providing haptic sensations through reconfigurable physical environments.

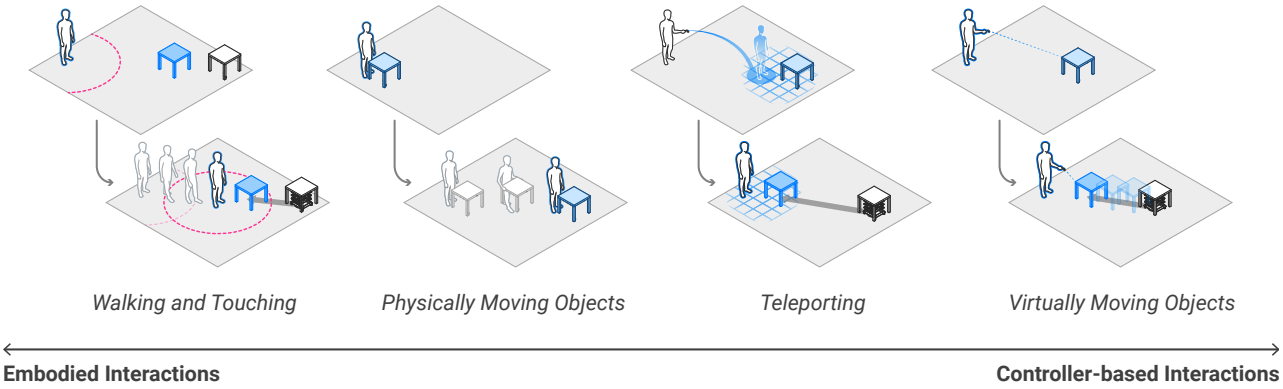


Figure 9.11: The interaction design space of RoomShift to provide haptics for VR.

Particularly, architectural application scenarios are interesting — such as rendering physical room interiors for virtual real estate tours and collaborative architectural design, two increasingly common application areas for VR [4]. Virtual real estate tours reduce the time and cost compared to on-site viewings, but currently lack the bodily experience of being able to touch surfaces and sit down. In architectural design, VR aids the communication between architects and clients, where proposed designs can be experienced, discussed and modified before building them.

[4] Ibayashi, H., Sugiura, Y., Sakamoto, D., Miyata, N., Tada, M., Okuma, T., Kurata, T., Mochimaru, M., and Igarashi, T. (2015). Dollhouse vr: a multi-view, multi-user collaborative design workspace with vr technology. In *SIGGRAPH Asia 2015 Emerging Technologies*, page 8. ACM

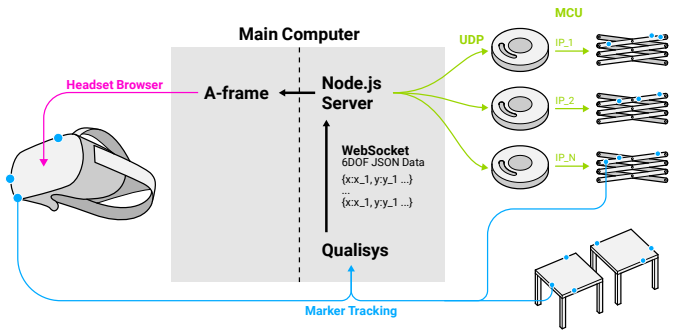


Figure 9.12: Software system to support VR applications. Based on the tracking data, a web browser client renders the VR scene with A-Frame. When the virtual scene changes, the system moves the robots to dynamically reconfigure the physical scene.

We are motivated by how RoomShift can enable people with various physical abilities to experience, test and co-design

these environments with their bodies. Most of the elements in these applications can be covered with a finite set of furniture and props (e.g., chairs, desks, and walls). We discuss some of the basic interactions to support these applications.

Experiencing Architectural Spaces: Walking and Touching

The most basic interaction is to render an architectural space that the user can walk around in and touch. To render the haptic proxies for a large space would require a large number of physical props and robots. On the other hand, the user's immediate physical reach is usually smaller than the entire virtual scene (e.g., 1.5 m radius). Therefore, the system only places haptic props within the user's immediate proximity. As the user walks around the space, the robots move the props to maintain the illusion of a larger number of objects. In this way, a small number of robots with a finite set of physical props can suffice to provide haptics for the scene as the system does not need to physically render the entire environment.

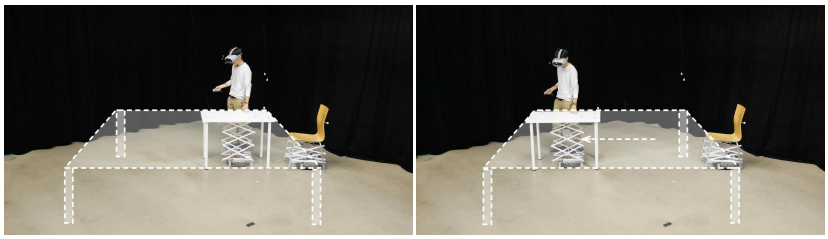


Figure 9.13: Simulating a larger table by moving a smaller surface.

In addition, the system can mimic larger objects with a single moving robot. For example, when the user is interacting with a large table, either new physical table segments can be added or a single robot can continually move the current table according to the user's position to simulate touching a larger one. This way, a limited number of robots and furniture can simulate large objects (Figure 9.13). We also employ this technique for rendering larger wall segments, where the robot moves, carrying the proxy, as the user walks along the wall, similar to a

technique proposed in PhyShare [He et al., 2017b].

Navigating Large Spaces: Teleporting in VR

The physical play area of a VR setup is often much smaller than the virtual scene. Teleportation is a common navigation technique that enables the user to point with a controller to a distant location in the scene and instantly move there [Langbehn et al., 2018]. RoomShift supports teleportation by reconfiguring the room layout to match the new view location (Figure 9.14). When the user teleports to a new location in the VR scene, the system calculates the positions of the virtual objects relative to the new location and moves the furniture and robots in and out of the play area to enable a fast scene reconfiguration and to avoid collisions with the user and each other.

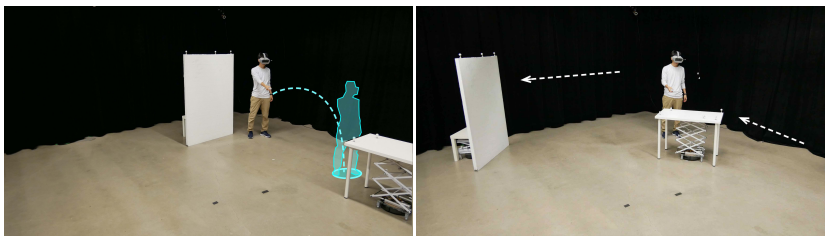


Figure 9.14: When teleporting, the robots move furniture to match the new scene position.

Architectural Co-Design: Physically Moving Furniture

VR can support teams of architects, designers and their clients to experience and discuss architectural and interior designs. For example, Dollhouse VR [Ibayashi et al., 2015] proposes such a possibility for the collaborative design of the home and office spaces, where a user experiences space in VR, while a designer views the layout on a desktop computer remotely and changes the design during the discussion. RoomShift system improves the immersion of this collaborative design process by enabling whole-body interactions with furniture. Suppose a situation where a designer and a client are remotely co-designing a new office space at two separated RoomShift

systems. When the designer reconfigures the furniture or space, the robots in the remote location can move the physical objects in real-time to render the designer's change. In this way, two remote physical environments can synchronize with a single virtual space. This aids co-design where the client can touch, feel, walk around, and modify the design in VR.

Virtual Scene Editing: Virtually Moving Furniture

RoomShift system also supports scene editing within VR. The virtual scene layout editing is similar to standard VR interactions and includes functionality like adding, removing, moving, resizing, rotating virtual building elements and furniture with a VR controller or a GUI. For example, the user can point the controller at a virtual object and move it to a target location. The robot then updates the virtual object position (Figure 9.15).

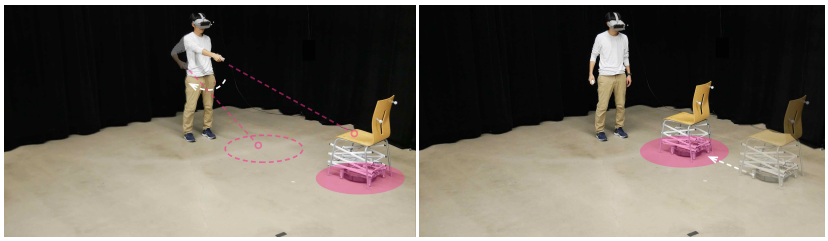


Figure 9.15: Pointing and moving with a gesture.

9.3.2 Ambient Space Reconfiguration

Beyond the VR haptic applications, RoomShift also has the potential for broader application space in ambient assistants. These distributed robots can help the automation of home, labs, store, and public space by automatically reconfigure the spatial elements based on the situation.

For example, these robots can set up the meeting space, desk, and chair based on the calendar event, and clean up and reconfigure the space after the meeting. RoomShift's capability

of actuating existing objects are particularly interesting for this application space, as the users do not need to modify the existing furniture, instead, they can just deploy these robots in space.

Part III

Interaction with Collective Elements

In the previous parts, I have mostly focused on *output* aspect of the dynamic physical interfaces — an aspect of how to make dynamic physical shape with both collective active and passive elements. In this part, I will explore **interaction design** aspect — an aspect of how the user can interact and program the collective elements.

More specifically, I will investigate an interaction technique for programming the behavior of the collective elements. In the following chapter, I will extend the notion of programming by demonstration for the spatially distributed collective elements, and how the user can program the dynamic behavior of them through direct physical manipulation.

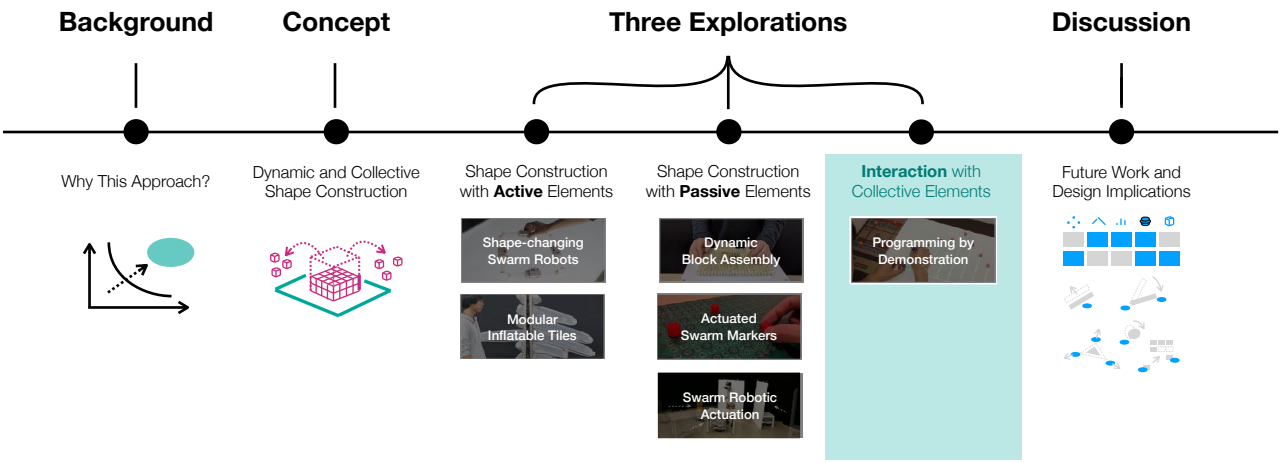


Figure 9.16: Part III: Interaction with collective elements.

10

Programming Dynamic Shape through Direct Manipulation

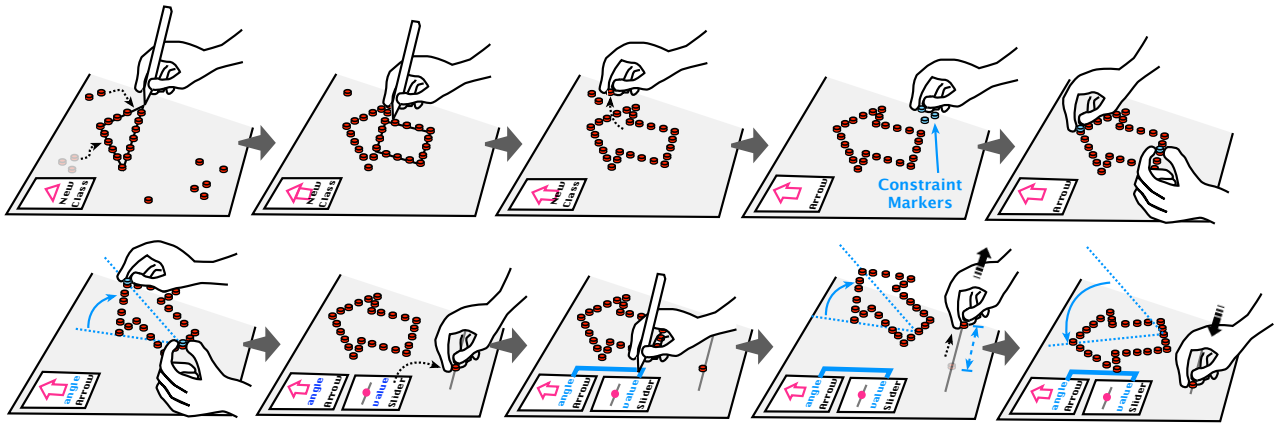


Figure 10.1: A programming environment for swarm user interfaces. The proposed workflow leverages physical demonstration for attribute abstraction and specification of data binding in Swarm UIs [Suzuki et al., 2018a].

10.1 Overview

There are also many aspects in the interaction design with collective elements. For example, what kind of modalities — such as voice, gesture, or direct touch — we should use to in-

teract with these elements, how the user interface elements — such as buttons, sliders, and a pointer — should look like with these collective elements, how can the users manipulate more number of elements than they can grab with their hands, what would be the interaction vocabulary for dynamic physical affordances [Follmer et al., 2013; Follmer, 2015] with discrete collective elements, how we should combine the overlaid graphics and physical display to complement each other.

Among these various aspects, this thesis focuses on the interaction for *programming* these collective elements. More specifically, in this last chapter, I will investigate how we can extend the notion of programming by demonstration for the spatially distributed collective elements, and how the user can program the dynamic behavior of them through direct physical manipulation.

Programming by demonstration is one way to program the physical behavior, such as robots or tangible user interfaces, through direct physical manipulation, instead of coding on a computer screen. When programming the behavior, it typically involves writing code on a computer screen and then deploying the code to see results in the physical space. In contrast, directly manipulating objects in the physical environment has many benefits over the traditional approach of programming on a screen. For example, prior work has shown that programming in the physical space can be significantly more engaging than a visual programming language, particularly in educational contexts [Horn et al., 2009]. Also, it can reduce a gulf of execution and evaluation for users [Fabry and Campusano, 2014], because when the user does not need to go back and forth between the computer program and the actual physical behavior to see how it behaves.

The idea of programming by demonstration is well explored

in robot programming [1] or constructive assembly [2], when we apply this idea for a swarm of elements, we face several challenges. First, the user cannot manipulate more than two objects even with their both hands, we need to give a user a way to program the whole object, instead of each single elements. Therefore, we need to have a certain abstraction of collectively constructed shape. One metaphor here is, for example, when we program an animation of a shape, say a square, on a graphical screen, we do not program each pixel on the screen nor each vertex of the square, rather we first construct an abstraction of the shape, then change the parameter of the shape (e.g., position, color, width, height, etc) to make it animate. Taking this inspiration, we investigate how we can apply these programming concepts of both abstraction and programming by demonstration for the context of swarm user interfaces. Second, the swarm of the elements have a unique property of becoming both the shape of the element (e.g., can construct a square, a star, or a circle) as well as the user interface element (e.g., buttons, sliders, or tokens). Therefore, it is interesting to explore how we can allow the user to pick up an element to program it as a button or a slider, so that the user can start using it as a button or a slider in improvisational ways. This improvisational program can be supported with the object-oriented programming (i.e., program the behavior based on messages between the different abstracted objects), and we explore techniques to support such interactions through direct physical manipulation.

