# PHRASING BIMANUAL INTERACTION FOR VISUAL DESIGN

A Dissertation

by

ANDREW MARTIN WEBB

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

| | |
|---|---|
| Chair of Committee, | Andruid Kerne |
| Committee Members, | Jinxiang Chai |
| | Tracy A. Hammond |
| | Weiling He |
| | Steven M. Smith |
| | Francis Quek |
| Head of Department, | Dilma Da Silva |

May 2017

Major Subject: Computer Science

ABSTRACT

Architects and other visual thinkers create external representations of their ideas to support early-stage design. They compose visual imagery with sketching to form abstract diagrams as representations. When working with digital media, they apply various visual operations to transform representations, often engaging in complex sequences. This research investigates how to build interactive capabilities to support designers in putting together, that is phrasing, sequences of operations using both hands. In particular, we examine how phrasing interactions with pen and multi-touch input can support modal switching among different visual operations that in many commercial design tools require using menus and tool palettes—techniques originally designed for the mouse, not pen and touch.

We develop an interactive bimanual pen+touch diagramming environment and study its use in landscape architecture design studio education. We observe interesting forms of interaction that emerge, and how our bimanual interaction techniques support visual design processes. Based on the needs of architects, we develop LayerFish, a new bimanual technique for layering overlapping content. We conduct a controlled experiment to evaluate its efficacy. We explore the use of wearables to identify which user, and distinguish what hand, is touching to support phrasing together direct-touch interactions on large displays. From design and development of the environment and both field and controlled studies, we derive a set methods, based upon human bimanual specialization theory, for phrasing modal operations through bimanual interactions without menus or tool palettes.

DEDICATION

To my parents, for all their love, support, and encouragement.

To all the comic book artists whose comics I read as a kid, inspiring my love for the visual.

ACKNOWLEDGMENTS

I would like to express my deep gratitude to my adviser and friend, Prof. Andruid Kerne, for his guidance and mentoring of my growth as a researcher. I feel so fortunate to have had you as my adviser. Our collaboration has been a long road so far, and I will miss working with you this closely. I would like to thank to my committee for providing feedback and critique of my work. A special thanks to Profs. Jun-Hyun Kim and Galen Newman from the Department of Landscape Architecture and Urban Planning for their excitement about the work and willingness to involve their courses in supporting this research. To my friends, former lab mates, and fellow D&D companions—Bill Hamilton, Nic Lupfer, Rhema Linder, and Sashikanth Damaraju—you kept me sane, providing a "small council" for discussing not only topics related to research, but matters of life. My life is forever changed and made more complete thanks to each of you. To all the undergraduate students who contributed to this research—Hannah Fowler, Alyssa Valdez, Zach Brown, Elizabeth Kellogg, Cameron Hill, John Goen, and Matthew Kiihne—I could not have done it without you.

## CONTRIBUTORS AND FUNDING SOURCES

### Contributors

This work was supervised by a dissertation committee consisting of Professor Andruid Kerne and Professors Tracy Hammond and Jinxiang Chai of the Department of Computer Science and Engineering, Professor Steven M. Smith of the Department of Psychology, Professor Weiling He of the Department of Architecture, and Professor Francis Quek of the Department of Visualization and Department of Computer Science and Engineering.

The interaction techniques presented in Chapter III were developed in part by undergraduate researchers Cameron Hill, Elizabeth Kellogg, and John Goen. The study presented in Chapter V was developed and conducted in part by undergraduate researcher Zach Brown. The tutorial videos used in the study was created by undergraduate researcher Matthew Kiihne. The data analysis presented in Chapter IV was conducted in part by undergraduate researcher Hannah Fowler. The work presented in Chapter VI was conducted by the student during an internship at Microsoft Research in collaboration with Michel Pahud, Ken Hinckley, and Bill Buxton. Papers on work presented in Chapter V and Chapter VI were published in 2016.

### Funding Sources

This work was made possible in part by the National Science Foundation under grants IIS-074742, IIS-1247126, and IIS-1528044. Any opinions, findings, and conclusions or recommendations expressed in this work are those of the author and do not necessarily reflect the views of the NSF.

Graduate study was supported by a dissertation fellowship from Texas A&M University during the 2015-2016 academic year.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

# CHAPTER I

# INTRODUCTION

Human-computer interaction (HCI) researchers face new challenges as we move away from mouse and keyboard toward sensing direct hand movement on and around interactive surfaces—e.g., through direct pen and multi-touch input [e.g., Hinckley et al. 2010], computer vision with cameras [e.g., Hilliges et al. 2009], and inertial sensing with wearables [e.g., Webb et al. 2016c]. Body movements, particularly gestures that attempt to 'give' information, play a central role in communicating meaning and expressing ideas in human-to-human dialogues [Kendon 2004]. New sensors enable creating *embodied dialogues* between humans and interactive systems, in which movements of the body, in particular the hands, convey intention (e.g., commands) and parameters of interactions. Interactions designed for mouse and keyboard typically use windows, icons, menus, pointer (WIMP) interfaces—performing interactions through a *single point*, the cursor, whose position is manipulated with one hand. In WIMP interfaces, users transition among different modal operations through interface widgets (e.g., tool palettes and menu bars) and keyboard shortcuts. These WIMP interfaces fail to take full advantage of the human body's ability to express more complex dialogues via gestural interaction, such as through multiple contact points with two hands. To meet the challenges of post-WIMP [van Dam 1997] interaction design, researchers need to take into account the human body's expressive capabilities for movement and communication, drawing on both hands and new input sensing technologies.

This research investigates embodied bimanual interaction in the context of visual design, developing post-WIMP interfaces to support creative visual thinkers, such as architects. Creative visual thinkers produce external representations for *ideation*—the process of generating new ideas. For example, architects create abstract diagrams to support early-

stage ideation by composing and manipulating visual imagery with sketching. They use a multitude of actions to create, transform, and organize visual representations of their ideas. Early-stage ideation processes are free-form and divergent. They branch along many paths, rather than working toward a specific solution. Ideation involves exploration through visual transformation of representations.

Digital design tools, such as Adobe Illustrator[1], allow creating and manipulating idea representations. However, these tools are focused on helping users create finished representations, rather than explore different possibilities. They use WIMP interfaces. Parameters of operations, such as the thickness of a stroke or the opacity of a layer, are often specified via text boxes and sliders that enable defining specific values. In early-stage design, desired values are not always known before hand, but emerge through exploration. While these tools provide a large number of operations for performing visual transformations, in the service of discoverability, they have relied on repeated access to tool palettes and hierarchical menus to switch among operations. We suspect this interferes with exploratory and creative cognition processes, as the user must continuously switch between performing transformations and selecting operations through interface elements.

To support early-stage design processes that are exploratory, we need to design post-WIMP interfaces that use new direct hand movement sensing. Devices that support the pen+touch modality, with simultaneous pen and multi-touch sensing, are growing more common and cost effective (e.g., Microsoft Surfaces[2] and Apple iPad Pro[3]). The increasing availability of these devices and the familiarity of the pen make this modality suitable for visual design interfaces. One key challenge for post-WIMP pen+touch interaction design is how to use gestures to improve support for transitions between operations.

[1] http://www.adobe.com/products/illustrator.html
[2] http://microsoft.com/surface
[3] http://www.apple.com/ipad-pro


Figure 1: Landscape architecture student using bimanual interaction to apply perspective transform to an image. Left-hand touch defines the pivot point on which to rotate the image, while the right-hand using pen position and pressure defines the amount of rotation.

The principle research question that we seek to address is:

> How can *bimanual interaction*—which takes advantage of the capabilities of the pen+touch modality and the human body's capabilities for movement and communication—effectively support designers, in early-stage ideation, as they transition between modal visual operations (e.g., scaling, layering, sketching, undo, opacity)?

We explored this research question through the iterative development and evaluation of a pen+touch diagramming environment, called Embodied IdeaMâché, or *Emmâ* for short (see Chapter III). Emmâ allows designers to compose imagery, text, and sketching within an infinite, zoomable canvas (see Figure 1). Through bimanual interaction, designers switch among various modal operations using a combination of gestures, affordances, and visual feedback. One example, *LayerFish*, is an interaction technique for layering overlapping content to support visual designers working with many layers of elements.

Methodologies in HCI research have grown beyond their initial roots in cognitive psychology, deriving a variety of methods for understanding how people engage with technology [Olson and Kellogg 2014]. Beyond evaluating user interfaces in the laboratory, other methods include performing ethnographic investigations of situated contexts [Suchman 1987], deploying interactive systems in the field to study user interactions in situ [Siek et al. 2014], and engaging in research through design, where "design artifacts, as outcomes, can transform the world from its current state to a preferred state" [Zimmerman et al. 2007]. We employ two approaches: (1) a hybrid study, combining a field study deployment of Emmâ in design studio education with a laboratory environment; and (2) a controlled laboratory experiment investigating LayerFish. Landscape Architecture students in design studios used Emmâ to create diagrams during early-stage design ideation on projects (see Chapter IV). Observations of students' use fed into iterative design cycles. In the controlled experiment, we evaluated layering task performance of LayerFish, isolated from the Emmâ environment. By isolating LayerFish, we can directly investigate specific aspects of the interaction design, such as the efficacy of fisheye visualizations [Furnas 1986] for layering tasks (see Chapter V).

Beyond work with Emmâ, we augmented multi-touch input on large displays with wearable sensing to support asymmetric assignment of roles to the hands, enabling designing bimanual interaction in accord with principles of bimanual specialization theory (see Chapter VI).

An emergent theme of this research is to identify key ways in which interaction designers can put together—i.e., *phrase*—bimanual interactions to support embodied dialogues with computing systems. We frame this theme and the associated research by articulating and connecting sensitizing concepts from choreography, bimanual interaction, embodied cognition, visual thinking, diagramming, and free-form web curation. From our investigation, we derive a set of methods that arose as common practices, when designing embod-

ied interactions, to support various modal operations used for architecture diagramming within a post-WIMP interface. These methods have implications for design of bimanual interaction (see Chapter VII).

## I.1 Sensitizing Concepts

We ground this research using sensitizing concepts across multiple disciplines, which guide and provide reference for ways of looking at and understanding the work [Blumer 1954]. We apply aspects of phrasing from choreography to support interaction design. We further develop a bimanual interaction design methodology that uses phrasing and principles from human bimanual specialization theory. Embodied cognition motivates our use of embodied dialogues to support designer ideation. We connect concepts of visual thinking, diagramming, sketching, and free-form web curation in our design of Emmâ.

### I.1.1 Phrasing

A phrase is a unit of structure used in linguistics [Levelt 1993], music [Nattiez 1990], and choreography [Blom and Chaplin 1982; Humphrey 1959; Maletic 1987]. The Oxford English Dictionary defines a *phrase* as "a small group or collocation of words expressing a single notion, or entering with some degree of unity into the structure of a sentence; a common or idiomatic expression." A phrase is a building block for constructing more complex sequences of words, musical notes, or movements. It expresses an idea or communicates meaning. We apply aspects of phrasing from choreography to interaction design, building embodied dialogues with computing systems.

#### I.1.1.1 Choreography

In choreography, a phrase is, "a grouping of related movements that have kinesthetic logic and intuition, and are connected by their cooperative creation of a unit" [Blom and Chaplin 1982]. A phrase has form and content. It has a beginning, middle, and end; it

rises and falls. Operational or expressive gestures typically have a rhythmic pattern that begins with preparation, followed by one or more efforts defining underlying movements, and then termination [Laban and Ullmann 1963]. A phrase conveys a notion. In contrast, non-phrased movement, much like a random sequence of words in language, provides little interest and conveys no meaning. Interaction with computing systems must convey meaning in order to perform operations. We suspect that choreographic phrasing will support interaction design, particularly gestural interactions where human movement directly enacts commands.

A natural part of a phrase is a high point [Blom and Chaplin 1982]. A high point is denoted by either extreme change or gradual buildup in movement. Examples of high points include the peak of a jump or the moment when a hammer hits a nail. It is considered the most important part of a phrase, providing noticeable form. The high point and associated movements give a phrase a recognizable shape.

A phrase's movement has rhythm, alternating between tension and relaxation [Maletic 1987]. Through kinesthetic tension, movement becomes consciously guided and controlled. Tension arises in gestural interactions with computing systems, where input most follow a specific path in order to be recognized by the system and perform an operation. Relaxation is not the absence of movement, but rather, a variation of the strength and shape of human effort, contrasted with tension [Laban and Ullmann 1963]. For example, using a thumb touch to scroll a web page on a mobile phone involves initial tension in the touch down and drag (with friction in the movement across the surface), followed by a relaxation as the touch is lifted up and moved back to its initial position. Scrolling further repeats this rhythmic pattern.

As structural units, phrases are combined together into sequences, with their own form and meaning. A sequence also has a beginning, middle, and end, as well as a high point.

Laban Movement Analysis (LMA) is formal method for describing human movement

[Laban and Lawrence 1947; Laban and Ullmann 1963]. LMA involves recognizing and describing patterns of change in terms of what is moving (Body), how the body is moving (Effort), where the body is moving (Space), the relationship of changes in body form to the environment and mover (Shape), and where the emphasis is put in the movement (Phrasing) [Fdili Alaoui et al. 2015]. Phrasing can be *Impulsive*, where changes in tension are highest at the beginning, *Swing*, where changes tension are highest in the middle, and *Impactive*, where changes in tension are highest at the end.

For bimanual interaction design, Body relates to the hands; Effort relates to qualities of the movement, such as speed and pressure; Space relates to where on the display the interaction occurs; and, Shape relates to what parts of the hand are touching. In gestural interaction, a touch flick to inertial scroll is an example of Impulsive Phrasing, emphasizing the initial acceleration. Dragging an element is an example of Swing Phrasing, emphasizing the movements for positioning an element. Pigtail [Hinckley et al. 2005] and other demarcation gestures are examples of Impactive Phrasing, emphasizing command selection.

#### I.1.1.2 Human-Computer Interaction

We apply components of phrasing, such as high points and tension, to embodied interaction design. Buxton introduced the notion of phrasing in HCI [1986]. He describes how operations articulated through gestures can use *tension* to phrase together "statements" of interaction with computing systems. The tension and its relaxation connect multiple operations required to perform a (compound) task. Through the use of appropriate tension, the interaction designer is able to reinforce specific operational chunking in users' mental models.

In WIMP interfaces, the tension is often minimal, involving clicking a button or menu option, and is typically more visual than kinesthetic. Yet, kinesthetic tension is more ef-

fective for reducing mode switching errors [Sellen et al. 1992]. Additionally, the duration of tension affects reinforcement of operational chunking [Hinckley et al. 2006]. Tension maintained throughout an operational sequence improves chances that users will connect the interactive sequence as a unit. While kinesthetic tension has been used in post-WIMP interfaces to phrase interactions [Frisch et al. 2011; Hinckley et al. 2006, 2010; Vogel and Casiez 2011], little has been articulated on phrasing as a paradigm for embodied interaction design. In particular, no prior work to the best of our knowledge has examined different methods to use and combine techniques that phrase bimanual interaction through tension, relaxation, high points, and effort.

### I.1.2 Bimanual Interaction

Guiard investigates human bimanual specialization [1987]. His emphasis on the logic of the division of labor between the hands has widely influenced the design of two-handed interfaces. Guiard introduces concepts to explain how roles become assigned to the hands in human activity: lateral preference, Left-Hand Precedence, and Right-to-Left Spatial Reference.

#### I.1.2.1 Lateral Preference

As opposed to the study of 'manual superiority,' which (perhaps misguidedly) frames one hand as 'superior' to the other, Guiard explains how the hands work together in terms of *lateral preference*, which frames two-handed action in terms of the hands' complementary roles:

> In a task consisting of two differentiated manual roles, A and B, there are two possible ways to assign role A and role B to the left and right hands (A–left and B–right, or A–right and B–left).

In typical multi-touch input, even if the designer desires specific roles (A vs. B) for the

hands (right vs. left, preferred vs. non-preferred), this desire becomes undermined by lack of input sensing that recognizes the correspondence of touch points to hands. Furthermore, when the user is unknown, the mapping of right=preferred and left=non-preferred (or vice-versa) injects further ambiguity during multi-user input. Thus, typical multi-touch cannot sense or abide by any particular choice for A–left and B–right versus A–right and B–left. The designed role of the hands (lateral preference) is fundamentally ambiguous.