In this chapter [3], I will first introduce the a four-step workflow of swarm user interface programming, and then propose a set of direct physical manipulation techniques to support each step in this workflow. Finally, I will demonstrate these techniques with Reactile system — a swarm of actuated elements with electromagnetic coil arrays to create a 2D shape

[1] Frei, P., Su, V., Mikhak, B., and Ishii, H. (2000). Curlybot: designing a new class of computational toys. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 129–136

[2] Raffle, H. S., Parkes, A. J., and Ishii, H. (2004). Topobo: a constructive assembly system with kinetic memory. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 647–654

[3] Suzuki, R., Kato, J., Gross, M. D., and Yeh, T. (2018a). Reactile: Programming swarm user interfaces through direct physical manipulation. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 199. ACM

— to show how the user can program the behavior of swarm elements on the fly through direct physical manipulation.

10.2 Designing Swarm UI Programming

10.2.1 Swarm User Interface Programming

In recent years, Swarm User Interfaces (Swarm UI) [Le Goc et al., 2016] have emerged as a new paradigm of human-computer interaction. In swarm user interfaces, a swarm of elements can dynamically form shapes and morph to other shapes to display information in response to user inputs and surrounding environments [Dementyev et al., 2016; Kim and Follmer, 2017; Le Goc et al., 2016]. As we showed, there is the great potential of Swarm UI in many application domains, such as dynamic data physicalization [Le Goc et al., 2016], simulations and problem-solving [Patten and Ishii, 2007; Patten, 2014], wearable and tangible displays [Dementyev et al., 2016; Kim and Follmer, 2017], and accessibility assistants [Suzuki et al., 2017]. However, the current practice of programming a Swarm UI application is closer to *robot programming* — users work on a computer screen and think in terms of low-level controls —, rather than *UI programming*. To design interactive UI applications, programmers often must think in terms of higher-level design for user interaction, whereas robot programming tends to focus on low-level controls of sensors and actuators. We stipulate that current approaches to programming Swarm UI are too robot-centric to be effective for building rich and interactive applications. As a first step toward answering this question, this project explores a new approach to programming Swarm UI applications.

We propose *Swarm UI Programming*, a new approach to building Swarm UI applications that focus on high-level UI design.

The workflow of Swarm UI programming is inspired by the existing UI programming paradigm. We first review the common workflow of UI programming and decompose it into four basic elements that represent high-level steps. Then we discuss how to apply this workflow to Swarm UI programming.

10.2.2 Four Elements of Existing UI Programming

As we see in well-known design patterns for interactive UI applications such as reactive programming paradigm, the Model-View-Controller, and the observer pattern, they share a common workflow consisting of four basic elements: **1) create elements**, **2) abstract attributes**, **3) specify behaviors**, and **4) propagate changes**.

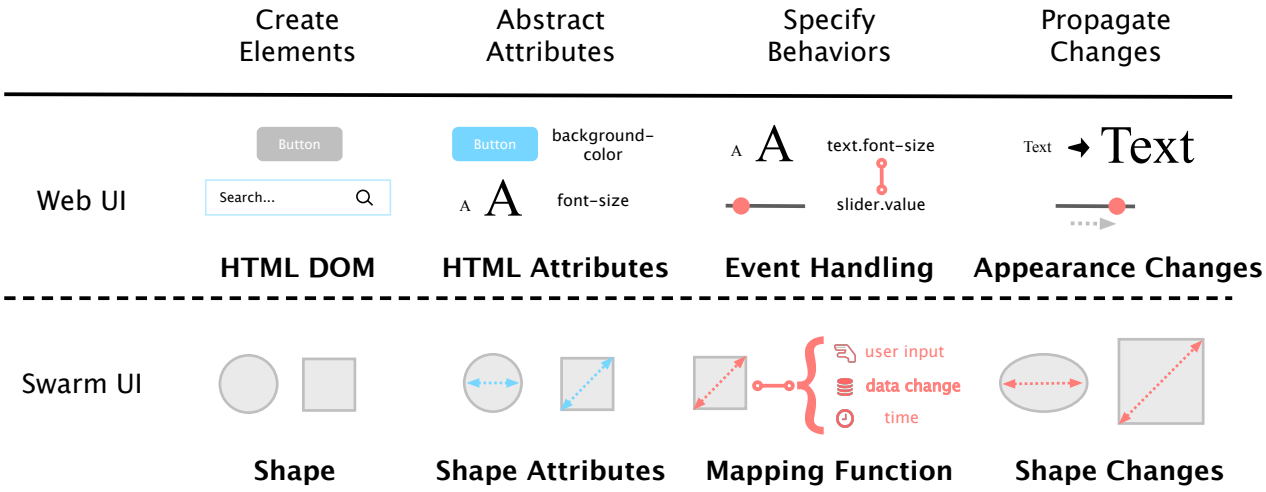


Figure 10.2: Four basic elements of Web UI and Swarm UI programming.

Consider, for example, making an interactive web application using HTML and JavaScript:

1. **Create elements:** A user first creates basic elements of interface with HTML DOM such as `div`, `button`, and `text`.
2. **Abstract attributes:** Then, the user abstracts these attributes as variables, such as the background color or font-size. These attributes can be changed dynamically by updating variable values.

3. **Specify behaviors:** The user specifies behaviors to describe how abstracted attributes will change with data-bindings. For example, one can specify that the button's `background-color` will change in response to the `text` attribute of the input element.
4. **Propagate changes:** Based on the user-defined data-bindings, the system automatically propagates the change by detecting user input or data changes. For example, detecting an input value such as "brown", automatically changes the `background-color` attribute of the button element.

10.2.3 Four Elements of Swarm UI Programming

Now, we draw a parallel between UI programming and Swarm UI programming by introducing the following four-step workflow:

1. **Create elements:** In Swarm UI programming, we propose that shapes are basic UI elements, as the swarm can represent information and communicate with a user through changing shapes. A shape in Swarm UI comprises of a swarm of small tangible objects. In this paper, we denote each unit as a "marker" which can be either a robot or an actuated tangible object.
2. **Abstract attributes:** As in a web application, a shape is a static element. To dynamically change a shape, the user must introduce attributes such as width, height, scale, position, angle, radius, and curvature. For example, the user can define an angle attribute of an arrow or a radius of a circle, which can be changed through programming.
3. **Specify behaviors:** To make an interactive Swarm UI application, the user can specify how a shape's attributes change when an event occurs. The event can be user input, changes in the external data source, or the progress of time.

4. **Propagate changes:** Once the user specifies the behavior, the system can watch for changes to the control unit. For example, if someone moves the control marker, thereby increasing `x`, the system automatically updates the arrow shape's `angle` attribute.

10.3 Swarm UI Programming via Direct Physical Manipulation

Given the physical nature of Swarm UIs, we propose to support this programming workflow via *direct physical manipulation*. Rather than coding in a separate IDE on a computer screen, a programmer should be able to program a Swarm UI by physically manipulating the swarm.

To achieve this goal, we propose the following direct manipulation workflow:

10.3.1 Step 1. Create Elements by Drawing and Construction

The first step to programming a swarm UI application is to make shapes. A programmer can make shapes in two ways; 1) moving and arranging individual swarm markers into the desired shape, 2) drawing the desired shape with a freehand stroke. In either case, the hand-made shape need not be perfect. The system should guess which basic shape (e.g., line, circle, triangle, rectangle) the programmer is trying to make and beautify it when possible. Then, a swarm of markers moves to corresponding positions to form a shape. The user can also manually modify the shape by placing or removing individual markers.

Once a shape is made, the system constructs a class for that

shape and adds it to the program space. This allows the programmer to later abstract the attributes of the shape and clone a shape as an instance. The current states of the program, such as the set of shape classes and associated variables, is visualized in a side panel as spatial information. Each shape class is represented by a similarly shaped icon in the control panel. To instantiate an object of the class, the programmer first places a marker at a class window and then moves it to the workspace, then markers in the surroundings form the shape.

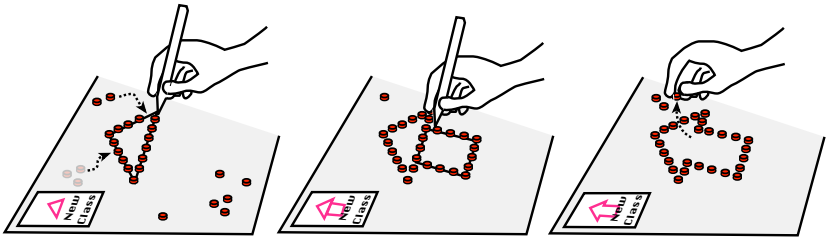


Figure 10.3: Create elements by drawing and construction. A programmer can create elements by arranging markers or drawing the desired shape.

Figure 10.3 shows a programmer making an arrow shape in the programming environment. The following pseudocode illustrates how a program evolves over the three steps. First, the programmer draws a triangle (Figure 10.3 left), then the system adds a triangle in the program space.

```
<triangle x="0" y="10"/>
```

By drawing a rectangle (Figure 10.3 middle), the system adds another shape.

```
<triangle x="0" y="10"/>
<rectangle x="5" y="15"/>
```

The programmer can remove a horizontal line by directly picking up markers and putting them aside (Figure 10.3 right). Once a shape is created, the environment adds the current shape as a class the programmer can name.

```
<arrow x="0" y="10"/>
```

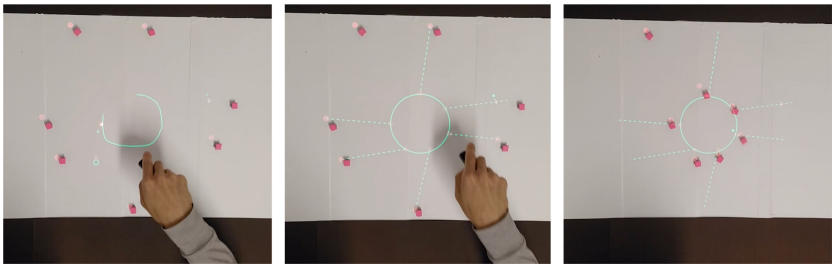


Figure 10.4: Reactile allows a user to draw a basic shape with a laser pointer.

10.3.2 Step 2. Abstract Attributes through Demonstrations

One important aspect of programming is the ability to generalize a specific case using a higher-level abstraction. Suppose a programmer has constructed an arrow shape and wants to change its orientation. To do so, the programmer can abstract an attribute of a defined shape by introducing a variable. For example, the following pseudocode illustrates how this operation can be done in a common programming language. To change the orientation of the arrow, the programmer can simply set `a` to a different value.

```
var a = 30
<arrow angle={ a } x="0" y="10"/>
```

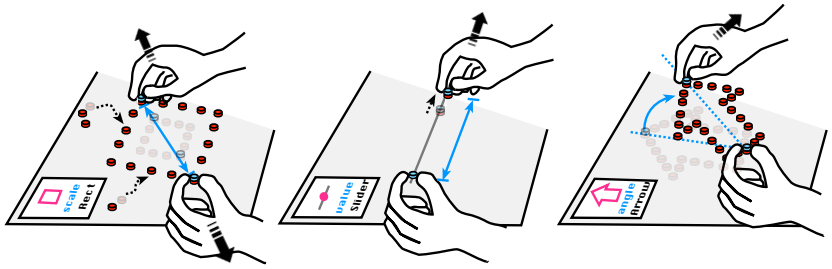


Figure 10.5: Using constraint markers to specify different shape attributes: diagonal length, position, and angle.

To support a programmer to abstract variables through direct manipulation, we take inspiration from constraint-based drawing [4]. Our system uses tangible constraint markers. To define a variable to represent a certain shape attribute, a programmer puts constraint markers on an existing shape. The system envi-

[4] Sutherland, I. E. (1964). Sketchpad a man-machine graphical communication system. *Transactions of the Society for Computer Simulation*, 2(5):R–3

ronment infers which shape attribute the programmer is trying to demonstrate. For example, Figure 10.5 illustrates how a programmer uses constraint markers and demonstrations to define a variable to abstract angle attribute, as in the pseudocode above.

Different demonstrations can define different variables such as position, width, height, scale, and orientation. For example, Figure 10.5 shows other examples of abstracting A) a rectangle's scale attribute, B) a marker's x position attribute, and C) an angle of an arc shape.

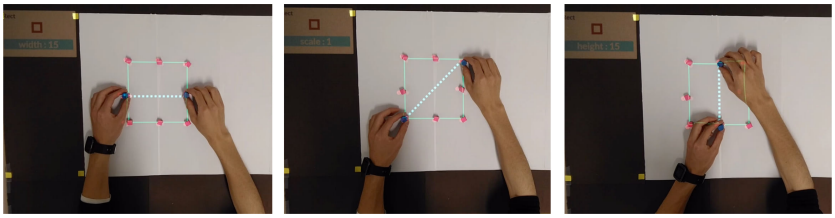


Figure 10.6: Reactile lets a user to abstract attributes as variables through demonstration with blue constraint markers.

If our system sees that the programmer exhibits a behavior matching one of the heuristics above, it creates a variable for the attribute suggested by the heuristic and adds it to the program space. Each variable is visualized as a window containing the shape's icon and the attribute's name.

10.3.3 Step 3. Specify Behaviors by Connecting Attributes

After creating shape classes and abstracting some of their attributes as variables, the next step is to specify their behaviors. To specify how certain attributes may change based on the user input, the programmer can create a mapping function to relate each variable. In the left panel where the program space is visualized, variables already defined show up as individual windows. To specify a data binding, the programmer selects two variables, then the system adds a data binding to the pro-

gram space. It also provides visual feedback by showing a line between the two variables.

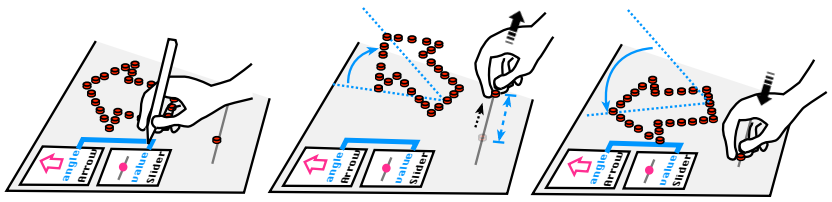


Figure 10.7: Specifying behaviors by creating bindings between variables. Once a programmer connects two attributes by placing selection markers, then the system automatically binds them and propagates the change.

Suppose a programmer wants to specify the following behavior: when a point is dragged to the right, the angle of the arrow rotates clockwise. The pseudocode implements this behavior.

```
1 var a = 30;
2 var b = 10;
3 bind(a, b)
4 <arrow angle={ a } x="0" y="10"/>
5 <point x={ b }/>
```

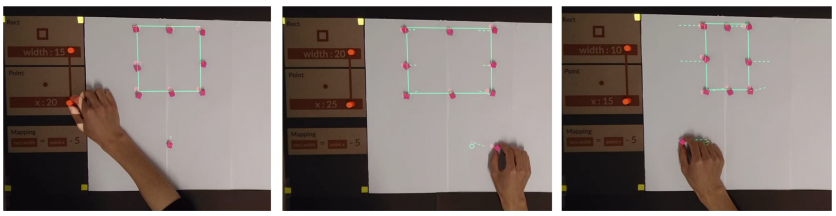


Figure 10.8: The user can create a mapping function with orange selection markers (e.g., rect.width = point.x - 5). Once the mapping function is created, the system can automatically propagate changes whenever the variable value is changed.

This implementation involves choosing a marker in the swarm to be the control (line 5) and abstracting the marker's x position attribute as a new variable b (line 2). Then, a binding is defined between a and b, using a pseudo-function `bind()` (line 3). Based on the current value (e.g., $a = 30$ and $b = 10$), the system automatically creates an appropriate mapping function (e.g., a = b + 20). If the user wants to define the different data binding, the user can select different expression suggested by the system (e.g., a = b * 3) or modify the expression (e.g, a = b * 360 / 100). When the program is running, the system can

watch for changes in b and propagate the changes to a, achieving the desired behavior—the arrow rotates as the marker is moved right.

10.3.4 Step 4. Propagate Changes through Physical Interaction

Once a programmer specifies the behavior by connecting attributes, the system automatically detects the change in the value of the associated variable and propagates the changes.

Figure 10.7 shows how a programmer uses direct manipulation to bind two variables to specify the dynamic behavior described above. In A), he puts two markers on each variable's window, which is equivalent to `bind(a, b)`. In B), he drags the control marker to the right; the arrow rotates accordingly. In C), he drags the control marker to the left; the arrow rotates in the opposite direction.

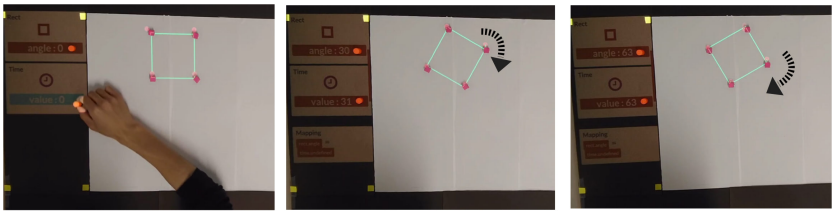


Figure 10.9: The user can also create a mapping function between at- tributes and time-dependent variable for continuous animation.

10.4 Application Scenarios

10.4.1 Data Physicalization

Data physicalization is a promising research area where Swarm UIs can be useful [Le Goc et al., 2016], particularly to help blind people understand and explore data [Suzuki et al., 2017]. While existing research work have studied how users interact with data, there is relatively less work investigating how users *author* their own dynamic data physicalizations. Using Reac-

tile, users can "physicalize" data by connecting data values to representative shape attributes, such as the size of a circle or the length of a line. This connection can be specified using the direct manipulation techniques described above.

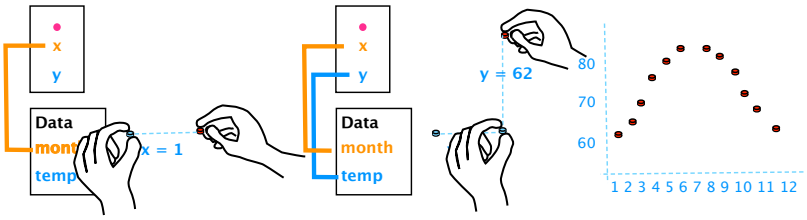


Figure 10.10: Application scenarios of data physicalization.

For example, in Figure 10.10, a user wants to create a graph that represents the temperature of a city throughout the year. She first defines the x and y variables using a reference point. When she connects the variable x to the month data (Figure 10.10), the system notices that the month data has twelve integer values and automatically instantiates eleven more objects from the same class and propagates to the next value with an one-to-many mapping. In this way, the user now has twelve single points that are horizontally distributed with different x values. Next, she connects a variable y to temperature data, and the system propagates the y value to each object. In this way, the Swarm UI displays a 2D plot whose x-axis represents the month and y-axis represents the temperature of that month.

10.4.2 Explorable Tangible Simulation

Tangible representation serves as a powerful medium to engage people with physical objects. Prior work has shown that two-handed tangible interaction helps users to explore simulations and problem-solving [Patten and Ishii, 2007]. With Reactile, users can not only interact with such explorable simulations as a consumer, but also create them as an author.

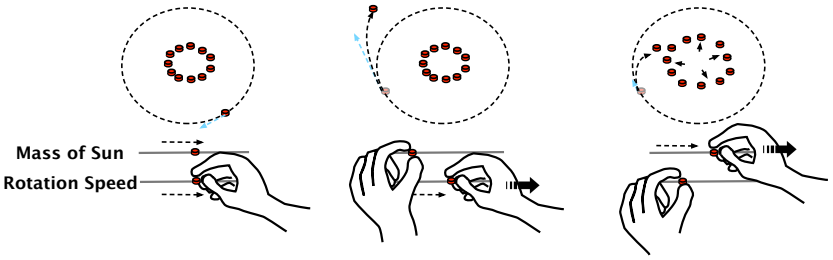


Figure 10.11: Application scenarios of tangible exploration.

For example, a teacher in a physics class wants to demonstrate how the mass of the Sun affects the orbit of the Earth. First, she makes a circle shape to represent the Sun and abstracts the circle's "radius" as a variable. Next, the teacher uses a marker to represent the Earth; she makes it revolve around the sun by connecting it to a time-dependent variable. She creates two slider objects: one controls the radius of the Sun and the other controls the velocity of the Earth's orbit. To demonstrate to her class, she holds the two slider objects and moves them sideways simultaneously to animate the shape of the Sun and the movement of the Earth (Figure 10.11). She shows that if the difference between the two values is too big, the Earth falls into the Sun or flies into outer space. By showing this, the teacher interactively demonstrates how gravity and the velocity of an orbiting object affect each other.

10.4.3 Ambient Display and Animation

Swarm UIs are also promising for ubiquitous interfaces which show information as an ambient display [Kim and Follmer, 2017]. Creating interactive animation of such displays could be also an interesting application. For example, a user could make a timer or a progress bar to indicate its status with Reactile. To make a radial progress bar, a user first creates an arc shape and abstract its angle as a variable, so that she can bind the angle variable to the real-time data. Then, when the progress data increases, the arc becomes a circle shape to indicate its

progress.

10.5 Discussion

10.5.1 User Evaluation

We conducted a survey study and a lab study to understand programmers' experiences as well as the appropriateness of our approach, focusing on the following research questions:

> **RQ1:** Is the representation and behavior of a program easy to understand, predict, and modify?
> **RQ2:** Do programmers find the proposed interaction techniques intuitive?

where, we used the term "intuitive" as "the behavior of the interface is easy to expect".

Participants

For the survey study, we recruited subjects from a large upper-level computer science course. Students were expected to all have prior programming experience. A total of 148 students participated in our survey. Because the survey was anonymous, we do not have demographics.

For the lab study, we recruited eight participants (7 male, 1 female), ages 19-31 (average: 24.3) years old from our institution. Having prior programming experiences was an inclusion criterion. All participants were from engineering majors (4 computer science, 2 mechanical engineering, and 2 electronic engineering). Each session approximately took 45 minutes.

Method

For the survey study, we designed a set of quiz questions to

test to what extent participants were able to understand the programming techniques we proposed for the four-step workflow. Before seeing the questions, participants watched a short demonstration video. Each question contained one or more photos to illustrate a direct manipulation technique and asked participants to predict the outcome by selecting from four choices. Twelve questions were included in the survey.

For the lab study, the goal was to provide participants with an opportunity to physically interact with our programming environment. Each participant was explained the purpose of the study, shown a demonstration of the system, and given a simple programming task to perform. After the task was finished, the participant received a short survey containing eight questions. Five questions asked them if the proposed interaction design was intuitive. The other three questions examined the participants' opinions on whether the program is easy to understand and modify, and if the proposed interface seems flexible for many different applications. Participants answered on a 7-point Likert scale where 1 is strongly disagree and 7 is strongly agree.

Result

Our survey study yielded mixed results. Participants performed relatively well on the two quiz questions about the prediction of Step 2 Abstract attribute with a correctness rate of 67%(93/139) and 87%(128/146). Among those who got incorrect answers, the most common type of confusion was between the *height* attribute and the *y* attribute of a rectangle. Note that because participants could skip questions, the n was slightly different for each question.

On the three questions concerning Step 4, however, only 43%(58/135), 46%(59/129), and 33%(46/138) of the participants answered

correctly. The two questions that most challenged the participants concerned Step 3. Only 22%(30/138) and 35%(44/126) of the participants answered correctly. The accuracy rate was close to random. The results were below our expectation. One reason could be that the survey instrument did not provide the fully *tangible* interaction experience; participants only saw video and photo illustrations.

	Average Score (SD)
Overall user interactions	6.0 (0.7)
Step 1: Create elements	5.8 (0.6)
Step 2: Abstract attributes	5.5 (1.3)
Step 3: Specify behaviors	5.4 (1.2)
Step 4: Propagate changes	5.0 (1.3)
Easy to understand	6.1 (0.9)
Easy to modify	5.0 (1.6)
Flexible for different applications	6.1 (1.3)

Table 10.1: Summary of 7-point Likert-scale responses.

Our lab study, on the other hand, showed more promising results. Overall, participants had a positive view of their experiences with our proposed Swarm UI programming environment. The table above shows a summary of 7-point Likert scale response to each question. Overall, participants generally agreed that the proposed interaction techniques were intuitive ($6.0, \sigma = 0.7$). Also, they thought that the program was both easy to understand ($6.0, \sigma = 0.7$) and to modify ($5.0, \sigma = 1.6$) and that it can be flexible for different applications ($6.1, \sigma = 1.3$). The next section discusses these results to gain insights for an appropriate design for Swarm UI programming.

10.5.2 Findings

Usability

The participants in our lab study generally agreed that the proposed interactions are natural and intuitive, by stating that using two-hand interactions makes programming fun (P7)

and engaging (P8). Particularly, three participants, who have prior experiences in robot programming, identified the benefits of programming in the physical space. P1 mentioned that how our approach reduces the barrier of programming such swarm user interfaces by comparing it to his past experience in programming swarm robots; *"while programming these robots, I usually need to compile it, deploy, and see how it works every single time."* (P1)

Moreover, participants are excited by the new opportunity for users to create Swarm UI applications without programming knowledge. For example, P2 saw a great potential for classroom use such as in math education, stating that *"One application I had in mind was education. For example, teachers in middle schools can teach geometry such as sine or cosine by interactively demonstrating with these markers. Students can also interact with it to understand math."* (P2)

Interpretability

Similar to survey study participants, some lab study participants found it difficult to predict program behaviors. This difficulty might be due to the task design. In general, participants may have difficulty with correctly understanding and predicting a program without actually constructing it, particularly in an unfamiliar system or programming language. Although participants generally agreed that Step 3 and Step 4 are easy to understand, they also commented that these steps can be improved. For example, P3 suggested that the system should visualize data-bindings directly on the swarm markers, as opposed to in the left panel only. Indicating the active attributes with highlighted auxiliary lines can help improve the interpretability of variable mappings and specified behaviors.

Flexibility

While contextual information helps, a separate program space contributes to the flexibility and generalizability of a program. For example, P4 stated that the displayed information in the left panel was helpful for him to understand the structure within a standardized view. Thus, one important design implication is the need to make the appropriate connections between the abstract (e.g., variables and class) and the concrete (e.g., shapes) spaces in order to enable better mental models between these representations, while still maintaining the flexibility and generalizability of the program.

Scalability

In the user study, P7 wondered if the program could scale to more than a few shapes and attributes. One way to handle a large number of shapes and parameters is to provide contextual information which only shows the related parameters or binding information in the left panel. As we see similar experimental programming interface in GUI, such as Apparatus [Schachman, 2015] and Sketchpad14 [Samimi et al., 2015], we expect this approach can also handle scalability with a similar design.

Part IV

Discussion

11

Discussion

"What will smart places do for us? They will, of course, collect and spit out information — much as computers and telecommunication devices have always done. More importantly, though, they will attend, anticipate, and respond to our daily needs in a vast variety of new ways. And they will become delivery points for a still-unimaginable range of services made available by providers scattered around the globe. [...] These developments suggest a new evolutionary stage for architecture. Our buildings will become less like protozoa and more like us. We will continually interact with them, and increasingly think of them as robots for living in."
— William J. Mitchell [1]

[1] Mitchell, W. J. (1999). *e-topia:" Urban life, Jim—but not as we know it"*. MIT press

Throughout this thesis, I have explored different methods, representations, and applications of dynamic and collective shape construction. I have also explored an interaction technique for programming these elements through direct physical manipulation. In this last chapter, I will discuss future research opportunities for collective shape construction, highlighting design implications and lessons learned through the previous

explorations.

11.1 Expanding the Design Space

As discussed in Chapter 4, there are several different ways to construct a dynamic shape. This thesis provides some examples of these possible representations (e.g., sparse dots, sparse lines, voxels, pin arrays), but this is still an open area for us to explore.

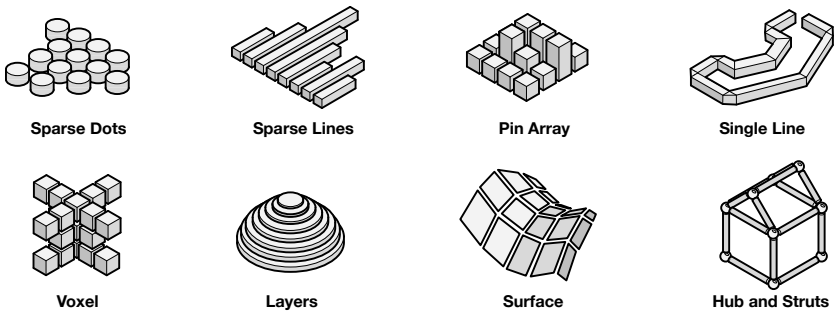


Figure 11.1: Different types of shape representations

For example, how can we construct an expressive dynamic shape with layer, surface, and hub and struts representations? And how can we practically achieve general-purpose shape-changing interfaces with these representations? For layer representation, consider for example a stack of transformable sheets that construct a cubic shape. By individually changing the shape of each sheet, can we transform into a sphere, a bowl, a cup, and a bottle-like shape? Also, what would be the design requirements for each robot that can construct an arbitrary shape with a surface-like representation? By expanding the design space, we could also discover new and interesting interactions with these shapes.

To effectively explore new potential domains, this thesis suggests two design exploration strategies: 1) combining the capability of each element, and 2) combining passive and active

elements.

11.1.1 Expanding by Combining Individual Capabilities

The first strategy is to explore a combination of the different individual capabilities of active elements. For instance, common capabilities of active elements include:

- change in the horizontal position of elements
- change in the vertical position of elements
- change in the orientation of elements
- change in the length of elements
- change in the volume of elements
- change in the connectability of elements
- change in the 2D shape of elements

Figure 11.2 also illustrates some of the capabilities that each active element could have in differenting representations.

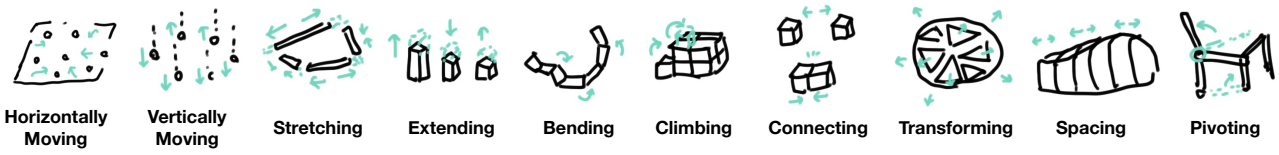


Figure 11.2: Exploring different types of individual capabilities as building blocks of active collective elements.

By combining these capabilities, we could expand the expression, interaction, and applications of such an interface. For instance, ShapeBots project [Suzuki et al., 2019b] illustrates how a combination of *horizontal movement* and *vertical or horizontal extension* of swarm robots can expand the expression of the current swarm and shape-changing interfaces (e.g., transforming into sparse dots, lines, and pin-array representations).

There are still many more different combinations that we can

explore. For example, consider collective elements that construct a single line (similar to LineForm [Nakagaki et al., 2015] or ChainForm [Nakagaki et al., 2016]), but each element can both bend and extend its length (Figure 11.3 left). Such a structure could enable more expressive shapes by creating shapes beyond a fixed length or by transforming shapes with scale.