#### I.1.2.2 Left-Hand Precedence

Guiard observes that (for right-handers) the left hand tends to lead the right in its contribution to a joint activity. For example, the left hand positions and orients a sheet of paper, then the right hand begins to write. In general, Guiard codifies this as the *Left-Hand Precedence* principle.

But users may touch with the preferred hand in isolation. If the touch cannot be distinguished as left vs. right, it remains ambiguous from the system's viewpoint. Is this indeed an isolated touch of the preferred hand? Or is it perhaps a non-preferred hand touch, which via Left-Hand Precedence should be interpreted as preparatory to the (complementary) action of the preferred hand? The system has no way of discerning the user's intent at the onset of the first touch.

#### I.1.2.3 Right-to-Left Spatial Reference

The principle of Right-to-Left Spatial Reference states that "motion of the right hand typically finds its spatial references in the results of the motion of the left." This means that the left hand generally should establish (and frame) the interaction, relative to which the right hand takes action.

#### I.1.2.4 Symmetric vs. Asymmetric Bimanual Interaction

While Guiard focuses on asymmetry, he does note that certain acts, such as lifting weights or turning a steering wheel, often invoke symmetric action of the hands. And certain idiomatic multi-touch gestures, such as bimanual shrink and stretch, show that symmetric mappings have utility as well. However, by their very nature symmetric mappings employ an interchangeable assignment of roles to the hands. This can mask a touchscreen's underlying inability to differentiate left from right. DiamondTouch [Dietz and Leigh 2001], for example, can sense when different users touch, but cannot distinguish which hand (left or right). This doesn't matter for symmetric mappings, but ambiguities still lurk: the designer cannot realize differentiated touches that fully abide by both Left-Hand Precedence and Right-to-Left Spatial Reference—nor support a flexible interleaving of symmetric and asymmetric interactions.

#### I.1.2.5 Guiard-abiding Interactions: Phrasing Activity of the Right Hand together with the Left

Our analysis highlights a novel design methodology, which we call *Guiard-abiding* [Webb et al. 2016c], that uses lateral preference and kinesthetic tension to phrase together bimanual interactions in a manner consistent with Guiard. Phrasing uses kinesthetic tension (which may be nothing more than a light touch to maintain contact with a touchscreen) as a way to tie together a series of inputs into a single, cohesive gestural phrase. Then, during bimanual interaction the left hand establishes and "holds down" the phrase. The right hand then inserts its actions—which may consist of multiple taps, drags, or other gestures—into the reference frame specified by the left hand, and which thereby collectively gives those actions (phrases) a specific interpretation, or meaning.

### I.1.3 Embodied Cognition

Embodied dialogues involve humans' perceived understandings of interactions with computing systems [Dourish 2001]. Humans sense, perceive, and interpret the world around us through our bodies [Merleau-Ponty 1965]. For example, people understand that we can sit on objects, with flat tops of certain heights, which we would not categorize as chairs (such as a low wall). Gibson calls this understanding of the potential for action based on the recognition of the physical relationship between our bodies and particular objects, *affordance* [Gibson 1979].

In turn, our cognitive processes and mental models are based in and connected to our bodies [Glenberg 1999]. For example, when readers manipulate objects to correspond to characters and actions in a text, it greatly enhances comprehension and memory, as measured by both recall and inference tests [Glenberg et al. 2007]. Gestures help people not just to communicate meaning, but further, to remember and to understand complex ideas [Tversky et al. 2009]. These gestures augment memory and the representation of meaning.

*Thus, we hypothesize that embodied gestural dialogues with computing systems promote cognition—particularly representations of meaning—through physical movement of the body and associated kinesthetic tension in the muscles.* We seek to employ cognitive and perceptual embodied resources associated with multi-modal interfaces in our bimanual pen+touch interaction design [Quek 2006].

### I.1.4 Visual Thinking

Visual thinking, from the initial perception of visual stimuli to the cognitive processes associated with concept formation and abstraction, plays a fundamental role in fostering creative ideas. Interpretation is rooted in perception [Hegel 1807; Hegel and Baillie 2003], and perception is inseparable from interpretation [Merleau-Ponty 1965]. Arnheim

explains the foundational role of visual perception in the formation of concepts, where the mind operates with more than just visual stimuli from the eye, bringing together visual imagery from memory and organizing a lifetime of experiences into visual concepts [1969]. Visual perception thus operates on higher order cognitive processes than just recognition of patterns. Visual perception can compare and discriminate, supporting recognition of conceptual combinations. In creative cognitive processes, emergent ideas can arise from conceptual combination and synthesis, that is mental blending of two unrelated concepts to form a new unique concept [Finke et al. 1992]. Dondis describes a similar phenomena with the polarities of visual composition, contrast and harmony, which sharpen or level visual clues affecting perceptual comparison and aiding visual communication [1974].

### I.1.5 Diagramming and Sketching

A *diagram* is a design thinking tool that enables and stimulates imagination, facilitating conceptualization and ideation. Diagrams mediate exploration of relationships between concepts, using ambiguous visual representations to foster varied, flexible interpretations. Ambiguity can lead to a sense of indeterminacy, lacking certainty or clarity, as relationships and concepts evolve through iterative cycles of refinement. This indeterminacy promotes reading "purposelessly, that is, jumping here and there and responding at the same time to environmental events," [Cage 1961] such as visual stimuli that provoke new ideas. Abstract diagrams used in conceptual design are non-linear and rhizomatic [Deleuze 1988], lacking a beginning and end as well as a clear sequence for reading. A diagram acts as an "engine of novelty" [Kwinter 1998], an abstract machine, "defined by its informal functions and matter and in terms of form [which] makes no distinction between content and expression, a discursive formation and a non-discursive formation" [Deleuze 1988]. Diagrams document aspects of integral design thinking processes [Peponis et al. 2002].

A common method for creating diagrams is sketching. Sketches act as interactive imagery [Goldschmidt 1991], in which strokes are drawn, re-drawn, drawn over, and erased, transforming ideas. Sketching stimulates and affords reflection [Arnheim 1969; Goldschmidt 1991; Schön and Wiggins 1992]. Ideas are externalized, manipulated, reflected upon, and reinterpreted [Suwa et al. 2001]. The physical medium of pencil and paper enables designers to quickly sketch out ideas while simultaneously manipulating diagrams in various ways, supporting creative discovery. Externalized ideas are combined and restructured to form new ideas [Verstijnen et al. 1998]. Sketches can be rigorous in visually describing details, but also ambiguous, using abstract forms and implicit visual features. When sketches have explicit structures, such as in mechanical engineering diagrams, programmatic languages can be defined to enable machine recognition of the sketches [Hammond and Davis 2005].

Suwa and Tversky [1997] found that sketches enable architects to see non-visual functional issues from visual features. Similar to Arnheim's idea of intelligent perception [1969], Suwa and Tversky exposit the importance of how visual attributes are perceived and implications for computational design tools. They suggest presenting visual stimuli as fluctuations in the visual features and spatial arrangements of elements when a designer sketches. Suwa et al. [2001] investigated how architects are able to reinterpret sketches over time to have new ideas. They discovered that architects regroup parts of sketches to form new wholes.

### I.1.6 Free-form Web Curation

Each day, on social media, hundreds of millions of people engage in *web curation*—they choose, comment on, and organize content. A limitation is that popular apps—e.g., Facebook, Twitter, and Pinterest—constrain organization of curated content to feeds and boards, using ranking algorithms based on parameters such as recency and popularity.

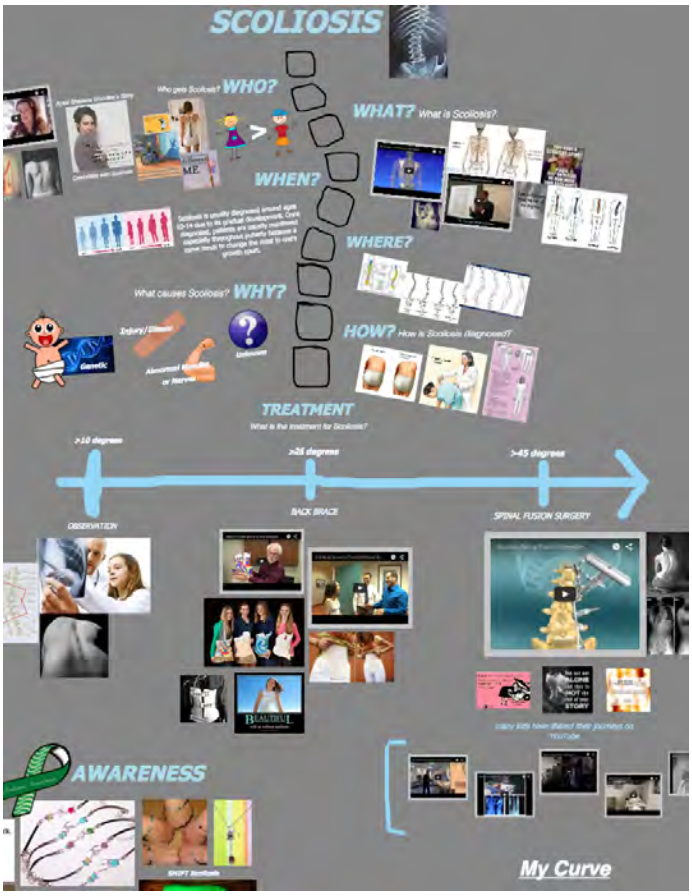


Figure 2: In *Scoliosis*, a personal health informatics free-form web curation, Katharine Kimmel manifests visual thinking by assembling her own sketches and writing with images from the web and YouTube videos (`https://ideamache.ecologylab.net/v/bM6IaLoVvT/`). As a whole, the curation is educational and hopeful, conveying a sense of complexity and connectedness in dealing with a great challenge. Reprinted from [Lupfer et al. 2016].

Linear feeds are great for showing people the latest content, but provide limited support for free-form visual thinking that articulates relationships among content elements. Visual patterns, blends and combinations that express relationships are significant generative mechanisms of creative cognition [Finke et al. 1992].

We define curation in a broader cultural context by drawing from the arts. *Curation* is the creative conceptualization and organization of an exhibition [O'Neill 2012]. Works

are gathered, interpreted, and arranged, creating context, giving form to visual thinking, and producing cultural meaning.

*Free-form web curation* overlaps with diagramming as a form for articulating relationships and composing them as whole. In particular, free-form web curation enables multimedia elements to be spontaneously gathered from the web, written about, sketched amidst, manipulated, and visually assembled in a continuous space (see example, Figure 2) [Lupfer et al. 2016; Webb et al. 2016b]. Cultural theorists, Deleuze and Guattari, describe the assemblage of cultures and societies as a *rhizome*, a meshwork of interconnected heterogeneous elements without a clear beginning or end [1987]. The relationships among elements are as important as the elements themselves. A rhizome is a map conveying relationships rather than a tracing, which accurately reproduces elements at the expense of the whole. The medium of free-form web curation constitutes a map, not a tracing, and so functions rhizomatically, liberated from the confines of particular websites by a zoomable user interface. The goal of this holistic medium is to support creative cognition [Finke et al. 1992] of relationships and the emergence of new ideas.

### I.2 Diagramming Environment: Emmâ

Our design goal for Emmâ was to support architects in early-stage design ideation by investigating the expressive potential of gestural interactions, using the pen+touch modality, for diagramming and free-form web curation. Prior tools, such as Adobe creative products and computer-aided design (CAD) programs, employ WIMP interfaces. They emphasize support for creating polished, later-stage design representations. These tools favor precision over exploration in performing visual transformations. New versions of these tools continuously add new transformation operations to an already large library of possible operations, rather than improving expressive capabilities in existing operations. We observed that the abstract, informal nature of early-stage design processes differ from


Figure 3: Abstract diagram for fashion design ideas involving metal rings and silk fabric, which elicit a contrast of warm and cold.

the explicit, formal structures often used in these design tools.

Emmâ enables designers to gather and assemble image, text, and video content from the Web and personal media sources (e.g., camera phones) in an infinite, zoomable 2D design space (Figure 3). Emmâ provides a blank canvas on which to organize a space of ideas. The zoomable space allows architects to work across varying scales. Through different input modalities, designers can create and transform content. Using the pen, users can sketch over and amidst content in the design space. Gathered content can be freely translated, scaled, and rotated using the familiar multi-touch two-finger stretchy gesture.

Within Emmâ, we developed a suite of bimanual interaction techniques to support processes used by designers, including stylized sketching, opacity, blending, layering,

duplicating, and perspective transformation (Figure 1). Our techniques use phrasing in combination with gestures, affordances, and visual feedback to allow effective transitioning among the different tasks associated with diagramming and free-form web curation. Our techniques are Guiard-abiding (Section I.1.2.5), in that the left hand initiates interaction and defines a frame of reference for the actions of the right hand. The assignment of roles to the hands is not explicitly enforced. Instead, affordances and input modalities are used to encourage specific roles for the hands. For example, Emmâ contains an edge-constrained gesture region [Hamilton et al. 2012] where touches from the left, non-preferred hand activate modal switches for right hand actions. The edge-constrained region is positioned in the bottom corner of a display on the user's left-hand side to afford use with that hand.

## I.3 Landscape Architecture Design Studios

We conducted a qualitative investigation of Emmâ in a landscape architecture design studio. We combined aspects of controlled laboratory experiments and field studies into a hybrid evaluation. This hybrid evaluation sought to understand how landscape architects would use Emmâ on "real world" design tasks in a studio-like setting. A studio-like lab space was created featuring four high-powered workstations with Wacom pen surfaces that supported touch input. Each workstation also consisted of two cameras to capture observations of participants hands on and around interactive surfaces. Graduate landscape architecture students in the Professional Study design studio were invited to our lab space to work on their projects. Students created diagrams using Emmâ to support their conceptual design for projects, which were initiated by industrial sponsors.

For data collection, we integrated camera and screen capture data sources into what we call, *embodied process recordings*, which collect both the design process and the embodied interactions of participants. These integrated recordings enable researchers to observe

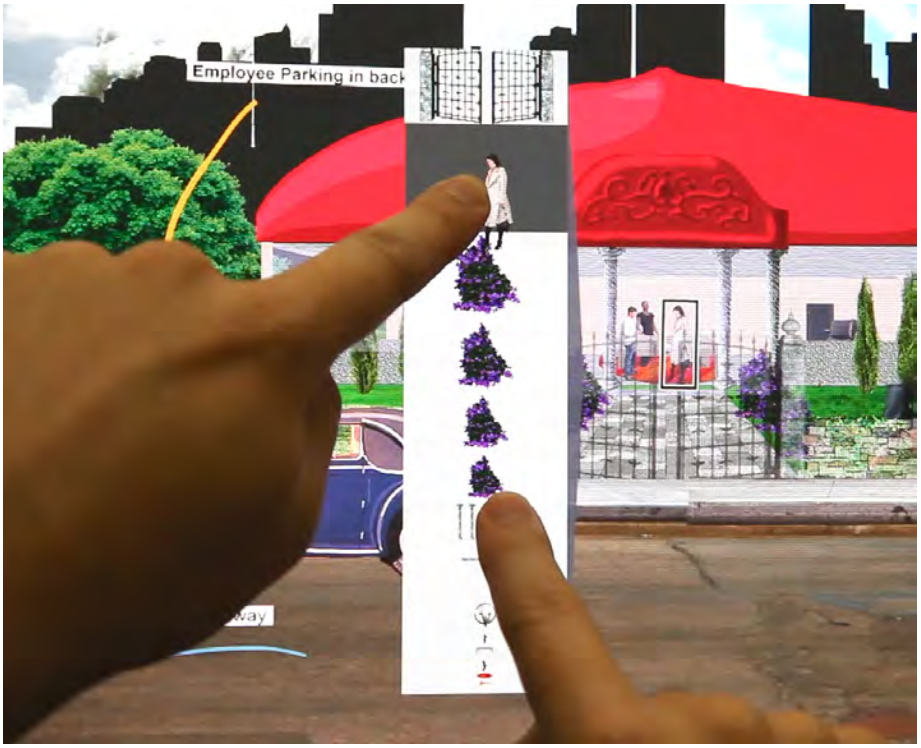


Figure 4: Example of user performing bimanual adjustment of overlapping content with LayerFish. Reprinted from [Webb et al. 2016a].

and analysis simultaneously actions performed in Emmâ and movement of the hands in relation to those actions. For example, using these recordings, one can understand how participants used different hands and fingers to perform certain actions. Diagrams created by students were also collected. Student feedback was gathered through interviews, observations, and informal discussions. Responses fed back into iterative design of Emmâ. We performed visual grounded theory analysis [Konecki 2011] on the embodied process recordings, transcribing, coding, and categorizing interesting phenomena related to bimanual activity. From our analysis, we derived implications for design of bimanual pen+touch interactions.

## I.4 LayerFish

While the design studio investigation provided a holistic understanding of how students engaged in diagramming and free-form web curation with Emmâ, it did not evaluate the effectiveness of specific interactions. A controlled study was conducted with LayerFish, a bimanual interaction technique for working with overlapping content in Emmâ (see Figure 4). LayerFish is unique among the set of bimanual interaction techniques designed within Emmâ in that it supports a number of different design tasks, including layering, accessing, and manipulating overlapping content. This unique quality made it suitable for a more focused investigation.

From interviews in the design studio investigation, we found that landscape architects often work with 10s to 100s of overlapping elements. When elements overlap, designers need tools for reordering of the visual stack, or *layering*, to place some elements in front of or behind others. A common approach in design tools is to provide a scene index, as an ordered list of layered elements. Scene indexes become difficult, in terms of physical effort and human cognition, to deal with when working with hundreds of elements. Further, the index is often located out of the user's visual focus, requiring her to split attention between the design space and the index.

LayerFish addresses issues of scene index scale and split attention through a fisheye visualization and in-place positioning of the scene index at the user's point of focus. The fisheye distorts the visual space, decreasing the sizes of correspondents away from a focus element, and so enables more to be visible than a typical scene index. However, interaction issues arise, as the spatial distortion disrupts layering and scrolling operations. Bimanual techniques within LayerFish address these issues, allowing the focus element to remain fixed while the scene index reorders layers and scrolls.

We evaluated layering task performance with LayerFish in comparison to a traditional

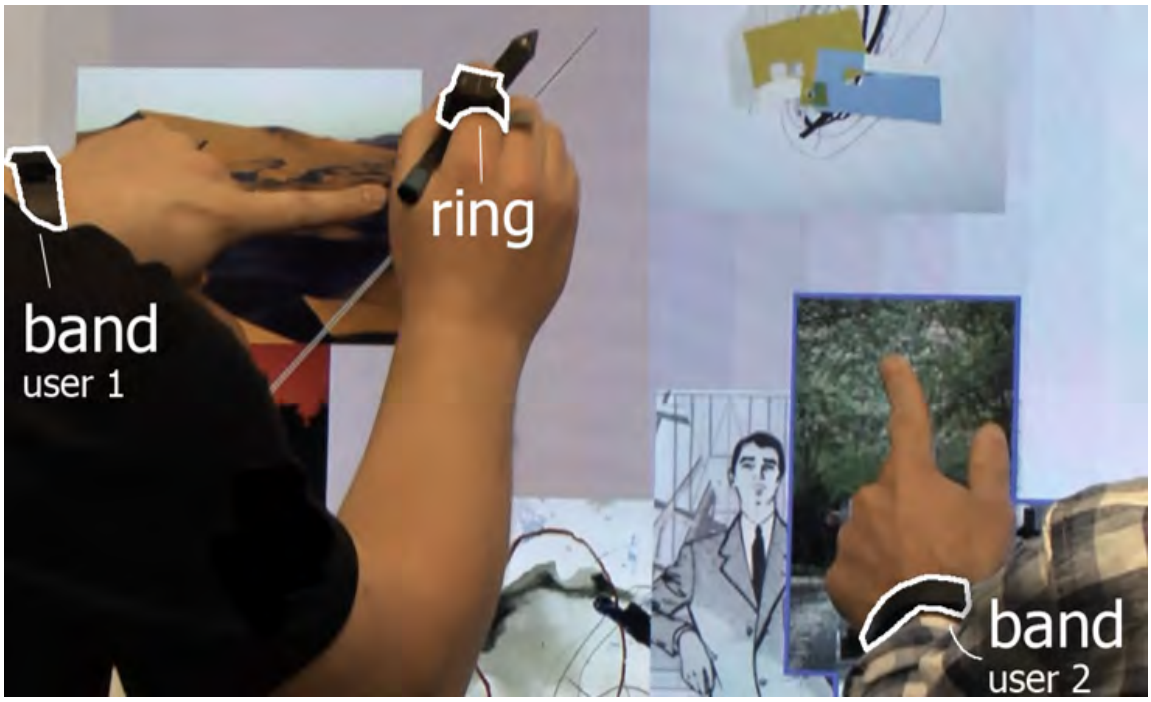


Figure 5: Wearables provide missing context (who touches, and with what hand) for direct-touch bimanual interactions. Reprinted from [Webb et al. 2016c].

scene index. Findings indicate that the fisheye reduces time to find an element and selection reduces layering time when elements do not heavily overlap.

### I.5 Wearables as Context for Guiard-abiding Bimanual Touch

As an additional line of inquiry into phrasing bimanual interactions, separate from Emmâ, we augment touch input with wearables to identify which user, and distinguish what hand, is touching a large display (see Figure 5)—and to thereby phrase together direct-touch interactions in a manner consistent with the bimanual-asymmetric assignment of roles to the hands—right vs. left, preferred vs. non-preferred—as set forth by Guiard [1987]. Our key point is that this added context—who touches and with which hand—enables design of interactive dialogues that can flexibly assign appropriate roles to each hand, including symmetric role assignments where appropriate. Because of the arbitrary

limitations and ambiguities inherent in standard multi-touch technology, most previous systems exploring bimanual touch—including efforts motivated by Guiard's insight—have not been able to fully support this design goal.

### I.6 Phrasing Bimanual Interaction

This research develops a methodology for bimanual interaction design to support transitioning among a set of operations within post-WIMP interfaces using phrasing with Guiard-abiding interactions. We derived a set of common practices that emerged in our interaction design. We found three techniques for performing interactive operations to be useful as methods for bimanual phrasing: quasimodes, transient interfaces, and contextual operations. Through making use of Guiard's principles of Left-Hand Precedence and Right-to-Left Spatial Reference, we define asymmetric roles to the hands for phrasing these interaction methods. Phrases begin with movements of the left hand, that are sustained throughout entire phrases. Phrases end when the left hand is lifted. We draw on Laban's choreographic concepts of Shape and Space to differentiate the function of interaction techniques for bimanual phrasing.

#### I.6.1 Quasimodes (Space)

Quasimodes, or spring-loaded modes, are transient modal states that allow a user to switch from a primary mode, such as drawing ink with the pen, to another mode, such as lasso selection, for a brief interaction before returning back to the primary mode [Hinckley et al. 2006]. We phrase quasimodes with gestures of the left hand. Performing a gesture activates the quasimode. Holding the gesture while performing the operation provides kinesthetic tension and sustained effort to connect the sequence interactions as a quasimode phrase. The high point or emphasis of the phrase switches to the right hand as actions are performed. Modifying the left-hand gesture switches quasimodes. A user engages in a rhythmic pattern alternating emphasis of effort between the two hands. Re-

leasing the left-hand gesture, ends the phrase, returning back to the default mode.

This phrasing already exists in mouse and keyboard interfaces, where holding a modifier key changes the actions of mouse clicks. Wacom pen displays provide physical buttons on edges of the screen that replicate presses of modifier keys when held. For pen+touch or multi-touch input, interaction designers can provide regions on or near the edges of the display where users can perform and hold gestures for phrasing quasimodes [Hamilton et al. 2012; Matulic and Norrie 2013]. Using regions on or near edges of the display provide fixed locations, away from the user's central focus, but with physical affordances (e.g., device edge or bezel) that support finding the space without too much visual attention.

In architecture diagramming, the pen is used to draw. Therefore, in Emmâ, the default mode for the pen is inking, as advocated by Hinckley et al. [2010]. Designers engage quasimodes using an edge-constrained gesture space, called the *Hotpad*, located in the bottom corner on the users' left-hand side. The Hotpad supports chorded gestures where the number of touches activates different operations as quasimodes. The user transitions between quasimodes by adding or releasing touches. One touch activates selection mode, where each new pen gesture defines a lasso or crossing [Apitz and Guimbretière 2004] selection. Two touches activates translucence mode, where pen or touch input on an element adjusts the translucence of the element with up or down movements. Three touches activates edge blending mode. While in this mode, pen or touch input near the edge of element adjusts the width of an opacity mask gradient. The gradient softens the edge of an element, so that it blends more easily with overlapping content.

### I.6.2 Transient Interfaces (Shape)

When activated by the user, transient interfaces appear in-place at the user's point of focus allowing for command selection and parameter manipulation [Webb and Kerne 2008]. Examples of transient interfaces include pop-up menus, marking menus in novice mode,

and dialog boxes. In WIMP paradigms, transient interfaces are typically activated via a button press on a pointer device (e.g., right-click on a mouse), creating initial kinesthetic tension to phrase the interaction. In some cases, the button must be held throughout the duration of the interaction, helping phrase a sequence of actions together.

For post-WIMP interfaces, we can use gestures to activate transient interfaces. Performing activation gestures with the left hand, allows the right hand to interact with the transient interface. Kinesthetic tension is created by holding the activation gesture. Moving the left hand adjusts the position of the transient interface, allowing the use of click-through interactions, such as with Toolglass [Bier et al. 1993]. Upon ending the phrase through releasing the left-hand gesture, the transient interface disappears.

In Emmâ, designers adjusts the color and thickness of ink strokes through a transient interface. The transient interface is phrased by a three touch right-angle gesture with the left hand, involving the index and middle fingers and the thumb. The right angle created by the gesture defines the transient interface's bottom left corner. Moving the gesture repositions the interface, so that the bottom corner remains at the gesture's right angle. Using the pen in the right hand, the user selects color and thickness for ink strokes. Without releasing the gesture, the user quickly transitions between inking and styling ink.

### I.6.3 Contextual Operations (Space+Shape)

Contextual operations are those performed within the confines of a reference frame. Typically, the reference frame is a specific element, but it could also be a spatial region in proximity with other input (see Chapter III, Straight Line Tool). Examples of contextual operations include delete, duplicate, crop, and discrete layering operations, such as "bring to front" or "send backward." Quasimodes and transient interfaces support global commands that may affect many elements (e.g., lasso selection) or ones yet to be created (e.g., ink styling), while contextual operations enact commands on a particular focus.

Figure 1 shows an example of a contextual operation being performed. The user first touches an image with her left hand, creating kinesthetic tension. From here, she drags the image around the design space. When she presses the pen (in her right hand) down on a corner of the image, a 3D perspective rotation operation is activated. Her initial touch defines a pivot point for the rotation. The operation is phrased through tension of her initial touch, followed by a pen press. She switches between translation and 3D rotation by applying or removing tension with the pen.

### I.7 Research Contributions

We contribute a set of design methods for using tension, relaxation, high points, and effort to support phrasing bimanual interactions with pen and multi-touch input. Without the need for tool palettes or menus, architects create diagrams within Emmâ using their hands. Through a field study and controlled laboratory experiment, we observed the effectiveness of these methods in supporting visual design. The design methods presented are not intended as an all encompassing set for phrasing bimanual interaction. They simply represent a set that emerged as common practice in my interaction design process. The methods, themselves, are not particularly novel. Prior work has exhibited these methods, but not all together, and not as an articulation of how they can be combined to support transitioning across complex modal states. The combination of using tension to phrase operations and assigning roles to the hands involving tension provides powerful and expressive capabilities for interaction designers to support embodied dialogues with computing systems.

# CHAPTER II

# BIMANUAL TECHNIQUES FOR COMMANDING INTERACTIVE SYSTEMS ACROSS MODALITIES

Human-computer interaction involves the relationship between human users and computing systems. One form in which this interaction takes shape is *commanding* — the directed execution of operations by a computing system through input from a human user. The prominent rise of graphical user interfaces and the personal computer in the 1980s let to an abundance of new input technologies and methods for commanding. For many years now, HCI researchers have been building and evaluating new techniques for commanding. An extensive collection of literature has been generated. Yet, the field lacks clear understanding of the scope of work that has previously been performed, and how techniques relate and compare to each other. To address this issue, we present a survey of commanding techniques, while contributing an analytic framework for comparing techniques.

We define a *commanding technique* as an interactive method in which a user, through some form of input, specifies to a computing system what operation(s) to perform. A commanding technique may support selecting from a number of commands. It may support only executing a single command. Commanding techniques may be preceded or followed by user adjustment of parameters, such as selecting the elements to operate upon or changing a numerical value (e.g. the width of an element). Some techniques combine command selection and parameter adjustment into one interaction [e.g. Guimbretiére and Winograd 2000; Pook et al. 2000].

The breadth of research on commanding techniques is large, covering many different input modalities and contexts. For our survey, we focus on commanding techniques for graphical user interfaces (GUIs). GUIs involve many popular interface paradigms, with an extensive and significant collection of prior literature. Examples of common GUI com-

manding techniques include menus, tool palettes, and keyboard shortcuts. The complete body of work on GUI commanding techniques is too large to encompass in a single article. Instead, the present research addresses techniques supported by popular input modalities of mouse and keyboard, pen, and touch. These modalities are commonly used in GUIs. The mouse has long been the standard input modality, along with the keyboard, for GUI interfaces on desktop computing systems. The pen is familiar to humans, as we use it to write and draw. Mobile devices, such as smart phones and tablets, support touch (and pen for some).

We derive an analytic framework for comparing GUI commanding techniques across several metrics: modality, coupling, handedness, learnability, expertise, scalability, fluidity, and embodiment (see Section II.2). The metrics describe qualities of the interaction techniques, in order to help us identify what has been done, what is common practice, and where are the gaps. Our framework is based upon a review of prior literature and our own personal experiences designing interaction techniques.

This chapter begins with a look at models of interaction and prior surveys of interaction techniques. Next, we present our analytic framework, with detailed descriptions of the metrics. We then apply those metrics to commanding techniques. We analyze the differences among techniques and discuss what was learned. This paper concludes with design implications for directing future research in commanding techniques.

## II.1 Interaction Models and Theories

Before delving into our analytic framework, we first review prior models of interaction, which describe, from cognitive and physical perspectives, the ways in which users command computing systems. These interaction models will serve as grounding for our analytic framework.

### II.1.1 Direct Manipulation

The direct manipulation model is the cornerstone of many GUI interfaces, particularly those used in design tools where users need to perform incremental transformations on visual objects. The model consists of four principles [Shneiderman 1983]:

- Continuous representation of the object of interest.
- Physical actions (movement and selection by mouse, joystick, touch screen, etc.) or labeled button presses, instead of complex syntax.
- Rapid, incremental, reversible operations, whose impact on the object of interest is immediately visible.
- Layered or spiral approach to learning, which permits usage with minimal knowledge.

In direct manipulation, movements of the user, sensed through an input device, invoke commands and manipulate parameters. Visual feedback is continuous as changes are made, allowing the user to explore changes without having to repeatedly invoke commands. Direct manipulation provides scaffolded learning with minimal knowledge required to engage initially. Novice users, through simple interactions with visual objects, learn a basic set of commands for using a system. As they become more experienced, they learn new commands. Since the interaction is highly visual, novice users can learn from watching experts.

A goal of direct manipulation is the straightforward translation of thoughts into physical actions that perform commands [Hutchins et al. 1985]. Direct manipulation seeks to reduce the distance "required to bridge the gulf between the user's goals and the way they must be specified to the system" [Hutchins et al. 1985]. Users experience direct manipulation as engagement with the interactive objects themselves, not the commanding interface

or computing system. In this way, direct manipulation becomes ready-to-hand [Heidegger 1962], as mechanical intermediaries (e.g., mouse or pen), computing systems, and commanding techniques become extensions of the hand and arm in the manipulation of digital objects.