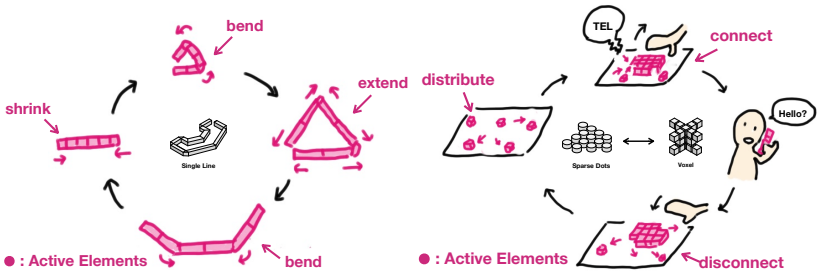


Figure 11.3: Left: Combining bending and extending capabilities to achieve scalable, shape-changing single line structure. Right: Combining the locomotion and connection/disconnection capability to switch between sparse dots and voxel representations.

Also, what if each swarm robot can also dynamically connect and disconnect with other robots? For example, these robots can construct a dynamic shape by spatially aligning elements (e.g., sparse dots representation), but — when needed — they can assemble together to create a graspable 3D shape that the user can hold (e.g., voxel representation). We could also imagine these robots, once no longer needed, can disassemble themselves to disperse into their environments (Figure 11.3 right). In this way, we can construct in-situ graspable objects and tools on demand. In addition, what if each element can also bend or extend each other? For example, if each line of ShapeBots [Suzuki et al., 2019b] can connect and disconnect, these elements can not only create a graspable shape that the user can pick up, but they can also interactively transform in the user's hands. This new type of collective shape-changing interface can expand the range of applications and interactions.

I am also interested in enhancing each element with additional input or visual output capabilities. For example, if each block element of voxel representations (e.g., Dynablock [Suzuki et al., 2018b]) can also dynamically change color, then they can serve

as a volumetric graphical display. Or if each block can sense the neighbors, these blocks could recognize the current shape and user interaction. In this way, such objects can serve as smart blocks that can automatically sense how the user modifies the shape. Such capability would be useful for remote collaboration, where if one user changes the physical shape, then such changes instantly reflect the shape in distance.

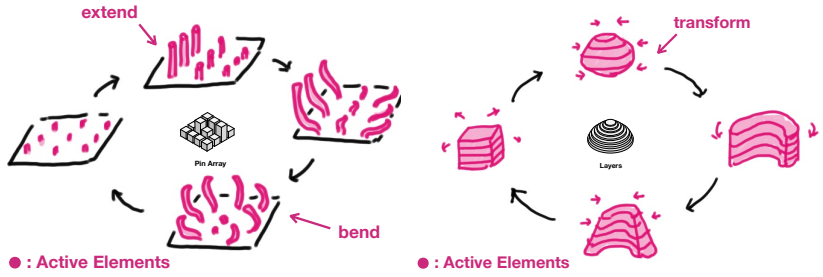


Figure 11.4: Left: Combining extending and bending capabilities for bendable pin-based shape displays. Right: Combining the transformation of each layer to change the overall shape.

In summary, by combining different individual capabilities (e.g., mechanical actuation capability, input capability, visual output capability), we could extend the current scope of collective shape-changing interfaces that can open up new interaction and application possibilities.

11.1.2 Expanding by Combining Passive and Active Elements

The second strategy is to further explore the hybrid approach of combining passive and active elements. As discussed in Chapter 4, creating a shape out of only active elements is often difficult to scale up the number of elements. Therefore, I have demonstrated leveraging passive materials as constructing elements could be an interesting alternative approach.

Yet, this hybrid approach is not previously well explored, so there are many open research areas that we can explore in different combinations of elements or different representations of shape construction. For instance, how can we construct a dy-

namic shape with passive materials in surface representation? Can we create a transformable surface out of passive materials in which a swarm of robots can selectively fold to create a 3D like origami? How about the dynamic shape representation of passive layer materials? For example, can a machine continuously feed clay-like materials and shape layer-by-layer? Or can the machine cut such materials with machines aligned in parallel (e.g., styrofoam cutter like a machine) to shape each layer and stack them to shape and reshape dynamically?

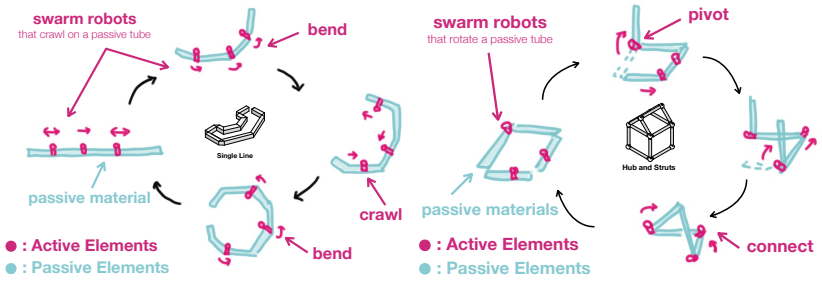


Figure 11.5: Examples of using a swarm of active elements to collectively transform passive materials to construct a shape.

Alternatively, how can we combine passive and active elements for line-based representations? For instance, consider swarm robots that can crawl on a passive material (e.g., a soft inflatable tube or rigid pipe), and bend a point with a wire-bending-like feature. They can collectively bend selective points of the line so that we can transform a line into a square, a star, or a circle dynamically. Since the structure is made out of passive materials, we can make much larger and stable objects.

Similarly, can we also explore the combination of active robots and passive materials to construct or transform a hub and struts shape? For example, imagine a swarm of robots that can pick, move, and connect timber-like structure. By collectively moving each element, such robots could dynamically construct a framework shape (e.g., similar to Protopiper [Agrawal et al., 2015], but autonomous shape creation and transformation with embedded active elements).

The combination of active and passive elements for shape-changing interfaces is a relatively new idea. In his thesis, Schoessler first explores this idea and introduces the concept of Shape Synthesis [2], in which combining inert objects with shape display promise to expand the expression (e.g., creating a topology that a shape display does not support) and interaction capability (e.g., the user can use an inert object as a token to manipulate). Beyond the pin-based shape display, I expect the combination of collective active elements (e.g., swarm robots) would open up a new research opportunity by broadening the application space. There is an interesting future research opportunity to further explore.

[2] Schoessler, P. (2015). Shape synthesis: physical object augmentation and actuation for display and interaction on shape changing interfaces. Master's thesis, Massachusetts Institute of Technology

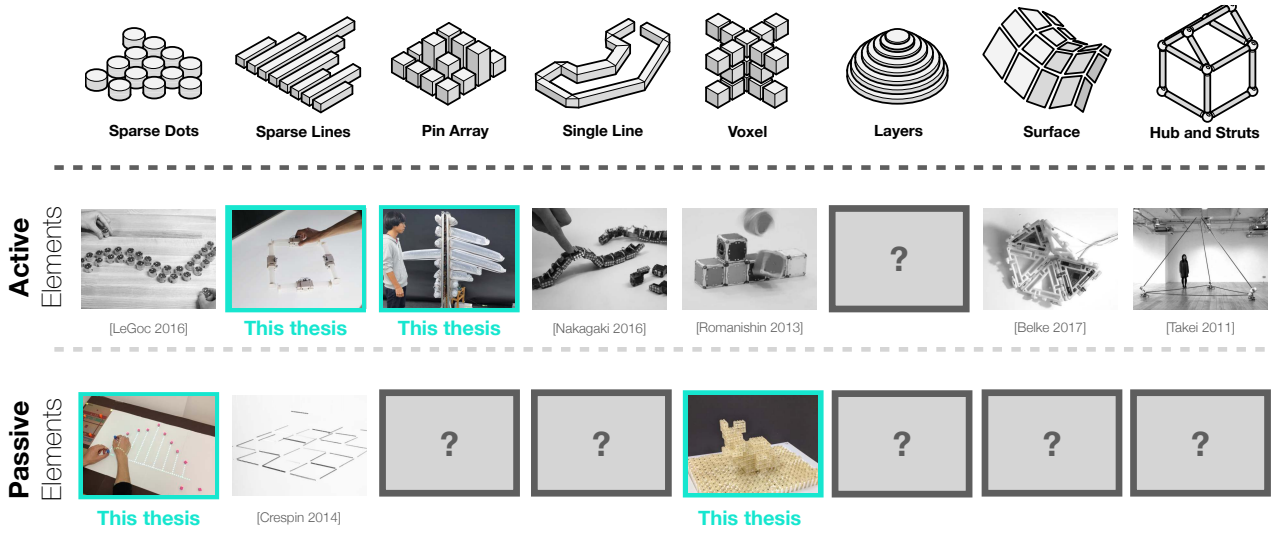


Figure 11.6: Future opportunities for *general-purpose* shape construction and transformation with different representations.

As we can see, there are still a lot of research opportunities in both active and passive shape construction. To effectively guide and explore the open field, this thesis contributed to formalizing the design exploration strategies, demonstrating them through actual implementations, and providing a framework to effectively explore for future opportunities.

11.2 Collective Actuation

11.2.1 Inter-material Interaction between Swarm Robots and Transformable Materials

Related to the above point, there is also a rich design space of how these robots can collectively actuate and engage the world. Active swarm elements can not only construct a shape by themselves, but also "transform existing materials" by interacting with them. For example, Figure 11.7 illustrates how a swarm of active elements can interact with a passive material to transform it. These types of actuation include moving, trembling, extending, contracting, expanding, shrinking, and vibrating. Some of these shape transformations are particularly unique because it may not be possible with the actuation with a single robot. For instance, stretching or enclosing passive materials is only possible with the collective behaviors of multiple robots.

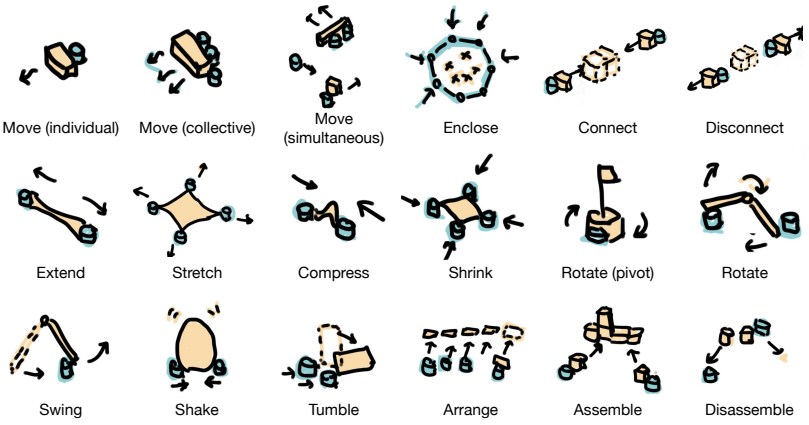


Figure 11.7: Design space of collective actuation.

11.2.2 Collective Actuation for Adaptive Environments

Such an actuation becomes particularly interesting when they interact with transformable passive materials. For example,

consider a swarm of robots that actuate transformable furniture made of flexible materials (Figure 11.8). By interacting with such furniture, they can adapt to the user. For example, if one person approaches, they can transform into a chair, and when another person comes, this becomes a sofa. These robots can also transform the furniture into a different shape, like a coffee table to adapt to the user or by changing the shape to provide affordances.

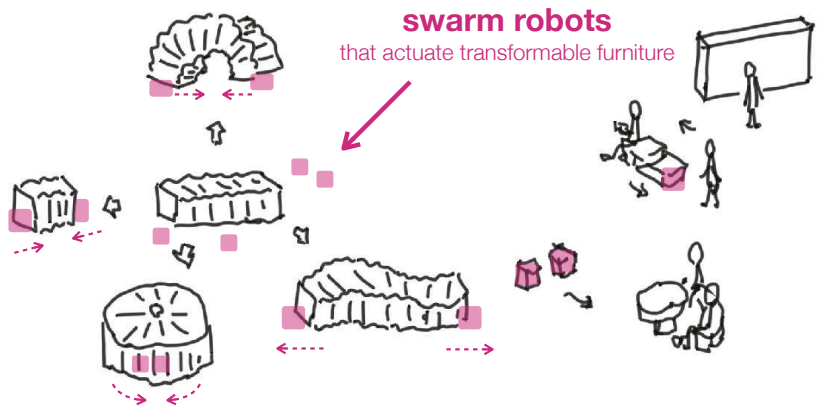


Figure 11.8: Transformable furniture enabled by collective actuation of distributed robots.

Such a collective actuation also provides an interesting prototype environment. Today, one of the biggest challenges in the shape-changing interface research is it is difficult to prototyping [Hardy et al., 2015; Alexander et al., 2018]. Because prototyping requires a lot of efforts to develop software and hardware systems, it is difficult to quickly explore for designers. But, collective actuation allows the user to quickly actuate the existing and passive materials, thus it can decreases the barrier of the exploration. A similar idea is explored with the prototyping environment with shape displays [Nakagaki et al., 2017], but we can also expand this concept for collective actuation. Some preliminary examples can be seen in (e.g., TOIO Papercraft Creatures Gesundroid), but there is an interesting research opportunity to expand vocabulary and techniques for this collective *inter-material interaction* [Follmer et al., 2013].

11.3 Transformation vs Construction

11.3.1 Towards Dynamic Physical Displays

In this thesis, I focus on collective shape construction because it promises the general-purpose shape transformation, which potentially leads us to the creation of dynamic physical displays.

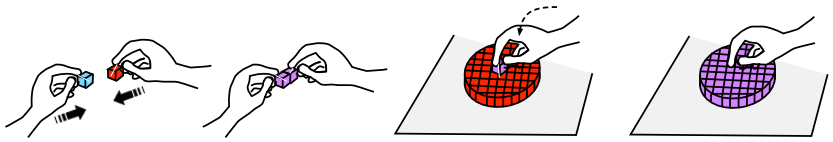


Figure 11.9: Future concept of a programmable physical display with collective elements: the property of the element can dynamically propagate to the other elements.

Despite this potential, some of the approaches are difficult to "transform" one shape to another, due to the current technical limitations. For example, Dynablock cannot dynamically transform the constructed shape. On the other hand, most of the shape-changing interfaces, although they lack the general-purpose shape creation capability, can transform more smoothly and rapidly. Therefore, there seems to be a certain trade-off between general-purpose shape construction vs single-purpose shape transformation.

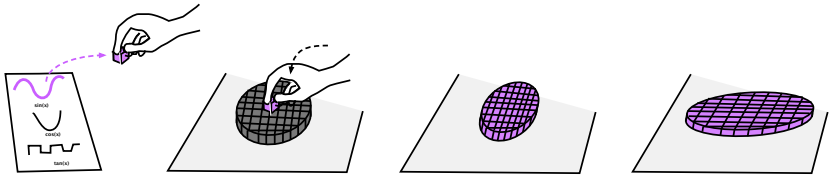


Figure 11.10: Future concept of transformable shape with programmable elements: the constructed object can also transform its shape dynamically in a programmable fashion.

However, this trade-off could disappear if the shape construction and reconstruction are done fast enough. In graphical displays, for example, the animated pictures seem to be transformed from one shape to another, but the reality is the display refresh the entire scene and then redraw the new scene again from the scratch. But, since the drawing and re-drawing of the scene are fast enough for human perception, we see them as an animated transformation from one shape to another. In

the same way, if the shape construction and reconstruction are done fast enough, this can be seen as a transformation from one shape to another. We envision this could be the future of dynamic physical displays.