Direct combination, an extension of direct manipulation, considers pairs of interaction objects, instead of single objects [Holland and Oppenheim 1999]. Direct combination requires, that between every pair of objects, there exits one or more operators. For example, dragging and dropping a file onto a directory, adds that file to the directory. Invoking commands with direct combination involves dragging (and dropping) elements on top of each other. Translucent click-through tools, such as those found in a toolglass [Bier et al. 1993], support direct combination. We observe that Agarawala and Balakrishnan [2006] use direct combination to support insertion of elements into piles, by crossing dragged elements through interface widgets.

### II.1.2 Instrumental Interaction

Instrumental interaction is a model that extends and generalizes direct manipulation [Beaudouin-Lafon 2000]. In the physical world, an instrument is a tool to manipulate objects. In this model, interaction instruments facilitate engagement between users and digital domain objects. For example, the scroll bar is an instrument which the user interacts with to navigate through the pages of a multi-page document (domain object). A non-instrumented alternative would be to use up and down arrow keys. An interaction instrument transforms user actions into commands and parameter adjustment.

Instruments are activated, as they come under direct user control. Beaudouin-Lafon identifies two activation methods: spatial and temporal. Spatial activation occurs through pointing at an instrument, such as a scroll bar. Temporal activation occurs due to a previous action, such as selecting a tool from a tool palette.

Interaction instruments have three properties: degree of indirection, degree of integration, and degree of compatibility. The degree of indirection measures the spatial and temporal offsets created by an instrument. The spatial offset is the distance in screen space between the instrument and the domain object. The temporal offset is the time difference between interactions with the instrument and the visible response on the object. The degree of integration compares the difference between the degrees of freedom provided by the instrument and the degrees of freedom supported by the input device. The degree of compatibility compares the similarity between the user's actions on the instrument and the object's response. Our analysis framework contains metrics that encompass indirection, integration, and compatibility.

### II.1.3 Activity Theory

Activity Theory describes the communicative and instrumental sides of human activity [Bødker 1991]. The communicative side involves communication with human beings to organize and coordinate activities. While, the instrumental side consists of actions directed towards objects or artifacts for material production. When applied to HCI, the user interface is the artifact upon which users instrument actions to accomplish tasks and communicate with the computing systems. Commanding techniques play a central role in HCI supporting specific use activity. They enable the user to perform actions either directly on an artifact or through objects such as interface elements. These actions may be communicative or instrumental. Some commanding techniques lend themselves to the communicative side (e.g. menus), while others lend themselves to the instrumental side (e.g. resize widgets).

## II.2 Analytic Framework

We develop an analytic framework for investigating the relationships among commanding techniques. This framework consists of a set of metrics for comparing the quali-

ties of each technique. We derive the metrics from analysis of prior literature and our own experiences in developing commanding techniques. The metrics are:

**Modality** What input modalities are supported for the technique?

**Hands** Is the technique bimanual or unimanual?

**Phrasing** Does the technique use phrasing? What forms of tension are involved?

**Coupling** How is user input connected to finding, selecting, and performing commands? (degree of integration)

**Learnability** How does a user learn the technique? Does it require recall or recognition?

**Expertise** Is the technique designed for novices, experts, or both?

**Scalability** How many commands can the technique support?

**Fluidity** How disruptive is executing a command to the user's current task? (degree of compatibility)

**Embodiment** How spatially similar does the human movement of the interaction map to the command (and parameter adjustment)? (degree of indirection, spatial offset)

Throughout our explanation of these metrics, we will use Kurtenbach's marking menus [1993] as an example. Marking menus are a post-WIMP technique designed to help expert users quickly invoke commands using simple marks. Marking menus are the basis for several other command techniques, and many researchers have implemented marking menus in their interactive systems. Thus, marking menus serve as a helpful exemplar for presenting our metrics.

### II.2.1 Modality

Users command a computing system through input with one or more devices. The modality metric describes what input technologies are supported by a technique. While some techniques only work with a specific input modality, others work with a number of different modalities. The number of input modalities is large, ranging from hand-based input with the mouse and keyboard to foot-based input with Multitoe [Augsten et al. 2010] to speech-based input with Siri on the iPhone. In this article, we focus on the currently more popular hand-based modalities used for interacting with desktop computers and mobile devices: mouse and keyboard, pen, multi-touch, and pen+touch (see Table 1).

An input modality is either direct (D) or indirect (I). With direct input, user actions are applied straight to an interactive surface. For example, with a direct pen display, the user places the pen directly on the display surface for input. Examples of direct input devices are smart phones (multi-touch) and Wacom Cintiq displays (pen and pen+touch). With indirect input, a user interacts separately from the surface via a sensing device, such as a mouse or trackpad. Input from the sensing device is mapped to actions on the interactive surface. Often, a proxy visual representation, such as a cursor, is used to convey the spatial mapping between the display and sensing device. Direct input is less error prone for pointing and selection tasks with pen [Forlines et al. 2007] and multi-touch [Schmidt et al. 2009] input. However, it introduces occlusion issues, as the hands partially block views of the interactive surface.

Since the introduction of the graphical user interfaces (GUIs), users of computing systems have been using pointer devices to interact with interfaces. Pointer devices specify points of input, represented by 2D coordinates. Interactive surfaces, whether direct or indirect, support a specific number of pointers for input (see Table 1).

| Devices | Directness | Pointers | Buttons |
|---|---|---|---|
| Keyboard | I | 0 | 101 |
| Mouse | I | 1 | 1-5 |
| Pen tablet (e.g. Wacom Intuos) | I | 1 | 0-3 |
| Multi-touch trackpad (e.g. Apple Magic Trackpad) | I | 5 | 0-1 |
| Pen display (e.g. Wacom Cintiq) | D | 1 | 0-8 |
| Multi-touch display (e.g. DiamondTouch) | D | 10 | 0 |
| Pen+touch display (e.g. Wacom Cintiq Touch) | D | 1 + 5 | 0-8 |

Table 1: Input properties of interactive surfaces.

### II.2.2 Coupling

When performing tasks with GUIs, users' intentions are actualized through interactions with the interface. Just as a woodworker makes intentional cuts into wood on a lathe through finding and applying the right tools, so too does a GUI user accomplish tasks (intent) by finding and performing the right commands (tools). Dourish [2001] describes how effective use of both physical and virtual tools involves *coupling* user intention to action through continual processes of engagement, separation, and re-engagement with those tools. To illustrate his meaning, Dourish draws on Heidegger's classic example of the hammer as equipment [1962] and how the orientation of the hammer to the body differs depending on context. When finding and picking up the hammer, it is separate from the body (present-at-hand). A person must consciously think about needing the hammer, finding it, and picking it up. This is analogous to a user thinking about what command to perform, finding where or how to activate that command, and activating it. When the hammer is used (e.g. to apply force to a nail), it becomes an extension of the arm (ready-to-hand). The hammer wielder is not consciously thinking about the hammer and its relation to the body, but is engaged in its use. This is analogous to a user manipulating the parameters of a command, such as increasing an element's width or changing its rotational angle. When coupling is effective, users simply manipulate parameters without thinking

about how or where to interact with commanding techniques.

#### II.2.2.1 Engagement and Separation

We identify two commons forms of coupling within commanding techniques for processes of engagement and separation:

**Persistent (P)** With persistent coupling, once a command is activated, it remains active until another command is selected. Commanding techniques that use persistent coupling tend to allocate screen space (typically near the edge) for visualizing the interface elements (such as a tool or menu bar at the top of screen). Persistent coupling allows for performing the same command repeatedly without having to reactivate the command. Since a command remains active, any accidental input by the user will also perform that command, making potentially disruptive changes (such as deleting an element).

**Transient (T)** With transient coupling, a command is deactivated once a user finishes using it. Transient coupling works well when a user needs to only perform a single command once or perform a series of different commands. However when the user needs to perform the same came repeatedly, transient coupling requires that the command be reactivated for each repeated step. Transient coupling allows for a single quasi-persistent default command that is activated initially and automatically reactivates after a user finishes another command (such as panning the view).

### II.2.3 Hands

Our review focuses on commanding techniques that use one or both hands through mechanical intermediaries or direct contact with input devices. The number of hands used can influence performance of tasks [Buxton 1986; Leganchuk et al. 1998]. Factors

that influence which hand is used for a technique include the input modality and physical constraints. Some input modalities only support a single pointer, requiring only one hand to use. Physical constraints can prevent a hand from interacting as that hand may be required to perform other tasks, such as holding the input device (e.g. smart phone).

#### II.2.3.1 Unimanual

We refer to techniques that use only one hand as unimanual (U). Any input modalities described in Table 1 can support a unimanual technique; however, some techniques may not translate well across modalities. For example, Marking Menus [Kurtenbach and Buxton 1993] were originally designed with pen tablets, where the stylus button served as modifier for activating the menu. When applying this technique to a multi-touch display, which doesn't have modifier buttons, some other form of activation must be developed.

#### II.2.3.2 Bimanual

We refer to techniques that use both hands as bimanual (B). In bimanual techniques, a division of labor exists between the two hands [Guiard 1987]. The division of labor can be symmetric (S), such as a zoom gesture, in which two fingers move simultaneously together and the distance between them determines relative zoom changes, or asymmetric (A), such as holding a modifier button with one hand while performing an operation with the mouse using the other hand [e.g. Sutherland 1964]. Guiard's kinematic chain model asserts that the asymmetric division between hands is done in series, where the actions of one hand influence those of the other hand [1987]. For example, when writing on paper, the non-preferred hand positions and orients the paper to support the preferred hand in making marks using a pen.

The pen tablet and pen display modalities do not support bimanual interaction. That is not to say that the other hand does not have a role to play in the experience (e.g. holding or positioning the device), but that the input technology does not support sensing input from

both hands.

### II.2.4 Learnability

An interaction technique's learnability is important to its successful adoption by new users. If a user has a hard time commanding a system, they will likely give up using the system. Some techniques, such as tool palettes or menus, use recognition of iconography or text to help users identify how to select a specific command. Other techniques, such as those involving hot keys or gestures, require that the user recall from memory how to select a specific command. Still other techniques, such as Marking Menus [Kurtenbach and Buxton 1993] and OctoPocus [Bau and Mackay 2008], combine recognition and recall through scaffolding [Guzdial 1994]. For techniques that use recognition, novice users often scan through the possible commands to find relevant functionality [Akers et al. 2012]. This helps them learn and discover what are the possible commands. While this can be helpful to users, it can also lead to problems in feature-rich systems where some commands have similar names or iconography [Lafreniere et al. 2015]. The novice user may select the wrong command, thinking it is the right command, producing undesired results and wasted time. Command disambiguation techniques help users detect and recover when these situations arise [Lafreniere et al. 2015].

Recognition and recall represent two ends of an axis for our learnability metric. Techniques that involve total recognition require that the user always interact with graphical representations to execute commands. Such techniques provide visual affordances for what commands a technique can perform, enabling users to search and find commands. They place less demand on human memory but at the cost of added visual complexity through the affordances. Conversely, a technique that involves total recall requires that the user remember the sequence of steps to execute a command. Unlike with recognition techniques, visual affordances are not displayed, instead the user commands by pressing

certain buttons on the input device or gesturing in a specific way. Commands that rely on recall place more demand on human memory, for remembering the sequence of steps, but tend to be faster and require less visual attention.

### II.2.5 Expertise

The expertise metric specifies whether a technique is designed to support activities of experts (E), novices (N), or both (B). Techniques for experts are designed to improve the efficiency of performing tasks through command selection that require minimal effort and introduce minimal graphical elements. For example, Kurtenbach and Buxton's marking menu enables expert users to make quick straight line marks in different directions to execute various commands [1993].

### II.2.6 Scalability

Scalability is a measure of how well a commanding technique can support an increasing number of commands. GUI environments vary in the number of commands a user can execute. Small, simple GUIs may allow for only a few commands. Large, complex GUIs may involve hundreds of commands. Thus, the adoption of commanding techniques to GUIs of varying complexity requires scalability.

### II.2.7 Fluidity

Fluidity is a measure of the disruptiveness that a commanding technique imposes on a user when performing tasks. In other words, how much cognitive and physical effort are required to transition between executing commands and performing tasks. Fluidity is a difficult quality to define and measure. Elmqvist et al. [2011] characterize fluid interaction with three components: promotes flow [Csikszentmihalyi 2009], supports direct manipulation [Shneiderman 1983], and minimizes the gulfs of evaluation and execution [Norman 1988]. States of flow involve concentration and a sense of control [Csikszent-

mihalyi 2009]. Commanding techniques that promote flow allow users to concentrate on the task at hand, minimizing steps required to execute a command and avoiding added demands on attention. They provide mechanism for directly manipulating parameters of commands. Fluid interactions support scaffolded learning [Guzdial 1994], enabling usage with minimal knowledge. Fluid interactions make clear the state of the system and the allowable actions, using visual feedback and affordances. From this conceptualization of fluid interactions, we derive a fluidity metric for commanding techniques based upon the following:

- number of interactive steps required to execute a command ( $S$=[1..$N_S$] )
- interactions are in-place at the point of focus ( $F$ = [yes, no] )
- number of new visual elements added ( $V$=[0..$N_V$] )
- enables parameter adjustment ($P$ = [yes, no] )

We represent values for the fluidity metric as a vector $< S, F, V, P >$ , where $S$ is the number of steps, $F$ is for in-place activation, $V$ is the number of new visual elements added, and $P$ is for parameter adjustment. For example, take the marking menu technique, which when used in expert mode has a high fluidity according to our metric. The number of steps required is two (S = 2) — press a modifier (e.g. pen barrel button) to activate commanding and then make a quick mark. Marking menus activate in-place, requiring minimal travel distance for the hands and eyes (F = yes). In expert mode, only one element is added, a line representing the mark (V = 1). However in novice mode, the radial menu is presented, increasing the number of elements by 8 for a full menu (V = 9). Marking menus do not enable parameter adjustment, only command selection (P = no).

### II.2.8 Embodiment

We humans sense, perceive, and interpret the world around us through our bodies [Merleau-Ponty 1965]. For example, we understand that we can sit on objects with flat tops of certain heights that we would not categorize as chairs (such as a low wall) because of the physical relationship between our bodies and the objects. In turn, our cognitive processes and mental models are also based in and connected to our bodies — that is *embodied* [Glenberg 1999]. This embodiment entails our perceived understandings of interactions with computing systems [Dourish 2001].

The embodiment metric describes how involved movements of the body are connected with commanding. We suspect that the embodiment of a commanding technique will promote learnability and fluidity when body movement is more directly mapped to command action. For example, the two finger pinch gesture for zooming a view or enlarging an element provides a direct spatial mapping between the finger contact points and the parameters of the command, resulting in a high level of embodiment.

## II.3 Commanding Techniques

We review and analyze techniques to command interactive systems using our framework. We examine both WIMP (Table 2) and post-WIMP (Table 3) techniques designed for at least one of the following input modalities: mouse and keyboard, pen, multi-touch, and pen+touch input.

### II.3.1 WIMP

Users execute commands in a WIMP interface using a single pointer (typically via mouse input). Users define contexts for interaction, such as a window or element, using this single pointer. Then, they invoke operations using a commanding technique, such as a menu (the "M" in WIMP). Menus can reside in bars located around the edges (typically

the top) of a window or display. Alternative to a menu bar is a floating tool palette, which contains a set of buttons with icons for enacting modal states. Unlike menu bars, the user may move floating tool palettes, placing them closer to areas of interaction. To keep the tool palette close, the user most repeatedly move it as she works in different areas. Beyond menu bars and tool palettes, in-place techniques allow for command invocation to occur at the user's current point of focus, such as on interactive elements.

Table 2 presents analysis of WIMP commanding techniques using our framework. WIMP techniques tend to be persistent in their coupling. They support learnability and discovery through recognition of icons and text. WIMP techniques are rarely activated in-place, and typically present a set of commands ($M$), adding visual elements, such as a text menu item, for each command.