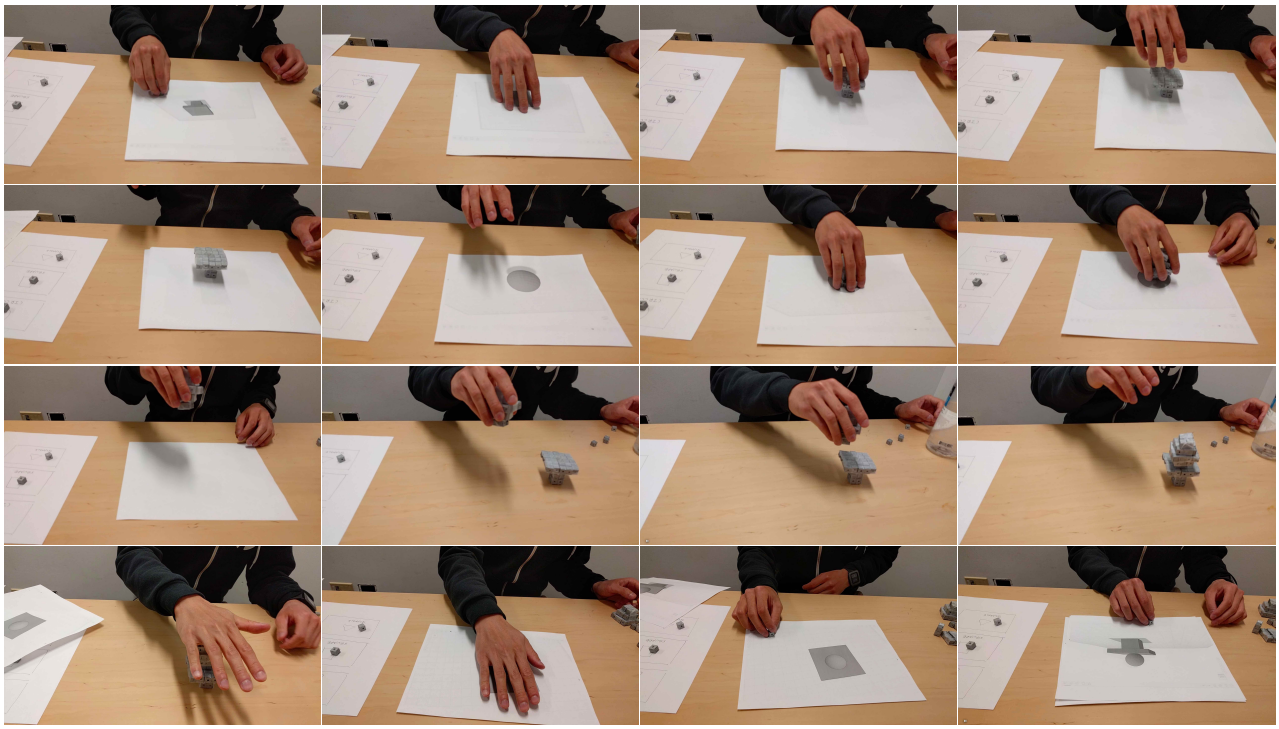


Figure 11.11: Future concept of the dynamic physical display with collective elements.

11.3.2 Swarm Fabrication and Construction

In practice, however, it might be difficult to technically achieve this speed for physical or mechanical movements in the near future. Instead, the dynamic shape construction can be more interesting in another application domain, which is a digital fabrication.

Dynamic and collective shape construction presented in this thesis also has a close connection with digital fabrication technologies. Both aim to physicalize the digital information, without limiting the expression. For example, 3D printers or milling machines can, in theory, create an arbitrary 3D shape, and laser cutter can cut materials into an arbitrary 2D shape.

The dynamic and collective shape construction shares a similar goal, but with the focus of the speed of creation as well as constructability.

If the shape does not need to frequently change, these approaches are particularly useful for reconstructable digital fabrication. This is particularly interesting when we scale it up the fabrication objects and leverage the passive materials with swarm robots. This can be an interesting solution for the current limitation of digital fabrication devices. One of the limitations of digital fabrication devices is its limited size of the resulting object. For example, 3D printers cannot print an object larger than its bed. Laser cutters cannot cut materials larger than its size. Printing or cutting larger scale objects also significantly increases the time. Therefore, such large-scale fabrication, like furniture or buildings, tend to rely on manual construction or leverage the gigantic 3D printer at the scale of buildings.

Instead, what if a swarm of robots can manipulate and construct with passive materials? This approach is promising because the size of the resulting objects can be much larger than each robot, thus it is possible to create larger-scale objects. Also, these robots are small, thus it is easier to bring without a significant cost, like creating a shelter in a disaster place or other planets.

One interesting aspect is its constructability. They can construct and reconstruct over time, thus, for example, they can be useful to construct a temporary building for an event overnight, and then automatically deconstruct or reconstruct for another event once finished. It is also possible to embed active elements in the constructed buildings to keep partly active and reconfigurable so that the resulting object can change its shape to adapt and respond to the user and environments.

11.4 Embedded vs Augmented

11.4.1 Towards Ubiquitous Shape-changing Interfaces

In the future, how can we make shape-changing interfaces distributed in the environment? Today, we live in a world where computers and graphical displays almost *disappear* and *weave into the fabric of everyday life*. Dynamic graphical interfaces — e.g., smartphones, smartwatches, projectors, and digital signage — are now distributed and embedded in our environment. I envision dynamic physical interfaces — e.g., actuated tangible interfaces, robotic graphics, and shape-changing interfaces — should also follow the same path as technology advances. If shape-changing interfaces will become truly ubiquitous, how can these interfaces be distributed and embedded into our everyday environment?

I believe collective shape-changing interfaces could be one of the promising approaches to this question because each element could be distributed and augment the existing environment, in contrast to large, heavy, and immobile devices that are embedded in the environment.

11.4.2 An Analogy of Embedded Graphical Displays vs Spatial Augmented Reality

To understand the difference of embedded vs augmenting, an analogy of graphical displays is useful. In graphical displays, there are fundamentally two approaches to make the display ubiquitous: one is to embed displays everywhere and another one is to augment the environments. For example, placing a computer screen to make our environment more dynamic is one approach. We can have different sizes of screens like mobile phones, tablets, and laptops on a table, and we can also

embed large screens in the wall, or even ceilings and floor. On the other hand, an alternative approach we can pursue is to display the images on top of the environment. For example, by using projection mapping, we can display content on top of a table and a wall.

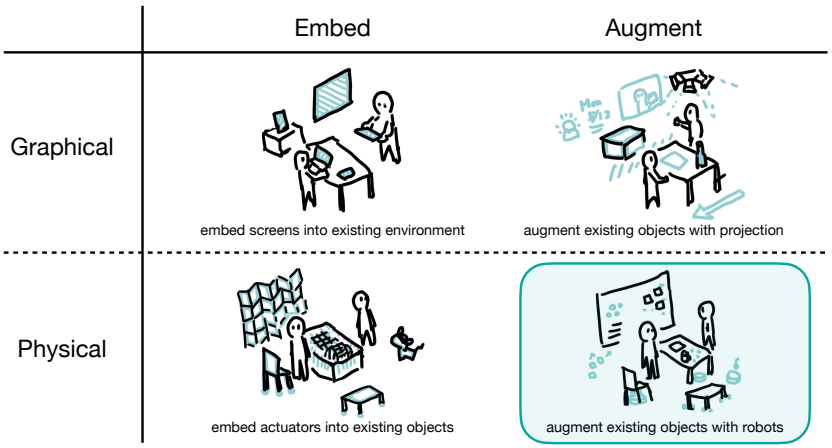


Figure 11.12: Embedded vs augmented. An analogy of graphical display, and parallel discussion for physical display.

Both approaches have benefits and drawbacks. For example, embedded displays enable much higher-resolution of the image and integration of input and output (e.g., touch tracking on a touchscreen vs touch tracking in the real world). On the other hand, embedding displays everywhere may have a scalability problem as it costs based on the size of the screen. It is possible to cover the entire room with screens, but this can be only available within a specific place, such as an entertainment theme park. The size and shape of the screen are fixed and cannot be changed. Also, it makes us feel this approach is not fully integrated between the digital and physical world.

In contrast, the augmented approach allows us to easily scale up to cover the entire surface. It also easily displays the dynamic content on top of a non-flat surface to augment them. In this way, this approach can make a static object visually dynamic, which makes us feel the digital content is fully integrated and seamlessly blended into the physical world. On

the other hand, there is a certain limitation in terms of the resolution, as well as the integration of input and output is also challenging.

11.4.3 Augmenting Environments with Distributed Collective Elements

We can see the parallel discussion for the physical display as well. As I discussed, most of the existing systems explore the embedded approach. For example, we can embed the pin-based shape-changing display onto a table [Follmer et al., 2013; Vink et al., 2015], mobile phone [Jang et al., 2016], wall [Goulthorpe, 2006], and floor [Tang et al., 2011; Suzuki et al., 2020b; Teng et al., 2019]. This approach has benefits in terms of resolution and dynamicity. For example, such property is highly beneficial for remote collaboration and teleexistence [Leithinger et al., 2014]. On the other hand, the size of the display needs to be fixed and difficult to scale it up.

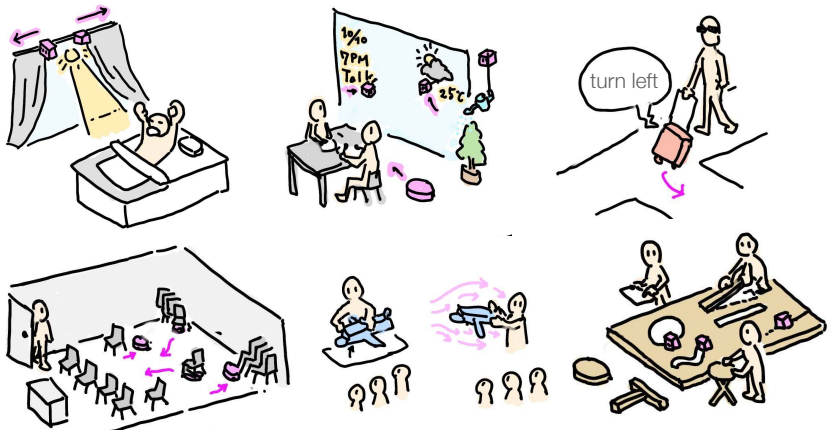


Figure 11.13: Living with swarm robots, where a swarm of distributed robots become a part of environments and calmly support our everyday life.

On the other hand, distributed and collective shape-changing elements could augment the existing environment (Figure 11.12). Since these elements can not only construct a shape but also actuate existing objects and reconfigure physical environments in a programmable fashion. Therefore, these elements can integrate and blend themselves with an existing environment and

make it more adaptive, dynamic, and programmable through interacting with both users and their surroundings. Follmer discussed the use of shape display is not only displaying contents or UI elements but also actuating passive objects to mediate interactions [3]. Particularly, for this second use scenario, the distributed and collective approach presented in this thesis provides a promising complementary way.

[3] Follmer, S. S. W. (2015). *Dynamic physical affordances for shape-changing and deformable user interfaces*. PhD thesis, Massachusetts Institute of Technology

Ultimately, these robots could become a part of our everyday environment, so that we are almost living with swarm robots (Figure 11.13). We still do not know how such a world looks like, but there is plenty of room to explore how these robots could augment and interact with our living environments. I hope this thesis provides inspiration to start exploring this idea in the future.

12

Conclusion

This thesis introduced dynamic and collective shape construction as a new form of interactive physical displays. This new type of display aims to expand the way we interact with digital information — by representing information as a three-dimensional physical object, it allows the user to touch, feel, grasp, and manipulate digital information through direct physical manipulation. Since it is dynamic, it also aims to capture and embody the dynamic computation of digital information — these shapes can be dynamically reconfigurable, so that it can reflect the real-time change of the underlying digital representation.

To achieve this goal, this thesis presented methods to construct, assemble, and transform the dynamic physical shape that consists of discrete active and passive elements. Each individual element can dynamically change its shape, position, and other physical properties through internal or external actuation, so that they can collectively update the overall shape in real-time. The dynamic shape that consists of discrete elements promises

the general-purpose shape-changing interfaces — in contrast to single-purpose shape transformation, it has potential for universal shape construction and transformation.

This thesis investigated this approach in the context of human-computer interaction and contributes in the following three ways:

First, I reviewed the methods of collective shape construction and identify opportunities to expand the current design space by exploring new ways of shape representation as well as new ways of combining the individual capability of active elements. Based on this exploration, this thesis demonstrated to expand the design space through a collective shape construction of active elements (Part I).

Second, this thesis showed the passive, externally-actuated elements can also make the dynamic shape construction by leveraging parallel and collective actuation. I expanded this design space and demonstrate it through new techniques and design architecture (Part II).

Third, this thesis explored a method and interaction technique to program the collective elements that construct an interactive shape. I demonstrate these techniques as a new way of tangible programming through direct physical manipulation (Part III).

Given these investigations, I discussed the design implications of these methods and what we can learn for the future investigations. As a conclusion, I discussed the potential research directions and opportunities for the future of dynamic and collective shape construction.

13

Bibliography

Africano, D., Berg, S., Lindbergh, K., Lundholm, P., Nilbrink, F., and Persson, A. (2004). Designing tangible interfaces for children's collaboration. In *CHI'04 extended abstracts on Human factors in computing systems*, pages 853–868.

Agrawal, H., Umapathi, U., Kovacs, R., Frohnhofen, J., Chen, H.-T., Mueller, S., and Baudisch, P. (2015). Protopiper: Physically sketching room-sized objects at actual scale. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 427–436.

Aish, R. (1979). 3d input for caad systems. *Computer-Aided Design*, 11(2):66–70.

Aish, R. and Noakes, P. (1984). Architecture without numbers—caad based on a 3d modelling system. *Computer-Aided Design*, 16(6):321–328.

Akiyama, H. and Yoshida, M. (2012). Photochemically reversible liquefaction and solidification of single compounds

based on a sugar alcohol scaffold with multi azo-arms. *Advanced Materials*, 24(17):2353–2356.

Alexander, J., Roudaut, A., Steimle, J., Hornbæk, K., Bruns Alonso, M., Follmer, S., and Merritt, T. (2018). Grand challenges in shape-changing interface research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 299. ACM.

Alonso-Mora, J., Breitenmoser, A., Rufli, M., Siegwart, R., and Beardsley, P. (2012). Image and animation display with multiple mobile robots. *The International Journal of Robotics Research*, 31(6):753–773.

Alves, R., Sousa, L., and Rodrigues, J. (2013). Poolliveaid: Augmented reality pool table to assist inexperienced players.

Andrews, C., Endert, A., and North, C. (2010). Space to think: large high-resolution displays for sensemaking. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 55–64.

Aronson, E. et al. (1978). *The jigsaw classroom.* SAGE Publications.

ART+COM (2008). Kinetic sculpture in bmw museum.

Asbeck, A. T., De Rossi, S. M., Holt, K. G., and Walsh, C. J. (2015). A biologically inspired soft exosuit for walking assistance. *The International Journal of Robotics Research*, 34(6):744–762.

Augugliaro, F., Lupashin, S., Hamer, M., Male, C., Hehn, M., Mueller, M. W., Willmann, J. S., Gramazio, F., Kohler, M., and D'Andrea, R. (2014). The flight assembled architecture installation: Cooperative construction with flying machines. *IEEE Control Systems Magazine*, 34(4):46–64.

Bailly, G., Sahdev, S., Malacria, S., and Pietrzak, T. (2016). Livingdesktop: Augmenting desktop workstation with actuated

devices. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 5298–5310.

Ball, R., North, C., and Bowman, D. A. (2007). Move to improve: promoting physical navigation to increase user performance with large displays. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 191–200.

Belke, C. H. and Paik, J. (2017). Mori: a modular origami robot. *IEEE/ASME Transactions on Mechatronics*, 22(5):2153–2164.

Benko, H., Holz, C., Sinclair, M., and Ofek, E. (2016). Normaltouch and texturetouch: High-fidelity 3d haptic shape rendering on handheld virtual reality controllers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 717–728. ACM.

Bimber, O. and Raskar, R. (2006). Modern approaches to augmented reality. In *ACM SIGGRAPH 2006 Courses*, pages 1–es.

Bondin, W., Mangion, F., and Glynn, R. (2015). Morphs 2.0.

Braley, S., Rubens, C., Merritt, T., and Vertegaal, R. (2018). Griddrones: A self-levitating physical voxel lattice for interactive 3d surface deformations. In *The 31st Annual ACM Symposium on User Interface Software and Technology*, pages 87–98. ACM.

Brand, S. (1995). *How buildings learn: What happens after they're built*. Penguin Publishing Group.

Cheng, L.-P., Roumen, T., Rantzsch, H., Köhler, S., Schmidt, P., Kovacs, R., Jasper, J., Kemper, J., and Baudisch, P. (2015). Turkdeck: Physical virtual reality based on people. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 417–426. ACM.

Cheung, K. C. and Gershenfeld, N. (2013). Reversibly assembled cellular composite materials. *science*, page 1240889.

Choi, I., Culbertson, H., Miller, M. R., Olwal, A., and Follmer, S. (2017). Grabity: A wearable haptic interface for simulating weight and grasping in virtual reality. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 119–130. ACM.

Coelho, M. and Maes, P. (2009). Shutters: a permeable surface for environmental control and communication. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, pages 13–18.

Coelho, M. and Zigelbaum, J. (2011). Shape-changing interfaces. *Personal and Ubiquitous Computing*, 15(2):161–173.

Cole, M. and Engeström, Y. (1993). A cultural-historical approach to distributed cognition. *Distributed cognitions: Psychological and educational considerations*, pages 1–46.

Council, N. R., Committee, G. S., et al. (2005). *Learning to think spatially*. National Academies Press.

Crespin, E. (2014). 4net inox quadra by atelier elias crespin.

Cruz-Neira, C., Sandin, D. J., and DeFanti, T. A. (1993). Surround-screen projection-based virtual reality: the design and implementation of the cave. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 135–142.

Czerwinski, M., Smith, G., Regan, T., Meyers, B., Robertson, G. G., and Starkweather, G. K. (2003). Toward characterizing the productivity benefits of very large displays. In *Interact*, volume 3, pages 9–16.

De Araùjo, B. R., Casiez, G., and Jorge, J. A. (2012). Mockup builder: Direct 3d modeling on and above the surface in a continuous interaction space. In *Proceedings of Graphics Interface 2012*, GI '12, pages 173–180, Toronto, Ont., Canada, Canada. Canadian Information Processing Society.

Dekan, M., Duchon, F., et al. (2013). irobot create used in education. *Journal of Mechanics Engineering and Automation*, 3(4):197–202.

Delazio, A., Nakagaki, K., Klatzky, R. L., Hudson, S. E., Lehman, J. F., and Sample, A. P. (2018). Force jacket: Pneumatically-actuated jacket for embodied haptic experiences. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 320. ACM.

Demaine, E. D. and Tachi, T. (2017). Origamizer: A practical algorithm for folding any polyhedron. In *33rd International Symposium on Computational Geometry (SoCG 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Dementyev, A., Hernandez, J., Follmer, S., Choi, I., and Paradiso, J. (2017). Skinbot: A wearable skin climbing robot. In *Adjunct Publication of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 5–6. ACM.

Dementyev, A., Kao, H.-L. C., Choi, I., Ajilo, D., Xu, M., Paradiso, J. A., Schmandt, C., and Follmer, S. (2016). Rovables: Miniature on-body robots as mobile wearables. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 111–120. ACM.

Do-Lenh, S., Jermann, P., Legge, A., Zufferey, G., and Dillenbourg, P. (2012). Tinkerlamp 2.0: designing and evaluating orchestration technologies for the classroom. In *European Conference on Technology Enhanced Learning*, pages 65–78. Springer.

Doorley, S., Witthoft, S., et al. (2012). *Make space: How to set the stage for creative collaboration*. John Wiley & Sons.

Dourish, P. (2004). *Where the action is: the foundations of embodied interaction*. MIT press.

Dunbar, K. (1995). How scientists really reason: Scientific reasoning in real-world laboratories. *The nature of insight*, 18:365–395.

Dunbar, K. (1997). How scientists think: On-line creativity and conceptual change in science.

Duncan, J. and Humphreys, G. W. (1989). Visual search and stimulus similarity. *Psychological review*, 96(3):433.

Everitt, A., Taher, F., and Alexander, J. (2016). Shapecanvas: an exploration of shape-changing content generation by members of the public. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 2778–2782.

Fabry, J. and Campusano, M. (2014). Live robot programming. In *Ibero-American Conference on Artificial Intelligence*, pages 445–456. Springer.

Fishkin, K. P. (2004). A taxonomy for and analysis of tangible interfaces. *Personal and Ubiquitous computing*, 8(5):347–358.

Fitzmaurice, G. W. (1997). *Graspable user interfaces*. PhD thesis.

Fitzmaurice, G. W., Ishii, H., and Buxton, W. A. (1995). Bricks: laying the foundations for graspable user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 442–449.

Follmer, S., Leithinger, D., Olwal, A., Cheng, N., and Ishii, H. (2012). Jamming user interfaces: programmable particle stiffness and sensing for malleable and shape-changing devices. In *Proceedings of the 25th annual ACM symposium on User Interface Software and Technology*, pages 519–528. ACM.

Follmer, S., Leithinger, D., Olwal, A., Hogge, A., and Ishii, H. (2013). inform: Dynamic physical affordances and constraints through shape and object actuation. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and*

Technology, UIST '13, pages 417–426, New York, NY, USA. ACM.

Follmer, S. S. W. (2015). *Dynamic physical affordances for shape-changing and deformable user interfaces*. PhD thesis, Massachusetts Institute of Technology.

Frazer, J. (1995). *An evolutionary architecture*. Architectural Association.

Frei, P., Su, V., Mikhak, B., and Ishii, H. (2000). Curlybot: designing a new class of computational toys. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 129–136.

Fujii, J., Matsunobu, T., and Kakehi, Y. (2018). Colorise: Shape- and color-changing pixels with inflatable elastomers and interactions. In *Proceedings of the Twelfth International Conference on Tangible, Embedded, and Embodied Interaction*, pages 199–204. ACM.

Fujinami, K., Kawsar, F., and Nakajima, T. (2005). Awaremirror: a personalized display using a mirror. In *International Conference on Pervasive Computing*, pages 315–332. Springer.

Furió, D., Fleck, S., Bousquet, B., Guillet, J.-P., Canioni, L., and Hachet, M. (2017). Hobit: Hybrid optical bench for innovative teaching. In *Proceedings of the 2017 chi conference on human factors in computing systems*, pages 949–959.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F. J., and Marín-Jiménez, M. J. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292.

Gershenfeld, N. (1999). *When things start to think*. Henry Holt and Co., Inc.

Gertner, J. (2012). *The idea factory: Bell Labs and the great age of American innovation*. Penguin Publishing Group.

Gibson, J. J. (1979). *The ecological approach to visual perception: classic edition*. Psychology Press.

Gilpin, K., Knaian, A., and Rus, D. (2010). Robot pebbles: One centimeter modules for programmable matter through self-disassembly. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2485–2492. IEEE.

Goldstein, S. C., Campbell, J. D., and Mowry, T. C. (2005). Programmable matter. *Computer*, 38(6):99–101.

Goldstein, S. C. and Mowry, T. C. (2004). Claytronics: A scalable basis for future robots.

Goulthorpe, M. (2006). Aegis hyposurface.

Green, K. E. (2016). *Architectural robotics: ecosystems of bits, bytes, and biology*. MIT Press.

Griffith, S. T. (2004). *Growing machines*. PhD thesis, Massachusetts Institute of Technology.

Grönvall, E., Kinch, S., Petersen, M. G., and Rasmussen, M. K. (2014). Causing commotion with a shape-changing bench: experiencing shape-changing interfaces in use. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 2559–2568. ACM.

Gross, M. D. and Green, K. E. (2012). Architectural robotics, inevitably. *interactions*, 19(1):28–33.

Guinness, D., Muehlbradt, A., Szafir, D., and Kane, S. K. (2018). The haptic video player: Using mobile robots to create tangible video annotations. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces*, pages 203–211. ACM.

Guizzo, E. (2008). Three engineers, hundreds of robots, one warehouse. *IEEE spectrum*, 45(7):26–34.

Guseinov, R., Miguel, E., and Bickel, B. (2017). Curveups: Shaping objects from flat plates with tension-actuated curvature. *ACM Transactions on Graphics (TOG)*, 36(4):1–12.

Hammond, Z. M., Usevitch, N. S., Hawkes, E. W., and Follmer, S. (2017). Pneumatic reel actuator: Design, modeling, and implementation. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 626–633. IEEE.

Hardy, J., Weichel, C., Taher, F., Vidler, J., and Alexander, J. (2015). Shapeclip: towards rapid prototyping with shape-changing displays for designers. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 19–28. ACM.

Harrison, C., Benko, H., and Wilson, A. D. (2011). Omnitouch: wearable multitouch interaction everywhere. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 441–450.

Harrison, C., Tan, D., and Morris, D. (2010). Skinput: appropriating the body as an input surface. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–462.

Hawkes, E., An, B., Benbernou, N. M., Tanaka, H., Kim, S., Demaine, E., Rus, D., and Wood, R. J. (2010). Programmable matter by folding. *Proceedings of the National Academy of Sciences*, 107(28):12441–12445.

Hawkes, E. W., Blumenschein, L. H., Greer, J. D., and Okamura, A. M. (2017). A soft robot that navigates its environment through growth. *Science Robotics*, 2(8):eaan3028.

He, L., Laput, G., Brockmeyer, E., and Froehlich, J. E. (2017a). Squeezapulse: Adding interactive input to fabricated objects using corrugated tubes and air pulses. In *Proceedings of the Eleventh International Conference on Tangible, Embedded, and Embodied Interaction*, pages 341–350. ACM.

He, Z., Zhu, F., and Perlin, K. (2017b). Physhare: sharing physical interaction in virtual reality. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 17–19. ACM.

Held, R., Gupta, A., Curless, B., and Agrawala, M. (2012). 3d puppetry: a kinect-based interface for 3d animation. In *UIST*, pages 423–434. Citeseer.

Hemmert, F., Hamann, S., Löwe, M., Zeipelt, J., and Joost, G. (2010). Shape-changing mobiles: tapering in two-dimensional deformational displays in mobile phones. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*, pages 3075–3080.

Hiller, J. (2011). Digital materials: Voxel design, rapid assembly, structural properties, and design methods.

Hiller, J. D. and Lipson, H. (2009). Fully recyclable multi-material printing. In *Solid Freeform Fabrication Proceedings*, pages 98–106. Citeseer.

Hinckley, K., Pausch, R., Goble, J. C., and Kassell, N. F. (1994). Passive real-world interface props for neurosurgical visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 452–458.

Hoang, T., Reinoso, M., Joukhadar, Z., Vetere, F., and Kelly, D. (2017). Augmented studio: projection mapping on moving body for physiotherapy education. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 1419–1430.

Hollan, J., Hutchins, E., and Kirsh, D. (2000). Distributed cognition: toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(2):174–196.

Horn, M. S., Solovey, E. T., Crouser, R. J., and Jacob, R. J. (2009). Comparing the use of tangible and graphical programming languages for informal science education. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 975–984, New York, NY, USA. ACM.

Hutchins, E. (1995). *Cognition in the Wild*. Number 1995. MIT press.

Ibayashi, H., Sugiura, Y., Sakamoto, D., Miyata, N., Tada, M., Okuma, T., Kurata, T., Mochimaru, M., and Igarashi, T. (2015). Dollhouse vr: a multi-view, multi-user collaborative design workspace with vr technology. In *SIGGRAPH Asia 2015 Emerging Technologies*, page 8. ACM.

Ion, A., Frohnhofen, J., Wall, L., Kovacs, R., Alistar, M., Lindsay, J., Lopes, P., Chen, H.-T., and Baudisch, P. (2016). Metamaterial mechanisms. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 529–539.

Ishii, H. (2004). Bottles: A transparent interface as a tribute to mark weiser. *IEICE Transactions on information and systems*, 87(6):1299–1311.

Ishii, H., Lakatos, D., Bonanni, L., and Labrune, J.-B. (2012). Radical atoms: Beyond tangible bits, toward transformable materials. *Interactions*.

Ishii, H., Ratti, C., Piper, B., Wang, Y., Biderman, A., and Ben-Joseph, E. (2004). Bringing clay and sand into digital design—continuous tangible user interfaces. *BT technology journal*, 22(4):287–299.

Ishii, H., Ren, S., and Frei, P. (2001). Pinwheels: visualizing information flow in an architectural space. In *CHI'01 extended abstracts on Human factors in computing systems*, pages 111–112.

Ishii, H. and Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, pages 234–241. ACM.

Ishii, H., Wisneski, C., Brave, S., Dahley, A., Gorbet, M., Ullmer, B., and Yarin, P. (1998). ambientroom: integrating ambient media with architectural space. In *CHI 98 conference summary on Human factors in computing systems*, pages 173–174.

Ishii, H., Wisneski, C., Orbanes, J., Chun, B., and Paradiso, J. (1999). Pingpongplus: design of an athletic-tangible interface for computer-supported cooperative play. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 394–401.

Iwata, H., Yano, H., Fukushima, H., and Noma, H. (2005). Circulafloor [locomotion interface]. *IEEE Computer Graphics and Applications*, 25(1):64–67.

Iwata, H., Yano, H., Nakaizumi, F., and Kawamura, R. (2001). Project feelex: adding haptic surface to graphics. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 469–476. ACM.

Iyengar, S. (2010). *The art of choosing*. Twelve.

Jacobson, A., Panozzo, D., Glauser, O., Pradalier, C., Hilliges, O., and Sorkine-Hornung, O. (2014). Tangible and modular input device for character articulation. *ACM Transactions on Graphics (TOG)*, 33(4):1–12.

Jang, S., Kim, L. H., Tanner, K., Ishii, H., and Follmer, S. (2016). Haptic edge display for mobile tactile interaction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 3706–3716.

Jansen, Y. (2014). *Physical and tangible information visualization*. PhD thesis.

Jansen, Y., Dragicevic, P., Isenberg, P., Alexander, J., Karnik, A., Kildal, J., Subramanian, S., and Hornbæk, K. (2015). Opportunities and challenges for data physicalization. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3227–3236. ACM.

Jenett, B. and Cheung, K. (2017). Bill-e: Robotic platform for locomotion and manipulation of lightweight space structures. In *25th AIAA/AHS Adaptive Structures Conference*, page 1876.

Jensen, F. and Pellegrino, S. (2001). Arm development review of existing technologies. Technical report, Cambridge Univ.(United Kingdom). Dept. of Engineering.

Jones, B. and Sohdi, R. (2012). The illustrated history of projection mapping.

Joosten, B. K. (2007). Preliminary assessment of artificial gravity impacts to deep-space vehicle design.

Jordà, S., Geiger, G., Alonso, M., and Kaltenbrunner, M. (2007). The reactable: exploring the synergy between live music performance and tabletop tangible interfaces. In *Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 139–146.

Kajastila, R., Holsti, L., and Hämäläinen, P. (2016). The augmented climbing wall: High-exertion proximity interaction on a wall-sized interactive surface. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 758–769.

Karagozler, M. E., Goldstein, S. C., and Reid, J. R. (2009). Stress-driven mems assembly+ electrostatic forces= 1mm diameter robot. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS '09, pages 2763–2769. IEEE.

Kay, A. and Goldberg, A. (1977). Personal dynamic media. *Computer*, 10(3):31–41.