#### II.3.1.1 Bars and Palettes

The menu bar is one of the most common commanding techniques for WIMP interfaces. A menu bar presents a set of commands, organized in hierarchical lists. At the highest level, a set of categories (e.g., File, Edit, Window, and Help), represented by text and listed horizontally, subdivide the set of possible commands into semantic groups. Selecting a top-level category presents a drop-down menu, listing a set of commands vertically. Commands can be further subdivided using cascading menus [Cockburn and Gin 2006], in which a new sub-menu is presented to the right of the currently selected menu item, allowing the user to access additional commands by moving the pointer to the right onto the cascading menu. A menu bar is always present, supporting command discovery and learnability. However, as the number of commands grows large in a single drop-down menu, the linear list requires extensive scrolling, slowing down command performance. Fisheye menus use a focus+context fisheye visualization [Furnas 1986] to help users deal with a large number of menu items [Bederson 2000]. Fisheyes expand the visual size of in

| Technique | Modality | Coupling | Hands | Learnability | Expertise | Scalability | Fluidity | | | | Embodiment |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | $S$ | $V$ | $F$ | $P$ | |
| Menubar with drop-down menus | Mouse (I) | P + T | 1 | Recognition | N | $M \times L$ | 4 | $M$ | no | no | None |
| Split Menus [Sears and Shneiderman 1994] | Mouse (I) | P + T | 1 | Recognition | N | $M \times L$ | 3+1 | $F+M$ | no | no | None |
| Fisheye Menus [Bederson 2000] | Mouse (I) | P + T | 1 | Recognition | N | 100 | 4 | $M$ | no | no | None |
| Toolbar | Mouse (I) | P | 1 | Recognition | N | $M \times L$ | 4 | $M$ | no | yes | None |
| Pie Menu | Mouse (I) | T | 1 | Recognition | N | $8^M$ | 3 | 8 | yes | no | None |
| Tool Palette | Mouse (I) | P | 1 | Recognition | N | $M$ | 4 | $M$ | no | no | None |
| Springboard [Hinckley et al. 2006] | Pen (D) | P | 1 | Recognition | N | $8^M$ | 3+1 | $M$ | yes | no | Low |
| LassoMenu [Agarawala and Balakrishnan 2006] | Pen (D) | T | 1 | Recognition | N | $8^M$ | 2 | $M$ | yes | no | Low |
| Tracking Menu [Fitzmaurice et al. 2003; Hinckley et al. 2007] | Pen (D) | P | 1 | Recognition | N | $M$ | 1+1 | $M$ | yes | yes | Low |
| PieCursor [Fitzmaurice et al. 2008] | Mouse (I) | P | 1 | Recognition + Recall | N+E | 8 | 2 | 0 | yes | yes | Low |
| Stacked Half-Pie menus [Hesselmann et al. 2009] | Touch (D) | T | 1 | Recognition | N | near infinite | 3 | $M \times L$ | no | no | Low |
| CommandMaps [Scarr et al. 2012] | Mouse + Keyboard (I) | T | 2 | Recognition + Recall | N+E | $M \times L$ | 3 | $M \times L$ | no | no | None |
| ExposeHK [Malacria et al. 2013] | Mouse + Keyboard (I) | P | 2 | Recognition + Recall | N+E | $M$ | 1 | $M$ | no | no | None |

Table 2: Framework analysis of selected prior WIMP commanding techniques.

focus menu items to support target acquisition.

The hierarchical representation, combined with the ability to scroll drop-down menus, allows menu bars to scale to support a large number of commands. However, human attention does not scale, as the number of menu items grows large. Split menus place a subset of menu items at the top of the drop-down menu that contains the most frequently used commands [Sears and Shneiderman 1994]. By placing these menu items at the top, the user can quickly find and execute commonly performed commands. However, split menus dynamically change the structure of the menu, which can affect performance for expert users who have memorized a menu structure [Tsandilas and schraefel 2007]. Bubbling menus enable accessing frequently used menu items, without changing the menu structure or the visual size of menu items, by employing a dynamically resizable cursor and motion-aware techniques [Tsandilas and schraefel 2007]. Bubbling menus provide two views for drop-down menus in a menu bar. The default view operates like a normal drop-down menu. The alternative view, activated with a drag on a top-level menu category, enables selection of only the frequently used items via a bubble cursor [Grossman and Balakrishnan 2005] that selects the nearest item. By using a bubble cursor, the user can quickly select items by simply moving towards the desired item, without having to place the cursor directly over the desired item. Enlarged activation area menus increase the motor space size of menu items that activate cascading menus, allowing for shorter diagonal paths from parent to cascade items [Cockburn and Gin 2006]. For touch input on tabletops and personal workstations, Stacked Half-Pie Menus provide hierarchical representations using concentric half-circles for nested levels [Hesselmann et al. 2009]. Commands are selected by dragging along a path from the center outward. Each half-circle may be rotated to reveal more commands, allowing for an infinite number of commands to be selected.

Another common WIMP technique is the tool bar. Similar to a menu bar, the tool bar presents a set of commands as a strip located near the edge of a window or display

(typically at the top). However, unlike a menu bar, a tool bar combines text and icons to represent commands. A tool bar may also contain widgets for modifying parameters of commands (e.g., such as specifying the font size via a text box). Using tabs and frames, a tool bar supports hierarchical representations similar to a menu bar. While hierarchical representations help novice users find the desired command, the extra time required to navigate these hierarchical structures is not beneficial to experts you already know where a command is located [Cockburn et al. 2007]. CommandMaps seeks to support expert users by flattening hierarchical representations of a tool bar in a consistent manner. Through repeated use, spatial memory builds up for frequently-used commands [Scarr et al. 2012].

WIMP interfaces for visual design environments (e.g. Adobe Illustrator) employ tool palettes. These tool palettes are initially positioned around the edges of a window or display. A tool palette consists of a set of tools, represented by buttons with icons. Selecting a tool in the palette activates a modal state. Actions performed in the design space operate within the selected modal state. For example, selecting the pencil tool allows mouse drags in the design space to produce thin strokes that follow the path of the cursor while the left-mouse button is held down.

One of the problems with modal states is mode errors, in which the user forgets what mode they are in and accidentally performs an unintended operation [Sellen et al. 1992]. Quasimodes seek to overcome mode errors by reverting back to the default state after an operation is performed or a button is released. Springboard enables performing modal commands in quasimodes via a button press on the keyboard with the non-preferred hand [Hinckley et al. 2006]. With the command key pressed, the user can then select a command in a tool palette or use a marking menu [Kurtenbach and Buxton 1993] (see Section II.3.2.1). While the key remains pressed, the selected command is active and all actions with the pointer execute functions of that command. Releasing the button ends the quasimode, reverting back to a default modal state.

#### II.3.1.2 In-Place

A common in-place technique for selecting commands is the context menu (typically activated with a right mouse button click). A context menu enables executing commands on specific elements, such as specifying the visual stacking order of overlapping elements with commands like Bring to Front and Send to Back. Visually, a context menu is similar to a drop-down menu in a menu bar. Menu items are represented as text in a vertical list. Context menus can be hierarchical.

Pie and radial menus provide alternatives to linear list for representing menu structures in-place [Callahan et al. 1988]. Pie menus arrange menu items in a circle around the activation point. Each item is equal distance from the single pointer upon menu activation. Typically, a pie menu has at most eight items. However, hierarchical representations are possible where selecting one of the eight items brings up a new pie menu, again centered around the current pointer position. LassoMenu is a pie menu activated after a lasso selection [Agarawala and Balakrishnan 2006]. The selection defines the context on which to perform the selected commands. By allowing selection before hand, the context of interaction is broaden beyond a single element. Radial menus, like pie menus, arrange items around an activation point, but radial menus do not constrain placement of items to within a circle. Hotbox is radial menu where each item is a menu bar that can bring up drop-down menus in-place [Kurtenbach et al. 1999]. Hotbox combines the advantages of semantic grouping provided by menu bars with the in-place activation supported by radial menus.

Pie menus are transient, appearing when activated and disappearing when no longer needed. Some in-place techniques are always present, rather than transient. These techniques follow the movement of the pointer, so that they remain positioned in-place. For example, a tracking menu provides a region containing a set of interface widgets for ex-

ecuting commands that the pointer accesses by moving over those widgets, just like any other menu technique [Fitzmaurice et al. 2003; Hinckley et al. 2007]. However, when the pointer moves to the edge of the tracking menu region, the menu moves with the pointer, dragging the menu along, so that it stays with the pointer. PieCursor is a form of tracking menu that shrinks its size down to a pie-shaped cursor [Fitzmaurice et al. 2008]. Commands are invoked by first moving the pointer in one of eight directions (just like a pie menu). Based upon the direction moved, one of the slices in the pie-shaped cursor, representing a command, is selected. Then, the user presses down a mouse button to invoke the selected command. PieCursor combines pointing and command selection, as opposed to a pie menu, in which the user first points to define the desired activation point, then selects a command following the pie menus activation. Later, we will encounter techniques that combine command selection and parameter adjustment [e.g. Pook et al. 2000].

#### II.3.1.3 Hotkeys

To support expert use, many WIMP interfaces employ keyboard accelerators or hotkeys to invoke commands without the need for accessing a menu. By pressing a combination of keys or a specific sequence of keys, users can invoke commands. Hotkey sequences typically involving pressing a modifier key (e.g., ctrl or ⌘) along with an alphanumeric key (e.g., C or 3). The pointer may be used to first select elements on which to perform commands. One of the primary problems with hotkeys is that they are not discoverable on their own. Often, these hotkeys are presented next to menu items to support novice users learning these shortcuts. However, users may ignore these labels, as they have already selected the desired commands [Grossman et al. 2007]. An alternative strategy for showing hotkeys is to use a feed-forward technique that overlays hotkey combinations on tool bar icons when a modifier key is pressed [Malacria et al. 2013]. In this case, the user is showing intention to use hotkeys by pressing the modifier key, but unlike with text in a

menu bar, the user has not already selected the command. Users learn through rehearsal rather than recall. Grossman et al. observed that auditory feedback and disabling menu items, so that hotkeys were required to invoke commands, both helped participants learn and effectively make use of hotkeys.

### II.3.2 Post-WIMP

Post-WIMP commanding techniques include in-place menus, which provide pie-menu style interfaces for novice users, while also supporting faster command selection through quick gestures. Methods for activating these in-place techniques vary depending on the input modality supported. In addition to the quick gestures used by in-place menus, more complicated gestures may also be used to command interactive systems without involving a menu. Without the discoverability supported by menus, these techniques must use alternative strategies to make visible how to invoke commands. Some post-WIMP techniques combine command selection and parameter manipulation.

Table 3 presents an analysis of post-WIMP commanding techniques using our framework. Post-WIMP techniques tend to be transient in their coupling. They typically support combined recognition and recall for learnability, in which a novice user discovers commands and learns gestures for invoking those commands as she transitions to performing gestures via recall as an expert. Post-WIMP commanding techniques are mostly activated in-place, and they add few new visual elements, particularly in expert mode. These techniques make use of spatial movements and gestures to embody interactions at a higher level than the WIMP techniques presented previously.

#### II.3.2.1 Menus and Tool Palettes

Arguably the most well known commanding technique for post-WIMP interfaces is the marking menu [Kurtenbach and Buxton 1993]. Marking menus have two modes: a novice mode, which functions similar to a pie menu, and an expert mode, which allows

| Technique | Modality | Coupling | Hands | Learnability | Expertise | Scalability | Fluidity | | | | Embodiment |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | $S$ | $V$ | $F$ | $P$ | |
| Toolglass [Bier et al. 1993] | Mouse+Trackb (I) | P | 2 | Recognition | N | $M$ | 3 | $M$ | yes | no | Medium |
| Marking Menu [Kurtenbach and Buxton 1993] | Pen (D) | T | 1 | Recognition (N) + Recall (E) | N+E | $8^N$ | 2 | 1 | yes | no | Low |
| Control Menu [Pook et al. 2000] | Mouse (I) | T | 1 | Recognition (N) + Recall (E) | N+E | $8^N$ | 2 | 1 | yes | yes | Medium |
| FlowMenu [Guimbretiére and Winograd 2000] | Pen (D) | T | 1 | Recognition (N) + Recall (E) | N+E | $8^N$ | 3 | 1 | yes | yes | Medium |
| Whole-hand gestures [Wu and Balakrishnan 2003] | Touch (D) | T | 2 | Recall | N | memory | 1 | 0 | yes | yes | High |
| Bimanual Marking Menu [Odell et al. 2004] | Mouse+Mouse (I) | T | 2 | Recognition (N) + Recall (E) | N+E | $8^N$ | 2 | 1 | yes | no | Low |
| Pigtail [Hinckley et al. 2005] | Pen (D) | T | 1 | Recognition (N) + Recall (E) | N | $8^N$ | 3 | $M$ | yes | no | Low |
| Hover widgets [Grossman et al. 2006] | Pen (D) | T | 1 | Recognition (N) + Recall (E) | N+E | near infinite | 2 | 0 | yes | no | Medium |
| Command strokes [Kristensson and Zhai 2007] | Pen (D) | P | 1 | Recall | E | memory | 2 | 28 | no | no | Low |
| Flower menu [Bailly et al. 2008] | Pen (D) | T | 1 | Recognition (N) + Recall (E) | N+E | $56^N$ | 2 | 1 | yes | no | Low |
| GestureBar [Bragdon et al. 2009] | Pen (D) | P | 1 | Recognition | N | $M$ | 3 | 3 | no | no | Low |
| Finger-count Shortcuts [Bailly et al. 2010] | Touch (D) | P | 2 | Recognition | N | 25 | 2 | $M$ | no | no | Low |
| Pen+Touch Quasimodes [Matulic and Norrie 2013] | Pen+Touch (D) | T | 2 | Recall | E | memory | 3+2 | 1 | yes | yes | High |

Table 3: Framework analysis of selected prior post-WIMP commanding techniques (in expert mode for techniques that support both novice and expert users).

for making directional marks to execute commands. The menu is activated in-place at the pointer's current position, using a button press, which could be a mouse button, pen barrel button, or keyboard key. If the pointer remains still for a short timeout, then the marking menu goes into novice mode, and a pie menu is displayed, with at most eight items. Dragging the pointer towards one of the menu items, draws a stroke between the activation point and the current pointer position. The stroke provides visual feedback to let the user know that they can activate menu items in expert mode by making marks. Following activation, the user can immediately move the pointer in one of eight directions (N, NE, E, SE, S, SW, W, or NW), producing a mark. The menu is now in expert mode. By making a quick mark, the user activates one of the eight commands, without showing the pie menu.

Marking menus allow for quickly executing commands in-place. They support hierarchical menu structures with compound marks that form "zig-zag" patterns. For example, one can mark NE, then S, to select a nested command. However, selection error issues arise with deeper nested structures, limiting the number of directions to four, instead of eight [Kurtenbach and Buxton 1993]. Rather than perform a compound mark, nested command selection is possible using multiple single marks where the pen is lifted (or mouse button released) after each mark [Zhao and Balakrishnan 2004]. Using curved marks, a marking menus can be expanded to support 20 items at each level [Bailly et al. 2008]. Bimanual marking menus allow the left (non-preferred) hand to specify the command through either a mark in expert mode or selecting a menu item from the pie menu in novice mode [Odell et al. 2004]. The right hand manipulates parameters for the command, once selected.

Several marking menu techniques combine command selection and parameter adjustment. Control menus support command selection like marking menus, but once a command is selected, further movements of the pointer (while the mouse button remains held) can manipulate parameters of that operation [Pook et al. 2000]. For example, after a user

selects the zoom command with a control menu, continuing to drag outwards, increases the zoom level, while dragging inwards decreases the zoom level. The final zoom level is defined when the mouse button is released. Control menus allow for specifying parameters that can be mapped to a 1D or 2D space. They support continuous adjustment of parameter values.

FlowMenu is another marking menu technique that combines command selection and parameter adjustment [Guimbretiére and Winograd 2000]. Unlike control menus, FlowMenus do not require an arbitrary threshold for command activation. Instead, they rely on marks that loop outwards and return back to the menu center to activate commands or make parameter value selections. Rather than providing continuous parameter adjustment, FlowMenus allow for discrete value selections, including text characters. Thus, FlowMenus can be used for text entry, such as specifying a specific zoom level. Multiple values selections can be explored through repeated loops.