Kay, A. C. (2011). A personal computer for children of all ages. In *Proceedings of the ACM annual conference-Volume 1*.

Khan, A. (2014). Megafaces.

Khan, O. (2010). Open columns: a carbon dioxide (co2) responsive architecture. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*, pages 4789–4792. ACM.

Kim, L. H. and Follmer, S. (2017). Ubiswarm: Ubiquitous robotic interfaces and investigation of abstract motion as a display. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3):66.

Kim, S., Kim, H., Lee, B., Nam, T.-J., and Lee, W. (2008). Inflatable mouse: volume-adjustable mouse with air-pressure-sensitive input and haptic feedback. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 211–224.

Kirsh, D. (1995). The intelligent use of space. *Artificial Intelligence*, pages 31–68.

Kirsh, D. (1996). Adapting the environment instead of oneself. *Adaptive Behavior*, 4(3-4):415–452.

Kirsh, D. (2010). Thinking with external representations. *AI & society*, 25(4):441–454.

Klatzky, R. L. and Lederman, S. (1990). Intelligent exploration by the human hand. In *Dextrous robot hands*, pages 66–81. Springer.

Klemmer, S. R., Hartmann, B., and Takayama, L. (2006). How bodies matter: five themes for interaction design. In *Proceedings of the 6th conference on Designing Interactive systems*, pages 140–149.

Knaian, A. N. (2010). *Electropermanent magnetic connectors and actuators: devices and their application in programmable matter*. PhD thesis, Massachusetts Institute of Technology.

Ladd, C., So, J.-H., Muth, J., and Dickey, M. D. (2013). 3d printing of free standing liquid metal microstructures. *Advanced Materials*, 25(36):5081–5085.

Langbehn, E., Lubos, P., and Steinicke, F. (2018). Evaluation of locomotion techniques for room-scale vr: Joystick, teleportation, and redirected walking. In *Proceedings of the Virtual Reality International Conference-Laval Virtual*, page 4. ACM.

Larkin, J. H. and Simon, H. A. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive science*, 11(1):65–100.

Le Goc, M., Kim, L. H., Parsaei, A., Fekete, J.-D., Dragicevic, P., and Follmer, S. (2016). Zooids: Building blocks for swarm user interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 97–109. ACM.

Le Goc, M., Perin, C., Follmer, S., Fekete, J.-D., and Dragicevic, P. (2019). Dynamic composite data physicalization using wheeled micro-robots. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):737–747.

Leder, S. and Weber, R. (2018). Itech m.sc. 2018: Distributed robotic assembly system for in-situ timber construction. Master's thesis, University of Stuttgart.

Leder, S., Weber, R., Bucklin, O., Wood, D., and Menges, A. (2019). Towards distributed in-situ robotic timber construction. In *Research Culture in Architecture*. Birkhäuser Verlag.

Lee, H., Lee, B. P., and Messersmith, P. B. (2007). A reversible wet/dry adhesive inspired by mussels and geckos. *Nature*, 448(7151):338.

Lee, J., Post, R., and Ishii, H. (2011a). Zeron: mid-air tangible interaction enabled by computer controlled magnetic levitation. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 327–336.

Lee, N., Kim, J., Lee, J., Shin, M., and Lee, W. (2011b). Molebot: mole in a table. In *ACM SIGGRAPH 2011 Emerging Technologies*, pages 1–1.

Leithinger, D. (2015). *Grasping information and collaborating through shape displays*. PhD thesis, Massachusetts Institute of Technology.

Leithinger, D., Follmer, S., Olwal, A., and Ishii, H. (2014). Physical telepresence: shape capture and display for embodied, computer-mediated remote collaboration. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 461–470. ACM.

Leithinger, D., Follmer, S., Olwal, A., Luescher, S., Hogge, A., Lee, J., and Ishii, H. (2013). Sublimate: state-changing virtual and physical rendering to augment interaction with shape displays. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1441–1450. ACM.

Leithinger, D. and Ishii, H. (2010). Relief: a scalable actuated shape display. In *Proceedings of the fourth international conference on Tangible, Embedded, and Embodied Interaction*, pages 221–222. ACM.

Leithinger, D., Lakatos, D., DeVincenzi, A., Blackshaw, M., and Ishii, H. (2011). Direct and gestural interaction with relief: A 2.5d shape display. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 541–548, New York, NY, USA. ACM.

Liang, R.-H., Cheng, K.-Y., Su, C.-H., Weng, C.-T., Chen, B.-Y., and Yang, D.-N. (2012). Gausssense: Attachable stylus sensing using magnetic sensor grid. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 319–326, New York, NY, USA. ACM.

Licklider, J. C. (1960). Man-computer symbiosis. *IRE transactions on human factors in electronics*, (1):4–11.

Liu, C., Chapuis, O., Beaudouin-Lafon, M., Lecolinet, E., and Mackay, W. E. (2014). Effects of display size and navigation type on a classification task. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 4147–4156.

MacCurdy, R., Katzschmann, R., Kim, Y., and Rus, D. (2016). Printable hydraulics: a method for fabricating robots by 3d co-printing solids and liquids. In *Proceedings of ICRA*, pages 3878–3885. IEEE.

Maeda, Y., Nakano, O., Maekawa, T., and Maruo, S. (2016). From cad models to toy brick sculptures: A 3d block printer. In *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS '16, pages 2167–2172. IEEE.

Marshall, M., Carter, T., Alexander, J., and Subramanian, S. (2012). Ultra-tangibles: creating movable tangible objects on interactive tables. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2185–2188.

Meng, M., Fallavollita, P., Blum, T., Eck, U., Sandor, C., Weidert, S., Waschke, J., and Navab, N. (2013). Kinect for interactive ar anatomy learning. In *2013 IEEE international symposium on mixed and augmented reality (ISMAR)*, pages 277–278. IEEE.

Mitchell, W. J. (1999). *e-topia:" Urban life, Jim—but not as we know it"*. MIT press.

Mueller, S., Mohr, T., Guenther, K., Frohnhofen, J., and Baudisch, P. (2014). fabrickation: Fast 3d printing of functional objects by integrating construction kit building blocks. In

Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '14, pages 3827–3834, New York, NY, USA. ACM.

Murata, S., Yoshida, E., Kamimura, A., Kurokawa, H., Tomita, K., and Kokaji, S. (2002). M-tran: Self-reconfigurable modular robotic system. *IEEE/ASME transactions on mechatronics*, 7(4):431–441.

Nakagaki, K., Dementyev, A., Follmer, S., Paradiso, J. A., and Ishii, H. (2016). Chainform: a linear integrated modular hardware system for shape changing interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 87–96. ACM.

Nakagaki, K., Fitzgerald, D., Ma, Z., Vink, L., Levine, D., and Ishii, H. (2019). inforce: Bi-directionalforce'shape display for haptic interaction. In *Proceedings of the Thirteenth International Conference on Tangible, Embedded, and Embodied Interaction*, pages 615–623.

Nakagaki, K., Follmer, S., and Ishii, H. (2015). Lineform: Actuated curve interfaces for display, interaction, and constraint. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 333–339. ACM.

Nakagaki, K., Umapathi, U., Leithinger, D., and Ishii, H. (2017). Animastage: Hands-on animated craft on pin-based shape displays. In *Proceedings of the 2017 Conference on Designing Interactive Systems*, DIS '17, pages 1093–1097, New York, NY, USA. ACM.

Nakamaru, S., Nakayama, R., Niiyama, R., and Kakehi, Y. (2017). Foamsense: Design of three dimensional soft sensors with porous materials. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 437–447. ACM.

Neubert, J., Rost, A., and Lipson, H. (2014). Self-soldering connectors for modular robots. *IEEE Transactions on Robotics*, 30(6):1344–1357.

Niiyama, R., Sun, X., Yao, L., Ishii, H., Rus, D., and Kim, S. (2015). Sticky actuator: Free-form planar actuators for animated objects. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction*, pages 77–84. ACM.

Niiyama, R., Yao, L., and Ishii, H. (2014). Weight and volume changing device with liquid metal transfer. In *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction*, pages 49–52.

Nilsson, M. (2002). Heavy-duty connectors for self-reconfiguring robots. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, volume 4 of *ICRA '02*, pages 4071–4076. IEEE.

Norman, D. A. (1993). Cognition in the head and in the world: An introduction to the special issue on situated action. *Cognitive science*, 17(1):1–6.

Norman, D. A. (1999). Affordance, conventions, and design. *interactions*, 6(3):38–43.

Norman, D. A. (2013). *The design of everyday things: Revised and expanded edition*. Basic books.

Northen, M. T., Greiner, C., Arzt, E., and Turner, K. L. (2008). A gecko-inspired reversible adhesive. *Advanced Materials*, 20(20):3905–3909.

Nowacka, D., Ladha, K., Hammerla, N. Y., Jackson, D., Ladha, C., Rukzio, E., and Olivier, P. (2013). Touchbugs: Actuated tangibles on multi-touch tables. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 759–762.

Ochiai, Y. (2015). *Graphics by Computational Acoustic Fields*. PhD thesis, The University of Tokyo.

Ochiai, Y., Hoshi, T., and Rekimoto, J. (2014). Pixie dust: graphics generated by levitated and animated objects in computational acoustic-potential field. *ACM Transactions on Graphics (TOG)*, 33(4):1–13.

Olwal, A. and Wilson, A. D. (2008). Surfacefusion: unobtrusive tracking of everyday objects in tangible user interfaces. In *Proceedings of graphics interface 2008*, pages 235–242.

Omirou, T., Marzo, A., Seah, S. A., and Subramanian, S. (2015). Levipath: Modular acoustic levitation for 3d path visualisations. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 309–312.

Oosterhuis, K. and Biloria, N. (2008). Interactions with proactive architectural spaces: the muscle projects. *Communications of the ACM*, 51(6):70–78.

Oppenheimer, F. (1972). The exploratorium: A playful museum combines perception and art in science education. *American Journal of Physics*, 40(7):978–984.

Ou, J., Ma, Z., Peters, J., Dai, S., Vlavianos, N., and Ishii, H. (2018). Kinetix-designing auxetic-inspired deformable material structures. *Computers & Graphics*, 75:72–81.

Ou, J., Skouras, M., Vlavianos, N., Heibeck, F., Cheng, C.-Y., Peters, J., and Ishii, H. (2016). aeromorph-heat-sealing inflatable shape-change materials for interaction design. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 121–132. ACM.

Özgür, A., Lemaignan, S., Johal, W., Beltran, M., Briod, M., Pereyre, L., Mondada, F., and Dillenbourg, P. (2017). Cellulo: Versatile handheld robots for education. In *2017 12th*

ACM/IEEE International Conference on Human-Robot Interaction (HRI, pages 119–127. IEEE.

Papadopoulou, A., Laucks, J., and Tibbits, S. (2017). From self-assembly to evolutionary structures. *Architectural Design*, 87(4):28–37.

Papert, S. (1980). Mindstorms: Computers, children, and powerful ideas. *NY: Basic Books*.

Papert, S. and Harel, I. (1991). Situating constructionism. *Constructionism*, 36(2):1–11.

Patten, J. (2014). Thumbles-robotic tabletop user interface platform. *TED. com*.

Patten, J. and Ishii, H. (2007). Mechanical constraints as computational constraints in tabletop tangible interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 809–818. ACM.

Patten, J., Ishii, H., Hines, J., and Pangaro, G. (2001). Sensetable: a wireless object tracking platform for tangible user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 253–260.

Pea, R. D. (1993). Practices of distributed intelligence and designs for education. *Distributed cognitions: Psychological and educational considerations*, 11:47–87.

Pece, F., Zarate, J. J., Vechev, V., Besse, N., Gudozhnik, O., Shea, H., and Hilliges, O. (2017). Magtics: Flexible and thin form factor magnetic actuators for dynamic and wearable haptic feedback. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pages 143–154, New York, NY, USA. ACM.

Peters, B. J. (2011). *Design and fabrication of a digitally reconfigurable surface*. PhD thesis, Massachusetts Institute of Technology.

Petersen, K. H., Nagpal, R., and Werfel, J. K. (2011). Termes: An autonomous robotic system for three-dimensional collective construction. *Robotics: science and systems VII*.

Piper, B., Ratti, C., and Ishii, H. (2002). Illuminating clay: a 3-d tangible interface for landscape analysis. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 355–362.

Playfair, W. (2005). *Playfair's commercial and political atlas and statistical breviary*. Cambridge University Press.

Popescu, G. A., Mahale, T., and Gershenfeld, N. (2006). Digital materials for digital printing. In *NIP & Digital Fabrication Conference*, volume 2006, pages 58–61. Society for Imaging Science and Technology.

Poupyrev, I., Nashida, T., Maruyama, S., Rekimoto, J., and Yamaji, Y. (2004). Lumen: interactive visual and shape display for calm computing. In *ACM SIGGRAPH 2004 Emerging technologies*, page 17. ACM.

Poupyrev, I., Nashida, T., and Okabe, M. (2007). Actuation and tangible user interfaces: the vaucanson duck, robots, and shape displays. In *Proceedings of the 1st International Conference on Tangible and embedded interaction*, pages 205–212. ACM.

Raffle, H. S., Parkes, A. J., and Ishii, H. (2004). Topobo: a constructive assembly system with kinetic memory. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 647–654.

Raskar, R., Beardsley, P., van Baar, J., Wang, Y., Dietz, P., Lee, J., Leigh, D., and Willwacher, T. (2004). Rfig lamps: interacting with a self-describing world via photosensing wireless tags and projectors. In *ACM SIGGRAPH 2004 Papers*, pages 406–415.

Raskar, R., Welch, G., Cutts, M., Lake, A., Stesin, L., and Fuchs, H. (1998). The office of the future: A unified approach to image-based modeling and spatially immersive displays. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 179–188.

Rasmussen, M. K., Pedersen, E. W., Petersen, M. G., and Hornbæk, K. (2012). Shape-changing interfaces: a review of the design space and open research questions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 735–744. ACM.

Ratti, C. (2016). Lift-bit designed by carlo ratti associati.

Rekimoto, J. (2012). Squama: modular visibility control of walls and windows for programmable physical architectures. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 168–171.

Rekimoto, J. and Saitoh, M. (1999). Augmented surfaces: a spatially continuous work space for hybrid computing environments. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 378–385.

Resnick, M., Martin, F., Berg, R., Borovoy, R., Colella, V., Kramer, K., and Silverman, B. (1998). Digital manipulatives: new toys to think with. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–287. ACM Press/Addison-Wesley Publishing Co.

Rheingold, H. (2000). *Tools for thought: The history and future of mind-expanding technology*. MIT Press.

Ricquebourg, V., Menga, D., Durand, D., Marhic, B., Delahoche, L., and Loge, C. (2006). The smart home concept: our immediate future. In *2006 1st IEEE international conference on e-learning in industrial electronics*, pages 23–28. IEEE.

Romanishin, J. W., Gilpin, K., and Rus, D. (2013). M-blocks: Momentum-driven, magnetic modular robots. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 4288–4295. IEEE.

Rønne Jakobsen, M. and Hornbæk, K. (2011). Sizing up visualizations: effects of display size in focus+ context, overview+ detail, and zooming interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1451–1460.

Roudaut, A., Karnik, A., Löchtefeld, M., and Subramanian, S. (2013). Morphees: toward high shape resolution in self-actuated flexible mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 593–602. ACM.

Roudaut, A., Krusteva, D., McCoy, M., Karnik, A., Ramani, K., and Subramanian, S. (2016). Cubimorph: designing modular interactive devices. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3339–3345. IEEE.

Roudaut, A., Reed, R., Hao, T., and Subramanian, S. (2014). Changibles: analyzing and designing shape changing constructive assembly. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2593–2596. ACM.

Rubenstein, M., Ahler, C., and Nagpal, R. (2012). Kilobot: A low cost scalable robot system for collective behaviors. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3293–3298. IEEE.

Rubenstein, M., Cornejo, A., and Nagpal, R. (2014). Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799.

Saito, S., Nobusue, S., Tsuzaka, E., Yuan, C., Mori, C., Hara, M., Seki, T., Camacho, C., Irle, S., and Yamaguchi, S. (2016).

Light-melt adhesive based on dynamic carbon frameworks in a columnar liquid-crystal phase. *Nature communications*, 7:12094.

Salomon, G. (1993). No distribution without individuals' cognition: A dynamic interactional view. *Distributed cognitions: Psychological and educational considerations*, pages 111–138.

Samimi, H., Warth, A., Eslamimehr, M., and Borning, A. (2015). Constraints as a design pattern. In *2015 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software (Onward!)*, Onward! 2015, pages 28–43, New York, NY, USA. ACM.

Scaife, M. and Rogers, Y. (1996). External cognition: how do graphical representations work? *International journal of human-computer studies*, 45(2):185–213.

Schachman, T. (2015). Apparatus. *URL: http://aprt.us/*.

Schnädelbach, H., Glover, K., and Irune, A. A. (2010). Exobuilding: breathing life into architecture. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, pages 442–451. ACM.

Schoessler, P. (2015). Shape synthesis: physical object augmentation and actuation for display and interaction on shape changing interfaces. Master's thesis, Massachusetts Institute of Technology.

Schoessler, P., Windham, D., Leithinger, D., Follmer, S., and Ishii, H. (2015). Kinetic blocks: Actuated constructive assembly for interaction and display. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 341–349. ACM.

Schrage, M. (1996). Cultures of prototyping. *Bringing design to software*, pages 191–205.

Sekijima, K. and Tanaka, H. (2015). Reconfigurable three-dimensional prototype system using digital materials. In *ACM SIGGRAPH 2015 Posters*, SIGGRAPH '15, pages 89:1–89:1, New York, NY, USA. ACM.

Shaer, O., Leland, N., Calvillo-Gamez, E. H., and Jacob, R. J. (2004). The tac paradigm: specifying tangible user interfaces. *Personal and Ubiquitous Computing*, 8(5):359–369.

Shneiderman, B. (1993). Direct manipulation: a step beyond programming languages. *Sparks of innovation in human-computer interaction*, 17:1993.

Siu, A. F., Gonzalez, E. J., Yuan, S., Ginsberg, J. B., and Follmer, S. (2018). Shapeshift: 2d spatial manipulation and self-actuation of tabletop shape displays for tangible and haptic interaction. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 291. ACM.

Smith, S. M., Ward, T. B., and Schumacher, J. S. (1993). Constraining effects of examples in a creative generation task. *Memory & cognition*, 21(6):837–845.

Snow, J. (1855). *On the mode of communication of cholera*. John Churchill.

Sproewitz, A., Billard, A., Dillenbourg, P., and Ijspeert, A. J. (2009). Roombots-mechanical design of self-reconfiguring modular robots for adaptive furniture. In *2009 IEEE international conference on robotics and automation*, pages 4259–4264. IEEE.

Sprowitz, A., Pouya, S., Bonardi, S., Van Den Kieboom, J., Mockel, R., Billard, A., Dillenbourg, P., and Ijspeert, A. J. (2010). Roombots: reconfigurable robots for adaptive furniture. *IEEE Computational Intelligence Magazine*, 5(3):20–32.

Staal, B. G. (2015). Irregular polyhedron study 1 by bjørn gunnar staal.

Stone, R. J. (2000). Haptic feedback: A brief history from telepresence to virtual reality. In *International Workshop on Haptic Human-Computer Interaction*, pages 1–16. Springer.

Strasnick, E., Yang, J., Tanner, K., Olwal, A., and Follmer, S. (2017). shiftio: Reconfigurable tactile elements for dynamic affordances and mobile interaction. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 5075–5086. ACM.

StretchDesign (2005). Flexible love.

Sturdee, M., Hardy, J., Dunn, N., and Alexander, J. (2015). A public ideation of shape-changing applications. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*, pages 219–228. ACM.

Sutherland, I. E. (1964). Sketchpad a man-machine graphical communication system. *Transactions of the Society for Computer Simulation*, 2(5):R–3.

Sutherland, I. E. (1965). The ultimate display. *Multimedia: From Wagner to virtual reality*, pages 506–508.

Suzuki, H. and Kato, H. (1995). Interaction-level support for collaborative learning: Algoblock—an open programming language. In *The First International Conference on Computer Support for Collaborative Learning*, CSCL '95, pages 349–355, Hillsdale, NJ, USA. L. Erlbaum Associates Inc.

Suzuki, R. (2019). Collective shape-changing interfaces. In *The Adjunct Publication of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19. ACM.

Suzuki, R., Hedayati, H., Bohn, J., Zheng, C., Do, E. Y.-L., Szafir, D., and Leithinger, D. (2020a). Roomshift: Room-scale dynamic haptics for vr with furniture-moving swarm robots. In *Proceedings of the CHI '20*, pages 199:1–199:13.

Suzuki, R., Kato, J., Gross, M. D., and Yeh, T. (2018a). Reactile: Programming swarm user interfaces through direct physical manipulation. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 199. ACM.

Suzuki, R., Nakayama, R., Liu, D., Kakehi, Y., Gross, M. D., and Leithinger, D. (2019a). Lifttiles: Modular and reconfigurable room-scale shape displays through retractable inflatable actuators. In *Adjunct Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM.

Suzuki, R., Nakayama, R., Liu, D., Kakehi, Y., Gross, M. D., and Leithinger, D. (2020b). Lifttiles: Constructive building blocks for prototyping room-scale shape-changing interfaces. In *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction*. ACM.

Suzuki, R., Stangl, A., Gross, M. D., and Yeh, T. (2017). Fluxmarker: Enhancing tactile graphics with dynamic tactile markers. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, pages 190–199. ACM.

Suzuki, R., Yamaoka, J., Leithinger, D., Yeh, T., Gross, M. D., Kawahara, Y., and Kakehi, Y. (2018b). Dynablock: Dynamic 3d printing for instant and reconstructable shape formation. In *The 31st Annual ACM Symposium on User Interface Software and Technology*, pages 99–111. ACM.

Suzuki, R., Zheng, Clement Kakehi, Y., Yeh, T., Yi-Luen Do, E., Gross, M. D., and Leithinger, D. (2019b). Shapebots: Shape-changing swarm robots. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM.

Swaminathan, S., Rivera, M., Kang, R., Luo, Z., Ozutemiz, K. B., and Hudson, S. E. (2019). Input, output and construction methods for custom fabrication of room-scale deploy-

able pneumatic structures. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(2):1–17.

Taher, F., Hardy, J., Karnik, A., Weichel, C., Jansen, Y., Hornbæk, K., and Alexander, J. (2015). Exploring interactions with physically dynamic bar charts. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3237–3246.

Takei, S., Iida, M., and Naemura, T. (2011). Kinereels: extension actuators for dynamic 3d shape. In *ACM SIGGRAPH 2011 Posters*, page 84. ACM.

Takei, S., Iida, M., and Naemura, T. (2012). Morphys: Morphing physical environment using extension actuators. In *ACM SIGGRAPH*.

Tang, W. Y., Tang, S. K., and Lee, Y. Z. (2011). Tangible pixels.

Teng, S.-Y., Kuo, T.-S., Wang, C., Chiang, C.-h., Huang, D.-Y., Chan, L., and Chen, B.-Y. (2018). Pupop: Pop-up prop on palm for virtual reality. In *The 31st Annual ACM Symposium on User Interface Software and Technology*, pages 5–17. ACM.

Teng, S.-Y., Lin, C.-L., Chiang, C.-h., Kuo, T.-S., Chan, L., Huang, D.-Y., and Chen, B.-Y. (2019). Tilepop: Tile-type pop-up prop for virtual reality. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM.

Tibbits, S. (2014). 4d printing: multi-material shape change. *Architectural Design*, 84(1):116–121.

Tibbits, S., McKnelly, C., Papadopoulou, A., Martin, C., Guberan, C., Zuniga, B., and Nam, H. A. (2014). Aerial assembly.

Toffoli, T. and Margolus, N. (1991). Programmable matter: concepts and realization. *Physica. D, Nonlinear phenomena*, 47(1-2):263–272.

Togler, J., Hemmert, F., and Wettach, R. (2009). Living interfaces: the thrifty faucet. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, pages 43–44.

Torres, B. (2014). Itech m.sc. 2014: Programmable matter. Master's thesis, University of Stuttgart.

Turkle, S. E. (2007). *Evocative objects: Things we think with.* MIT press.

Ullmer, B. and Ishii, H. (2000). Emerging frameworks for tangible user interfaces. *IBM systems journal*, 39(3.4):915–931.

Ullmer, B., Ishii, H., and Glas, D. (1998). mediablocks: physical containers, transports, and controls for online media. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 379–386.

Ullmer, B., Ishii, H., and Jacob, R. J. (2005). Token+ constraint systems for tangible interaction with digital information. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 12(1):81–118.

Underkoffler, J. and Ishii, H. (1999). Urp: a luminous-tangible workbench for urban planning and design. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 386–393.

Usevitch, N. S., Hammond, Z. M., Schwager, M., Okamura, A. M., Hawkes, E. W., and Follmer, S. (2020). An untethered isoperimetric soft robot. *Science Robotics*, 5(40).

Vallgårda, A. (2014). The dress room: responsive spaces and embodied interaction. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*, pages 618–627. ACM.

Victor, B. (2011). A brief rant on the future of interaction design.

Victor, B. (2013). Media for thinking the unthinkable. *Vimeo, May*.

Victor, B. (2014a). Humane representation of thought: A trail map for the 21st century.

Victor, B. (2014b). Seeing spaces. In *Talk at EG conference*.

Victor, B. et al. (2018). Dynamicland.

Vink, L., Kan, V., Nakagaki, K., Leithinger, D., Follmer, S., Schoessler, P., Zoran, A., and Ishii, H. (2015). Transform as adaptive and dynamic furniture. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pages 183–183.

Wang, L. and Liu, J. (2014). Liquid phase 3d printing for quickly manufacturing conductive metal objects with low melting point alloy ink. *Science China Technological Sciences*, 57(9):1721–1728.

Weichel, C., Alexander, J., Karnik, A., and Gellersen, H. (2015). Spata: Spatio-tangible tools for fabrication-aware design. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction*, pages 189–196. ACM.

Weichel, C., Lau, M., Kim, D., Villar, N., and Gellersen, H. W. (2014). Mixfab: a mixed-reality environment for personal fabrication. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3855–3864. ACM.

Weiser, M. (1991). The computer for the 21 st century. *Scientific American*, 265(3):94–105.

Weiss, M., Schwarz, F., Jakubowski, S., and Borchers, J. (2010). Madgets: actuating widgets on interactive tabletops. In *Proceedings of the 23nd annual ACM symposium on User Interface software and technology*, pages 293–302. ACM.

Weller, M. P., Do, E. Y.-L., and Gross, M. D. (2008). Posey: instrumenting a poseable hub and strut construction toy. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 39–46.

Wellner, P. (1993). Interacting with paper on the digitaldesk. *Communications of the ACM*, 36(7):87–96.

Werfel, J. and Nagpal, R. (2008). Three-dimensional construction with mobile robots and modular blocks. *The International Journal of Robotics Research*, 27(3-4):463–479.

Whitesides, G. M. and Grzybowski, B. (2002). Self-assembly at all scales. *Science*, 295(5564):2418–2421.

Willett, W., Jansen, Y., and Dragicevic, P. (2016). Embedded data representations. *IEEE transactions on visualization and computer graphics*, 23(1):461–470.

Willis, K. D., Poupyrev, I., Hudson, S. E., and Mahler, M. (2011a). Sidebyside: ad-hoc multi-user interaction with handheld projectors. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 431–440.

Willis, K. D., Poupyrev, I., and Shiratori, T. (2011b). Motionbeam: a metaphor for character interaction with handheld projectors. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1031–1040.

Willis, K. D., Shiratori, T., and Mahler, M. (2013). Hideout: mobile projector interaction with tangible objects and surfaces. In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*, pages 331–338.

Willis, K. D., Xu, C., Wu, K.-J., Levin, G., and Gross, M. D. (2011c). Interactive fabrication: New interfaces for digital fabrication. In *Proceedings of the Fifth International Conference*

on Tangible, Embedded, and Embodied Interaction, TEI '11, pages 69–72, New York, NY, USA. ACM.

Winther, M. and Vallgårda, A. (2016). A basic form language for shape-changing interfaces. In *Proceedings of the TEI'16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, pages 193–201.

Wolfe, J. M. (2010). Visual search. *Current biology*, 20(8):R346–R349.

Woodworking, H. (2019). Standard furniture dimensions.

Wurman, P. R., D'Andrea, R., and Mountz, M. (2008). Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI magazine*, 29(1):9–9.

Xiao, R., Cao, T., Guo, N., Zhuo, J., Zhang, Y., and Harrison, C. (2018). Lumiwatch: On-arm projected graphics and touch input. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–11.

Xiao, R., Harrison, C., and Hudson, S. E. (2013). Worldkit: rapid and easy creation of ad-hoc interactive applications on everyday surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 879–888.

Xiao, R., Hudson, S., and Harrison, C. (2017). Supporting responsive cohabitation between virtual interfaces and physical objects on everyday surfaces. *Proceedings of the ACM on Human-Computer Interaction*, 1(EICS):1–17.

Yamaoka, J. (2014). Morphing cube.

Yao, L., Niiyama, R., Ou, J., Follmer, S., Della Silva, C., and Ishii, H. (2013). Pneui: Pneumatically actuated soft composite materials for shape changing interfaces. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 13–22, New York, NY, USA. ACM.

Yim, M., Shen, W.-M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., and Chirikjian, G. S. (2007). Modular self-reconfigurable robot systems [grand challenges of robotics]. *IEEE Robotics & Automation Magazine*, 14(1):43–52.

Yim, S. and Sitti, M. (2014). Softcubes: Stretchable and self-assembling three-dimensional soft modular matter. *The International Journal of Robotics Research*, 33(8):1083–1097.

Yim, S., Sung, C., Miyashita, S., Rus, D., and Kim, S. (2018). Animatronic soft robots by additive folding. *The International Journal of Robotics Research*, 37(6):611–628.

Yu, M., Ye, Z., Liu, Y.-J., He, Y., and Wang, C. C. (2019). Lineup: Computing chain-based physical transformation. *ACM Transactions on Graphics (TOG)*, 38(1):1–16.

Yui, T. and Hashida, T. (2016). Floatio: Floating tangible user interface based on animacy perception. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 43–45.

Zhang, J. (1997). The nature of external representations in problem solving. *Cognitive science*, 21(2):179–217.

Zhang, J. and Norman, D. A. (1994). Representations in distributed cognitive tasks. *Cognitive science*, 18(1):87–122.

Zhang, J. and Patel, V. L. (2006). Distributed cognition, representation, and affordance. *Pragmatics & Cognition*, 14(2):333–341.

Zhang, K. and Follmer, S. (2018). Electrostatic adhesive brakes for high spatial resolution refreshable 2.5 d tactile shape displays. In *2018 IEEE Haptics Symposium (HAPTICS)*, pages 319–326. IEEE.

Zhao, Y., Kim, L. H., Wang, Y., Le Goc, M., and Follmer, S. (2017). Robotic assembly of haptic proxy objects for tangible interaction and virtual reality. In *Proceedings of the 2017*

ACM International Conference on Interactive Surfaces and Spaces, pages 82–91. ACM.

Zigelbaum, J., Kumpf, A., Vazquez, A., and Ishii, H. (2008). Slurp: tangibility spatiality and an eyedropper. In *CHI'08 Extended Abstracts on Human Factors in Computing Systems*, pages 2565–2574.