Toolglass is a bimanual commanding technique, in which the left hand positions a see-through tool palette, while the right moves a pointer to select a command [Bier et al. 1993]. Both hands can then function to manipulate parameters of the command. For example, the user can draw a rectangle by first positioning the toolglass so that the Draw Rectangle tool is in the desired location, then using the cursor (controlled by the right hand) to select the tool and drag the cursor outward. The toolglass disappears, and a drawn rectangle appears. The right hand now defines the bottom-right corner of the rectangle, while the left hand controls the top-left corner. Toolglass applies Guiard's principles of Right-to-Left spatial reference and Left Hand Precedence [1987]. The left hand provides a frame of reference for the right hand actions through positioning the toolglass. The toolglass must first be positioned before the right hand can invoke the desired command. Similar to a floating tool palette, a toolglass is always-present providing discoverability of commands.

A common interaction pattern is to select a set of elements and perform a command

on those elements. For pen input, with only a single pointer for input (no buttons), several approaches have been developed to delimit element selection from command selection. Gestures, such as a pigtail (small loop) [Hinckley et al. 2005], have been used to activate in-place menus, such as marking menus. The small pigtail shape is distinguishable from larger lasso selections. LassoMenu presented small affordances at the start of a lasso selection, which when entered, brings up a pie menu [Agarawala and Balakrishnan 2006]. Hover widgets use the sensing capabilities of direct pen input devices to track the pen hovering above the interactive surfaces to provide a layer for command selection [Grossman et al. 2006]. The user gestures with the pen in the tracking state, then presses the pen down to invoke the selected command. Visual feedback conveys the currently selected command before the user presses the pen down, allowing the user to cancel the actions by moving the pen away.

### II.3.3 Gestures

In addition to in-place menus, commands in post-WIMP interfaces can also be selected using gestures. For example, the quick, directional marks made in the expert mode for marking menus are an example of command selection with gestures. Embodied post-WIMP interfaces seek to take advantage of the human body's ability to express and convey meaning using more than just a single point of input. Wu and Balakrishnan [2003] developed whole-hand gestures for multi-touch interaction in furniture arrangement design environment, using not only the finger tips—traditionally used in multi-touch input—but also the sides of the hand and the palm. They used metaphor to map commands to gestures. For example, a vertical hand on its side is used to sweep elements across a tabletop surface in order for users to make space for working. This action is similar to how one would quickly brush aside loose small physical items on a table to make space for working. In another example, a horizontal hand on its side is used to create a private space, just

as when playing cards, one uses a similar gesture to conceal her cards from other players. Wilson et al. [2008] used physics as a metaphor in gestural interaction, allowing users to directly manipulate elements similar to how humans interact with physical objects. The physics metaphor allows for same manipulation commands to be performed using different gestures. The gesture recognition is based on physics simulations rather than the number, size, and relative positions of touches that is commonly used in other gesture-based systems. Matulic and Norrie [2013] developed pen+touch quasimodes, in which the left-hand gestures define commands to perform, while actions of the right hand select command targets and manipulate parameters. In their approach, the left hand gestures in an edge-constrained area on the left side of a tabletop surface.

One of the problems with gestural interaction is how to support discoverability and learnability of commands and the gestures required to invoke those commands. GestureBar leverages the WIMP toolbar as a means for revealing gestures and helping users practice those gestures [Bragdon et al. 2009]. Commands are not executed through GestureBar. Instead, when a command button is clicked, an animated demonstration of the gesture is presented, along with an area for the user to practice the gesture.

Another approach for gesture learning is self-revealing techniques. Octopocus is a self-revealing technique that uses visual feedforward representations to show users the possible gestures [Bau and Mackay 2008]. Feedforward representations convey possible interaction steps for a user. With gestural interaction, these representations show direction and position of input movement to perform gestures for specific commands. Octopocus presents these feedforward representations when a pointer is pressed (pen or touch down or mouse button down).

The choice of what gesture to perform which operation is important for interaction designers. Researchers have used gesture elicitation studies to gather design possibilities from potential users [Frisch et al. 2009; Morris et al. 2010; Wobbrock et al. 2009]. In

these studies, participants—typically users that the researchers are seeking to support—are asked to come up with gestures to perform specific commands. An advantage of this approach is that the participants do not need to consider limitations of sensing technologies. As a result, gestures elicited tend to be more favored by users than those designed by professionals, which are more physically and conceptual complex [Morris et al. 2010]. However, legacy bias from participants' past experiences using interactive systems influence the gestures elicited [Morris et al. 2014]. The participants' familiarity with WIMP-style interaction often produces gestures that are fixated on single pointer input involving menus, icons, and buttons.

Command strokes provide a gestural alternative to drop-down menus for pen-based interactions [Kristensson and Zhai 2007]. The user gestures over an on-screen keyboard tracing out the set of hotkeys for performing a command. For example, to perform the copy command, the user places the pen down on ctrlvirtual key and traces a path over the virtual C key. The user must know the keyboard shortcut for performing a command in order to use command strokes.

Finger-count shortcuts are a gestural technique for invoking frequently used commands from drop-down menus in a menubar with multi-touch gestures [Bailly et al. 2010]. The left hand selects the drop-down menu by placing a number of touches. The number of touches equals the position of the drop-down menu. In other words, one touch selects the left-most drop-down menu. Two touches selects the second left-most drop-down menu, and so on. The right hand then selects with the number of touches, a menu item from the currently activated drop-down menu. Again, the number of touches equals the position of a menu item within the drop-down menu. Radial-stroke shortcuts, similar to finger-count shortcuts, use directional strokes (á la marking menus) to select a drop-down menu via the left hand and an menu item within that drop-down menu using the right hand [Bailly et al. 2010]. These approaches provide post-WIMP gestural interaction as shortcuts for WIMP

interfaces.

## II.4 Post-WIMP Systems

We now examine the use of commanding techniques and phrasing in prior post-WIMP systems (see Table 4). We look at systems that support indirect input, direct pen-based interaction, and the pen+touch modality.

### II.4.1 Indirect Input

T3 is a design environment involving transparency, two-handed interaction, and tablets [Kurtenbach et al. 1997]. The design goal for T3 is to address human attention issues created by WIMP interfaces, such as tool palettes and menubars, in which the user must divide their attention between the design work—their context—and the interface elements for command selection. T3 uses two Wacom tablets with stylus and puck for indirect bimanual input. A toolglass is positioned using the left hand puck, while the right hand, with stylus or puck, performs command selection and parameter adjustment. The user may activate a marking menu at the top of the toolglass to switch between different toolglass palettes. T3 demonstrates how graphical widgets of WIMP interfaces can be replaced with graspable widgets of a post-WIMP interface.

CPN2000 [Beaudouin-Lafon and Lassen 2000] is a redesign of a graphical editor and simulator of Colored Petri Nets (CPNs) [Jensen 2013]. Command selection in CPN2000 is performed through floating palettes, direct manipulation, toolglasses, and hierarchical marking menus. CPN2000 supports one quasimode for activating direct manipulation of elements. When the user has a tool selected via the floating palette, a long click (i.e., pressing the mouse button and waiting for a short delay) activates a direct manipulation quasimode, in which the user can drag to move elements. Ending the long click ends the quasimode, returning back to the previous tool. Beaudouin-Lafon and Lassen found that simply juxtaposing the different commanding techniques was problematic. Additional

| System | Modality | Phrasing | Quasimodes | Transient Interfaces | Contextual Operations |
|---|---|---|---|---|---|
| T3 [Kurtenbach et al. 1997] | Pen+Puck (I) | Left Hand | 0 | 1 (toolglass) | 2 (marking menu, Two handed stretchies) |
| CPN2000 [Beaudouin-Lafon and Lassen 2000] | Mouse+ Trackball (I) | Left Hand | 1 (long click) | 1 (toolglass) | 2 (marking menu, Two handed stretchies) |
| interactive wall [Guimbretière et al. 2001] | Pen (D) | Button + Gesture | 0 | 0 | 3 (ink, rotation, FlowMenu) |
| BumpTop [Agarawala and Balakrishnan 2006] | Pen (D) | Gesture | 2 (pile browsing, drag'n'cross insertion) | 1 (pile widgets) | 3 (LassoMenu, PressureLock, Lasso'n'Cross) |
| InkSeine [Hinckley et al. 2007] | Pen (D) | Button + Gesture | 1 (gesture mode) | 1 (pigtail menu) | 2 (Tracking Menu, scroll ring) |
| Manual Deskterity [Hinckley et al. 2010] | Pen+Touch | Left Hand | 0 | 2 (Color Palette, Bezel Menu) | 4 (Stapler, X-acto Knife, Copy, Tape Curve, Brushing) |
| Document editing application [Matulic and Norrie 2013] | Pen+Touch | Left Hand | 5 (text, shape, command, selection, content association) | 1 (Page Format) | 4 (Copy, Create New, Text alignment, Delete) |
| Emmâ (Chapter III) | Pen+Touch | Left Hand | 3 (Selection, Translucence, Edge Blending) | 2 (Ink Palette, Ink Menu) | 4 (Perspective, Duplicate, Straight Line, Delete) |
| MoodBoard (Chapter VI) [Webb et al. 2016c] | Pen+Touch | Left Hand | 1 (Straightedge) | 3 (ChopZ, Ink Palette, Group Sheets) | 1 (Single-Touch Translate) |

Table 4: Comparison of post-WIMP systems by modality, phrasing, and use of quasimodes, transient interfaces, and contextual operations.

interaction design is required to support combining these different techniques. T3 avoided this issue by using toolglass as the primary method for command selection. Marking menus served to switch between different toolglass palettes.

Both T3 and CPN2000 provide early examples of how bimanual interactions with post-WIMP interfaces can support design processes. The creators of these two systems were able to design bimanual interactions using two pointer input devices. While two-pointer device setups were not common when these systems were developed, they provided motivation and direction for the design of future systems. Presently, the growing availability of new input sensing devices that support pen+touch interaction, provide space to further advance the design ideas originally brought up by these systems.

### II.4.2 Pen-based

HCI researchers have long investigated pen-based input to support interaction with computing systems [Sutherland 1964]. Commanding these systems, which use only the pen, for post-WIMP interfaces relies on gestures to distinguish strokes for selection or inking with those to invoke commands. Here, we look at three different examples of how commands are invoked in pen-based systems: an interactive wall environment [Guimbretière et al. 2001], BumpTop [Agarawala and Balakrishnan 2006], and InkSeine [Hinckley et al. 2007].

Guimbretière et al. [2001] developed an interactive wall environment to support free-hand sketching while working with high resolution materials. Commanding in their environment is performed through a FlowMenu, which allows for text entry and parameter adjustment along with command selection, removing the need for keyboards, prompts, and dialog boxes for specifying filenames of materials or specifying attributes of visual elements. The FlowMenu is activated by pressing a button on the pen. Pressing and holding the button, phrases FlowMenu interactions. When the button is not pressed, the pen

marks on the surface (pen down, drag, then up) perform contextual operations depending on which type of element the drag was performed. For example, a mark on 3D object, rotates that object. A mark on a 2D image creates digital ink. Ink is automatically merged with 2D objects, when they are sketched over. Marks beginning on the wall background invoke different commands depending on the path followed. If the path moves across a 2D object, the ink is merged with the object. If the path doesn't cross any objects, a new ink element is created.

BumpTop is a desktop organization environment that combines physically-based modeling with various interaction and visualization techniques [Agarawala and Balakrishnan 2006]. Commands in BumpTop are invoked on a lasso selection using the either the LassoMenu or the Lasso'n'Cross technique. The Lasso'n'Cross technique allows the user to create a pile of desktop documents. During lasso selection, the user crosses through the center of the lassoed region to indicate that this selection is to create a pile, rather than for performing a command in the LassoMenu. Hovering over a pile with the pen brings up a menu of pile widgets, allowing the user to invoke commands to change the visual representation of a pile. Using these widgets, the user may precisely insert elements into a pile by dragging an element and crossing the drag through one of these pile widgets. The result is a quasimode for inserting elements into piles. The pile temporarily changes its representation. The user drags the element to the desired position in the pile. Lifting the pen or dragging the element out of the pile, ends the insertion quasimode.

InkSeine is an active note taking application that supports inking, selection, and commanding with the pen [Hinckley et al. 2007]. The pen by itself makes ink. The user holds down either a left hand keyboard button or a stylus button to activate the gesture quasimode. In the gesture quasimode, the user invokes commands by either performing a pig tail gesture or lasso selecting ink strokes and then performing a pigtail gesture. The pigtail gesture activates a marking menu, in which the user selects commands. Users

create breadcrumbs—small icons that represent potential search queries—by drawing an incomplete lasso around some handwritten ink notes. This quick interaction allows users to create a search context for later examination without having to formally engage in the search process at this time. The user can continue engaging in note taking, with their attention still focused on the note taking task, and later return to the breadcrumb to search, when their tasks warrant it. Pen marks on the breadcrumb can invoke marking menu commands, such as open a search query. When browsing search results and documents, a tracking menu is used for commanding. The user scrolls search results and documents using a scroll ring [Moscovich and Hughes 2004] in the tracking menu, in which circular motions are mapped to vertical scrolling motions. Buttons in the tracking menu invoke other commands.

### II.4.3 Pen+Touch

Manual Deskterity is a pen+touch scrapbook application for tabletop surfaces [Hinckley et al. 2010]. Commands in Manual Deskterity are invoked and parameters manipulated via the combination of left hand gestures and right hand pen input. Commands are phrased through kinesthetic tension, in which the left hand gesture is maintained throughout performing an operation. For example, the copy command is executed by holding a scrapbook element with a left-hand touch, then performing a peel-off gesture with the pen in the right hand. Manual Deskterity primarily makes use of contextual operations and transient interfaces. A single finger touch with the left hand is used in three different contextual operations—brushing, tape curve, and color palette—depending on what element is touched. Touching an image activates the brushing command, which changes the brush style for ink to match the contents of the object touched with the left hand. Touching an ink stroke activates the tape curve command, allowing the user to draw a mix of straight lines and curves. Tapping an empty area of the canvas activates the color palette, allowing

the user to engage in finger painting with different colors. In the case of tape curve, the left hand not only phrases the command, but also manipulates a parameter of the operation, the pivot point for the next tape segment.

Matulic and Norrie [2013] developed a document editing system for interactive pen+touch tabletops. They used a combination of quasimodes and contextual operations to support document authoring. For quasimodes, the left hand gestures in an edge-constrained area, invoking different quasimodes depending on the posture of the left hand. Then, the pen, in the right hand, performs operations and manipulates parameters. Lifting the left hand while the pen is still interacting cancels the command, reverting any changes. Contextual operations for copying existing pages and creating new ones are performed by holding one or two left-hand touches on an existing page and dragging with the pen. Chopping gestures with the sides of the hand are used to specify text alignment for a paragraph. For example, a chopping in the center of a paragraph, produces a center alignment. Chopping on both sides produces a justified alignment. Similar to Emmâ (Chapter III) and MoodBoard (Chapter VI, [Webb et al. 2016c]), their document authoring environment mixes bimanual and uni-manual interactions using quasimodes, transient interfaces, and contextual operations. Kinesthetic tension through holding left hand gestures provides phrasing for commands invoked with bimanual interactions. They, like us, rely on gestural interaction, rather than menu or tool palette techniques, for commanding their system.

## II.5 Discussion

Our analysis of commanding techniques identifies key differences in terms of coupling, fluidity, and embodiment between WIMP and post-WIMP interfaces, and examines how the use of quasimodes, transient interfaces, and contextual operations in post-WIMP systems supports phrasing beyond our work.

### II.5.1 Coupling and Discoverability

WIMP interfaces, such as menubars and toolbars, were designed to support discoverability of a large command set by novice users. Many post-WIMP commanding techniques were designed to support expert users, such as visual designers. The design goal for these post-WIMP techniques was to keep the user focused on her current context, which for a designer involves creative cognitive processes. WIMP techniques have persistent visual interface elements, typically located near the edge of the display, which can distract or require the user move away from the context of her interactions, in order to invoke commands.

Alternatively, post-WIMP techniques rely on transient interface elements that appear in-place, at the user's focus point. They seek to minimize adding new interface elements. The cost of transient interfaces is that they become less discoverable. To address this issue, some techniques, such as marking menus, provide novice and expert modes. In the novice mode, additional interface elements are presented to support discoverability of commands. However, activating the novice mode is still not easily discoverable in these interfaces. For walk-up-and-use scenarios, post-WIMP techniques that lack discoverability, even for simply activating a novice mode, are ill advised. For design environments, where the user will have some documentation or assistance in learning to command the system, techniques that help maintain flow [Csikszentmihalyi 2009] and context are more important than discoverability.

A pen+touch interactive surface affords [Gibson 1979; Norman 1988] touching and pressing the pen against it. We suspect that new users to a design environment will pick up the pen and begin making marks on the surface, as this is a familiar human practice with pen and paper. In our design studio evaluation of Emmâ, we observed participants engaging in sketching as their first actions, providing an icebreaker before performing

more complex commands (see Chapter IV). Users will likely also touch elements on the screen to try and directly manipulate them. These interactions provide entry points for post-WIMP interface designers to support command discoverability. Using feedforward techniques, interface designers can visualize a set of possible gestural commands [e.g., Bau and Mackay 2008]. These visualization would differ based upon what elements the user touches, supporting contextual operations without overloading the user with all the possible gestures. Edge-constrained gesture areas provide visual affordances for where to touch to activate quasimodes.

### II.5.2 Fluidity

Post-wimp techniques to command interactive systems tend to employ more fluid interaction principles over their WIMP counterparts. The gestural qualities of post-WIMP techniques allow for in-place activation of commands with adding only minimal visual content. For example, a marking menu in expert mode only adds one visual element—the mark. By allowing commanding in-place, the number of steps required is also reduced, since the user does not have to move away (e.g., to a menubar at the top of the screen) to select a command, and then back to adjust parameters. However, post-WIMP techniques lack the scalability of most WIMP techniques. Marking menus support eight possible commands at each hierarchical level, while a drop-down menu can support many more, especially when using a fisheye representation [Bederson 2000]. Kurtenbach et al. [1999] sought to address this scalability issue by providing in-place menubars with drop-down menus in Hotbox. While this reduces the distance the user needs to travel, it still introduces many new visual elements and requires the user find the desired command, rather than simple gesture using marks with the pen or hand postures with touch. A solution that supports improves both fluidity and scalability is unlikely to exist. Yet, one possible option is to provide two methods for performing commands, gestures for frequently used

commands and in-place menu techniques, like Hotbox, for all commands. This is only required for interactive systems that have a large command set. Systems with a smaller command set, such as Emmâ, can take full advantage of post-WIMP gestural interfaces.

### II.5.3 Embodiment: Sensory Input

We observe differences in embodied interaction design for WIMP versus post-WIMP commanding techniques. Post-WIMP techniques seek to take advantage of the human body's ability to express and convey meaning beyond a single point of input. For pointing tasks, such as selecting an element, a single pointer provides an equally embodied experiences, particularly for direct input, such as with a pen. However, embodiment of command selection and parameter adjustment, often requires more than one point of contact. In tangible interfaces, different graspable physical objects represent different command selections, much like picking up different tools for different tasks in the physical world. In the multi-touch and pen+touch modalities, which have no tangible objects, we can still embody command selection with gestural interaction. Wu and Balakrishnan [2003] used whole hand gestures—such as a chop gesture to create a virtual wall that could be used to push elements or stop their movements—that mimicked working with physical devices. Cao et al. [2008] and Harrison et al. [2014] designed gestures that mimicked grasping physical tools or objects, using the shape of touch contacts to recognize different gestures. However, some commands lack physical metaphors for mapping gestures, such as adjusting the translucence of an element. More abstract gestures are required. While physical metaphors may help users remember gestures, they may not be the most efficient, nor easy to transition between or perform repeatedly. Finger count and chorded [Lepinski et al. 2010] gestures provide quick methods for command selection that can be transitioned between by simply adding and removing touches.

Post-wimp techniques support multi-modal input. Even the pen, a single pointer, pro-

vides additional sensing parameters, such as pressure, tilt, hover state, and tracking position. Hover widgets used pen tracking above the surface to provide an additional commanding layer [Grossman et al. 2006]. Unlike a mouse, the pen supports direct input on an interactive surface, both pressing onto the surface and lifting the pen off. In the pen+touch modality, the pen can be tucked between fingers, allowing the right hand to touch the surface. Sensing when a user tucks the pen provides context for interaction and possibly phrasing [Hamilton et al. 2012; Hinckley et al. 2014], without the need for vision-based sensing [e.g., Annett et al. 2011] or wearable sensors [e.g., Webb et al. 2016c]. Further, bimanual interaction is common among post-WIMP techniques. In particular, many techniques followed Guiard's Right-to-Left Spatial Reference principle, in which the left hand either performed command selection [Matulic and Norrie 2013] or created a frame of reference for invoking commands with the right hand [Bier et al. 1993].

### II.5.4 Techniques for Phrasing

Quasimodes, transient interfaces, and contextual operations were used in varying degrees in the post-WIMP systems that we examined. For systems that supported bimanual input, left hand tension phrased interactions. Both Emmâ and Matulic and Norrie's document editing environment used edge-constrained gesture regions for phrasing quasimodes. This design approach keeps the left hand from occluding content and interfering with motions of the right hand. It also is similar to quasimode design for mouse/pen and keyboard input, in which the user holds down a key or set of keys with their left hand to activate quasimodes. The keyboard has physical affordances that allow users press keys without looking down at the keyboard, much like the edge of an interactive surface is recognizable via tactile feedback. For pen-based systems, kinesthetic tension from pressing a button on the stylus provided phrasing for gestural command selection. In BumpTop, visual tension of crossing through elements and widgets provided a key method for command selection.

Contextual operations were the most common method used, followed by transient interfaces, and then quasimodes. An advantage of contextual operations for gestural interaction is that we can reuse gestures for different commands, depending on the context. For example, in Manual Deskterity, a single touch with the left hand performs different commands depending on where the user touches. We suspect that contextual operations might help in gesture recall as the user does not need to remember every possible gesture, only those associated with a particular context. Since human memory is associative, we hypothesize that gesture for contextual operations become associated not only with the command, but also the context. Transient interfaces provide in-place command selection and parameter adjustment. They operate outside of a specific context, functioning globally. Examples of transient interfaces include an ink color palette, the Bezel Menu in Manual Deskterity, and a page formatting interface. Quasimodes are temporary modal states. Only Emmâ and Matulic and Norrie's document editing environment made extensive use of multiple quasimodes. In both systems, the default mode for pen input is inking. In other pen-based systems, gestures, such as pigtail, were used to delimit inking from selection and commanding. The advantage of quasimodes is that the same command may be performed multiple times (globally or on different elements) without having to select the command each time.

# CHAPTER III

# EMMÂ: PEN+TOUCH DIAGRAMMING ENVIRONMENT

We present Emmâ, a pen+touch diagramming environment that uses embodied dialogues through post-WIMP interfaces to support early-stage design processes. Designers typically sketch out ideas with a pen (or pencil) and create abstract representations in early-stage conceptual design processes. The objective is to explore a diverse set of possible solutions, rather than focus in on a single design. Existing digital design tools do not directly support conceptual design processes, instead focusing on creation of polished, concrete representations used in later stages of design. These tools primarily employ WIMP interfaces for use with mouse and keyboard input. They do not support the expressiveness of the human body, such as the simultaneous manipulation of multiple parameters by the hands that is made possible with multi-touch gestural input. *We hypothesize that Emmâ's pen+touch gestural interfaces will promote creative and embodied cognition, enabling designers to express and explore ideas in early-stage design.*

The wealth of information and media made accessible through the Web provides valuable exploratory resources as stimuli for early-stage design ideation. Designers gather materials related to a design problems or solutions and construct diagrams from these materials to explore and express ideas. Let us take, for example, a team of landscape architecture students working on designing a modern courtyard in a historic area of a city. During the conceptual design phase, they sketch out representations of their ideas. They might use Google images to search and find reference images of architectural style for that historic area to help provoke designs that would fit in with the surrounding buildings. Then, they might print out those images and sketch over them with tracing paper as they explore design possibilities. This process involves a number of different devices and media, requiring designers attention to shift from ideation to the technical steps required to transition rep-

resentations among the different media. As designers' collections of representations and materials grow larger, management of these resources and their ideas becomes unwieldy.

Emmâ seeks to address these issues of a growing collection of abstract representations developed during early-stage design by providing a holistic space for diagramming and free-form web curation—enabling designers to gather, assemble, sketch, write, explore, and express their ideas. Our objective is to support an embodied, holistic medium in which ideas could be gathered and assembled in a near-infinite space, rather than scattered across different media and files. By embodied, we mean based in the body, in an effort to engage embodied cognitive processes and harness designers' familiarity and practice of using their hands to work directly on media. Through iterative development of Emmâ, we investigate how to design bimanual interactions that support diagramming and free-form web curation processes. From our interaction designs, we identify methods for phrasing interactions that enable complex intermixing of operations for these different processes.

In this chapter, we first present a scenario depicting use of Emmâ by a hypothetical architecture student. Next, we describe the diagramming canvas, a near-infinite, zoomable space where users can engage in diagramming and free-form web curation. Then, we present interaction techniques in Emmâ for processes of gathering, assembling, and sketching. We explain how Emmâ was implemented using software frameworks and hardware sensing. We conclude with a discussion on how we used phrasing in design of bimanual interactions with quasimodes, transient interfaces, and contextual operations.

### III.1 Scenario: Architecture Design Studio

Rosalind is a fourth year architecture student in studio working on a fashion design assignment. She will design a light, transformable structure that attaches to the body. The objectives are to understand how the body and its movements define space, incorporating materials and tectonics. Rosalind must engage in creative visual thinking by


Figure 6: Emmâ diagramming example created by a hypothetical female architecture student for a fashion design assignment. The student searched for information on materials, such as textiles, gathering information, and developing an idea of warm vibrant silk contrasting with cold dark metal (right). Circle shapes inspired design of metal rings with silk strands. Images of human movement were gathered on the left. Ideas for how the fashion design would function were explored by sketching over these images.

gathering information about how fashion structures can connect to the body and exaggerate its movement, drawing over gathered images of body gestures to evolve her ideas, and diagramming a new design.

Rosalind launches Emmâ, and creates a new diagram, revealing an empty space that is near-infinite and zoomable. Unlike in other digital design tools, such as Adobe Illustrator, she does not need to specify dimensions for her diagram. She activates an integrated web browser, bringing up a Google search page. She realizes information needs on conceptual,

visual, and tectonic levels. She begins by searching for tectonics of making fashion using textile, structure, texture, contrast, transparency, cut, folding, and pleating as search terms.

Rosalind notices an interesting textile design with three orange slices in her search results. She selects the image and clips it from the browser. Rosalind gathers the image in her diagram by dragging and dropping the clipped image. Returning to her search results, she notices an image of a spinning frame used in cotton mills. The image contains three circles geometrically similar to the orange slices in the other image. Her visual perception not only recognizes similarity of shape, but also compares the cold, mechanical aspects of the spinning frame with the warm, organic aspects of the orange slices. She clips and drags the spinning frame from the integrated web browser into the diagram, and places it next to the oranges (see Figure 6, right). She begins reflecting about contrast between the warmth of the oranges and the coldness of metal spinning frame. Rosalind picks up the stylus with her right hand. She begins writing ideas next to the two images. She circles them for emphasis. She continues gathering material and annotating.

She opens another Google search page and queries fashion, body, form, style, and expression to represent her conceptual information needs. In the search results, Rosalind sees an image of brightly colored silk fabric. Reflecting on her diagram, a new idea emerges: combining metal and silk. She clips and drags the silk fabric image into her diagram. She gathers more images and text about silk and metal. She assembles, connecting cold dull metal with warm vibrant silk centered on the original two images. She activates Emmâ's edge blending quasimode with three touches from her left hand on area in the bottom-left corner of the display. Then, while in this quasimode, she blends together images of silk and metal by dragging the pen in her right hand from the corners of images inward to define varying lengths of translucent gradients. These gradients fade out the e

Circles remain central to Rosalind's ideas. Suddenly, through reflection-in-action [Schön 1983; Webb et al. 2013], she thinks of rings. The idea of using metal rings with

long colorful silk fabric emerges. With a solid fashion theme concept in hand, she proceeds.

Rosalind uses a selection quasimode to selects all elements in her diagram. Then, she drags them to the right to make room for the next part of her assignment. She switches to the integrated web browser, where she finds the Flickr tag page full of images representing movement.

For visuals, she queries form fitting, A-line, proportion, and color. Rosalind again gathers by dragging elements from the integrated web browser to her diagram. She assembles, with scaling and overlapping.

To diagram precisely, she picks up the stylus for inking. Using her left hand, she gestures and activates a transient interface for specifying the color and thickness of ink strokes. With her left hand held, phrasing the interaction, she uses the pen in her right hand to select a color and ink thickness in the transient interface. She sets the ink color to white, since most images she gathered have dark backgrounds. She lifts her left-hand gesture to end the phrase, removing the transient interface. She begins sketching her ideas over gathered images.

Through diagramming, Rosalind discovers visual design components. She diagrams shapes based on individual images and texts, and combinations. These shapes, although drawn two-dimensionally, can be read in both two and three dimensions in later design phases.

Rosalind iterates through gathering, assembling and sketching. She annotates themes with handwritten text. After formulating ideas, Rosalind sketches a set of rings with three silk ribbons running through them, between the images of movement and the materials composition. This connects the sides, emphasizing the warm vs. cold and light vs. heavy themes.

### III.2 Diagramming Canvas

The previous scenario depicts how Emmâ can support processes of diagramming and free-form web curation, showing examples of gathering visual media, assembling that media to show relationships between ideas, and sketching amidst the assembled media to explore design ideas. The primary component that supports these processes in Emmâ is the *diagramming canvas*, a near-infinite zoomable 2D space. Initially, the canvas is blank, inviting the user to add content through gathering techniques, such as an integrated web browser or making marks with the pen. Image or text media elements gathered from the Web automatically maintain referential links back to their source documents. This referentiality enables connecting more than just visual media, but also the ideas that are represented in the source documents. For example, an architect designing a new building in Barcelona may gather information from an article reviewing the style of the famous Catalan architect, Antoni Gaudí. Image and text elements clipped from this article maintain links back, allowing the architect to revisit these ideas later by simply navigate via one of the gathered elements.

The canvas is free-form, enabling multimedia elements to be spontaneously gathered from the web, written about, sketched amidst, manipulated, and visually assembled in a continuous space (see Figure 7). The user can adjust various visual qualities of the elements, such as position, size, orientation, and translucence. Relationships among gathered and assembled elements emerge through similarities and contrasts of these visual qualities.

As a near-infinite space, users can pan the view of the diagramming canvas using a single touch drag, starting from an empty area of the canvas. A user can pan her view to provide more empty space for gathering elements or to orient the canvas for sketching with the pen, perhaps over a specific image element.

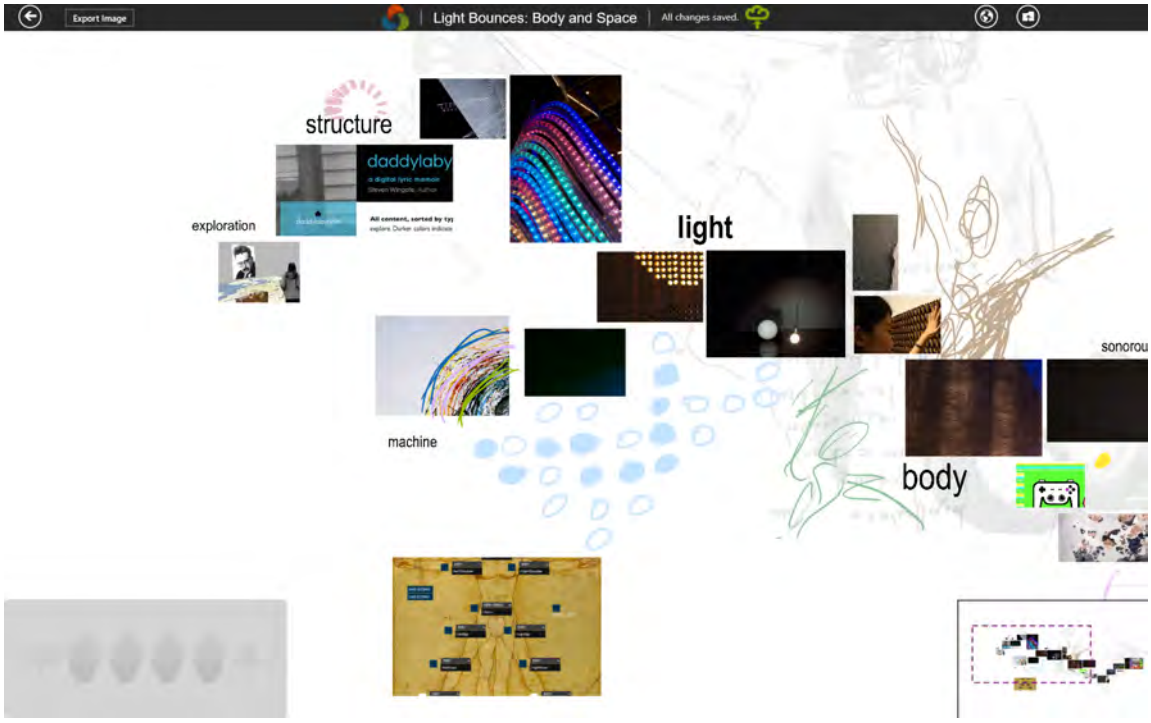


Figure 7: Emmâ diagramming space with mix of images, text, and sketching gathered and assembled. Hotpad in the bottom-left corner. Mini-map in the bottom-right. Dashed outline shows current view within larger diagramming space. Buttons for toggling integrated web browser and local image browser in the top-right.

### III.2.1 Zoomable User Interface: Working Across Scales and Levels of Detail

The zoomable canvas allows designers to work across multiple scales and levels of detail. Zooming supports both focused and contextual views through spatial scaling of visual content—albeit not simultaneously, unless combined with overview+details features (e.g., a mini-map, see below) [Cockburn et al. 2009]. Zooming in magnifies content. Elements increase in size. Elements in the periphery are removed from view, as the focus content—the center of zooming—fills the space. Zooming out shrinks content. Elements decrease in size. Peripheral elements, previously removed from view, are added back. From the initial view showing the entire diagram, the user—whether the original designer

that created the diagram or someone else viewing it, such as a client—can think about the ideas holistically, and then focus in on specific details by zooming in. They can easily switch between macro and micro levels of detail—with a single multi-touch gesture. We observed this practice by landscape architecture participants in our qualitative investigation of Emmâ in the design studio (see Chapter IV). We hypothesize that a zoomable canvas will help users cognitively manage the overwhelming amount of information that arises when the number of gathered elements and ideas grows large.

In Emmâ, continuous zooming is performed using the familiar two-touch stretchy gesture [Kurtenbach et al. 1997]. Both touches must be on an empty area of the canvas—not over an element—to distinguish touches for zooming versus touches for interacting with elements. Dragging the two touch outward, away from each other, stretches the view, magnifying content as the canvas zooms in. Dragging the touches inward, towards each other, compresses the view, shrinking content as the canvas zooms out. This gesture embodies the transformation of zooming through the stretching and compression defined by the relative distance between the two touch points.

A mini-map in the bottom-right corner shows a small view of the entire diagram with a dash-outlined rectangle indicating the user's current zoomed view (see Figure 7). The mini-map provides an overview to help users avoid getting lost in a near-infinite, zoomable space.

### III.3 Sketching Techniques

Sketching is primary to diagramming with Emmâ. Our landscape architecture participants used sketching techniques more than those for gathering or assembling (see Chapter IV). In Emmâ, sketches are created with the pen. As suggested by Hinckley et al. [2010], the pen, by itself, always makes ink. Using only the pen, users can sketch out ideas or annotate thoughts. For example, landscape architecture participants working fre-

quently sketched over gathered images to explore or express ideas about form and space. Additionally, the user can draw arrows or lines to express relationships between groups of elements, while annotating labels for the groups with handwritten notes.

Sketching consists of making a series of ink strokes. Users create ink strokes by pressing down the pen, then dragging the pen tip across the interactive surface, and finally lifting the pen to end the stroke. By pressing the pen down again and repeating, the user can draw additional ink strokes.

### III.3.1 Ink Palette

The user adjusts the color and thickness of ink applied when the pen is pressed using the Ink Palette. The Ink Palette is a transient interface, appearing in-place when requested by the user. The Ink Palette is activated using a three-finger right-angle gesture with the left hand (see Figure 8), performed anywhere in the diagramming canvas. The near right-angle formed where the index finger and thumb meet defines a bottom corner of the Ink Palette. Moving the right-angle gesture re-positions the Ink Palette. Using the pen in the right hand, the user can adjust ink stroke thickness via a slider and use a color picker to select ink stroke color.

Unlike with quasimodes, where all input performs a specific operation, the pen can still be used to make ink strokes on the diagramming canvas while the Ink Palette is activated. As a result, the user can quickly and repeatedly adjust the color and thickness of ink strokes without having to reactivate the Ink Palette. This interaction is similar to a painter working with a paint palette, where the left hand holds and orients the palette, while the right hand, with a brush or knife, grabs or mixes paint.

### III.3.2 Wet Ink

All ink strokes are initially considered wet. *Wet ink* strokes are not part of the diagramming canvas. The user must first decide how to handle the wet ink strokes. Wet ink can be

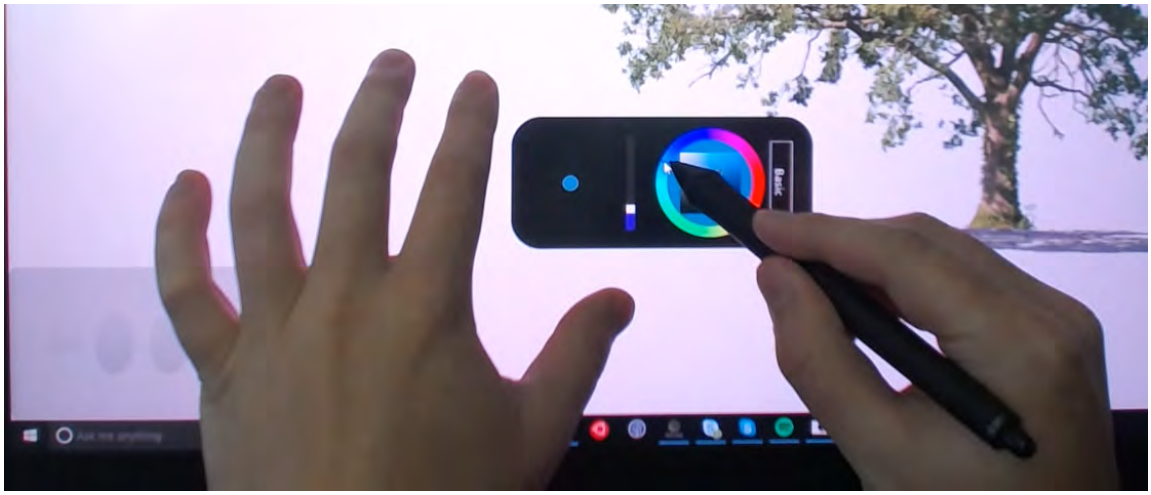

Figure 8: Example of Ink Palette. The left hand performs a right-angle gesture to activate Ink Palette. The right hand, with the pen, adjusts the hue of the ink to a shade of blue.

dried, combining all wet ink strokes into a single sketch element and adding that element to the diagramming canvas. A sketch element supports the same visual transformations as image and text elements. Wet ink can be discarded, removing all wet ink strokes from view. Finally, wet ink that is handwriting can be converted into text. The Ink Menu allows users to specify which of these three options to apply to the wet ink.

### III.3.3 Ink Menu

Following a 500 ms timeout after the pen is lifted to create a wet ink stroke, the Ink Menu appears at the top-right edge of the bounding box containing all current wet ink strokes (see Figure 9). The Ink Menu consists of three buttons and a selection outline showing a bounds of all the wet ink strokes. By pressing the check mark button, the wet ink is dried, adding the current set of wet ink strokes to the diagram as a single sketch element. Instead of drying ink, the user can discard all the wet ink strokes by pressing the middle button with a trash icon. Or, if the wet ink strokes represent handwriting, the user can press the handwriting recognition button (bottom most) to convert the we ink to text.

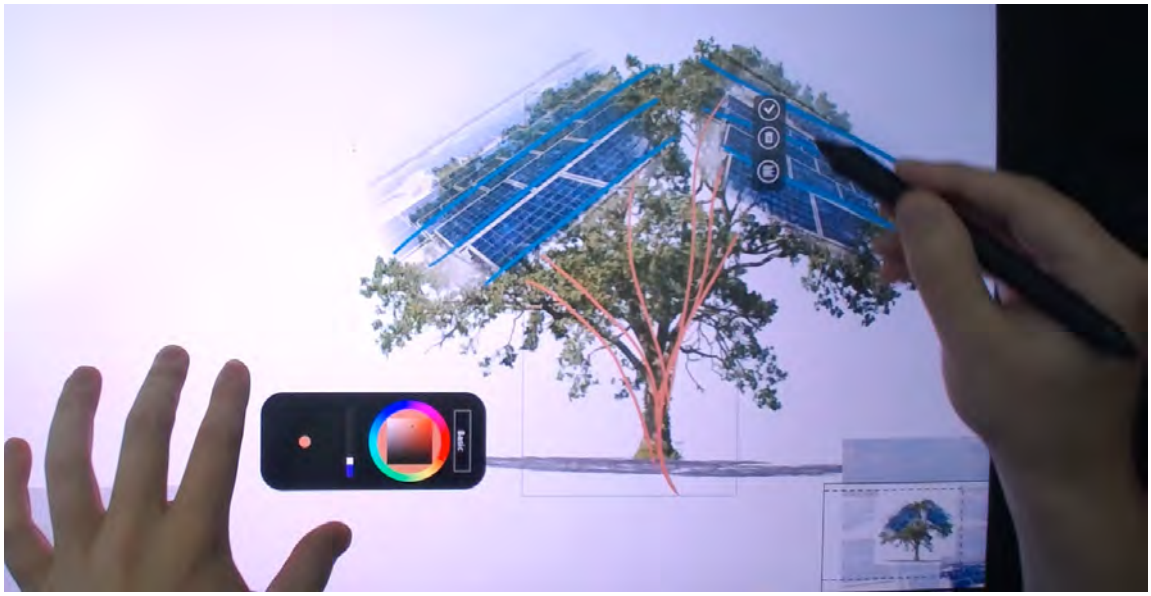

Figure 9: Example of both Ink Menu and Ink Palette activated. In the previous figure, the user changed the ink color to blue. He then draws blue lines on the solar panels to indicate the need for structural pieces to support rows of solar panels. He leaves the Ink Palette activated during this process. After drawing the blue lines, he use the Ink Palette to change the ink color to orange. Then, he traces orange lines over the branches and trunk of the tree to represent conductive tree-like structures for holding up the solar panels and drawing energy from them. The Ink Menu appears after he finishes sketching these lines. The pen in the right hand moves to press the check mark button to add the orange ink strokes to the diagram.

Pressing any of the Ink Menu buttons removes the Ink Menu and performs the associated operation.

When the Ink Menu is activated, the user can still add more wet ink strokes with the pen. When the pen makes contact with the interactive surface, the Ink Menu is removed to avoid interfering with the user's sketching. The Ink Menu appears again, following the timeout after pen lift.

### III.3.4 Straight Line Tool

The user can create a straight line in Emmâ using the Straight Line Tool, a contextual operation activated with a pen+touch interaction. The Straight Line Tool is activated by making a left-hand touch anywhere in the diagramming space, followed by a pen press in close proximity to the left-hand touch (within 50 pixels on a 96 dpi display). The contextual operation is defined by proximity, rather than the bounds of an element, like in other contextual operations we presented. Any pen presses too far away will be considered regular ink strokes. The pen and left-hand touch each define an endpoint of the line. Dragging either will adjust the positions of those endpoints. Lifting the pen, adds the straight line in its current position to the diagram. Another straight line can be created by pressing down the pen in close proximity to the touch. Lifting the left-hand touch ends the operation and the phrase.

## III.4 Gathering Techniques

Free-form web curation in Emmâ involves gathering visual media, such as images or text, from the Web or the local file system. Rosalind, in the scenario above, began by searching for and gathering interesting imagery as stimuli for design. In Emmâ, we provide three methods for gathering visual media: the Integrated Web Browser, the Local Image Browser, and the duplicate operation.

### III.4.1 Integrated Web Browser

The Integrated Web Browser is a functional, but simple web browser—with address bar and forward and back buttons— built into Emmâ that enables users to gather content from the Web (see Figure 10). The process of gathering web content typically begins with the user typing a few search terms into the address bar, which initiates a Google search, similar to functionality in Google Chrome. The user then browses the search results, looking for

content to add to her diagram.

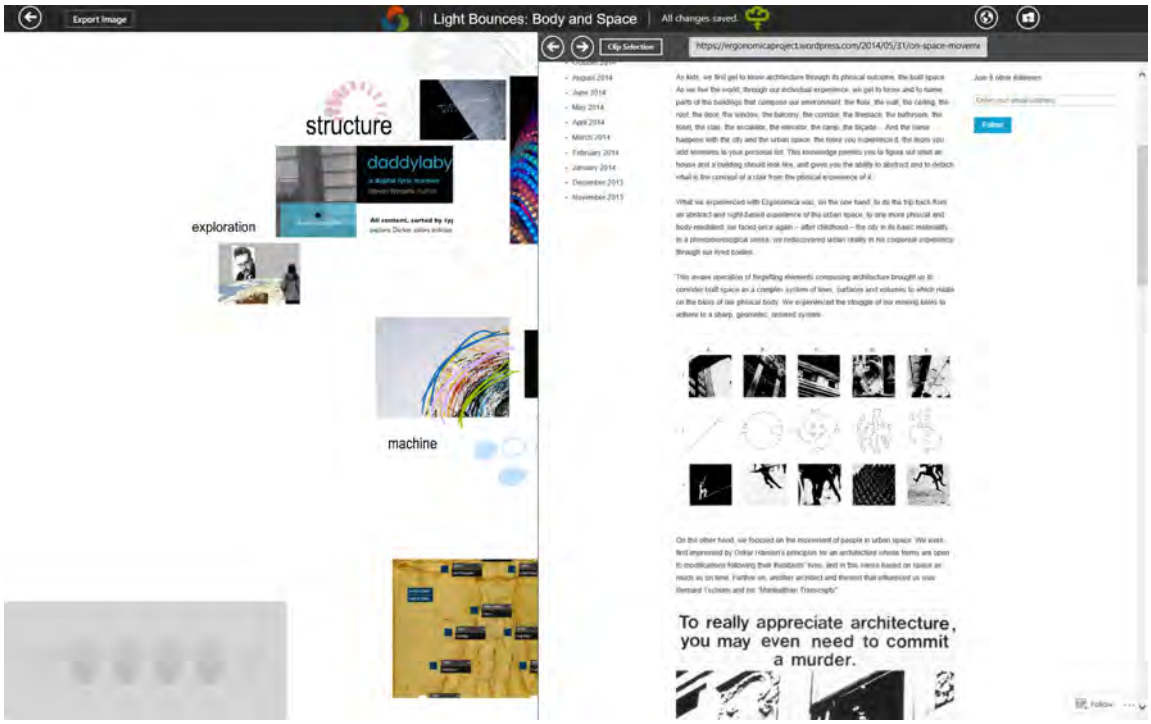


Figure 10: Snapshot of Integrated Web Browser showing an article on the relationships of space and human movement in architecture design.

The user gathers desired content from the browser, by first selecting the content with pen or touch input, and then pressing the clip button at the top, near the address bar. Clippings of selected content are presented in a contextual pop-up next to the browser (see Figure 11). The user can then drag and drop clippings from the pop-up into her diagram.

The interaction for clipping content is not the desired design, but was forced upon us due to limitations in the Windows Runtime framework. Security restrictions for the WebView control in the Windows Runtime framework prevented adding JavaScript callbacks for input events that occur on a web page. While we could not get input event callbacks, we could execute JavaScript calls on the WebView. Thus, we created a button (Clip Se-

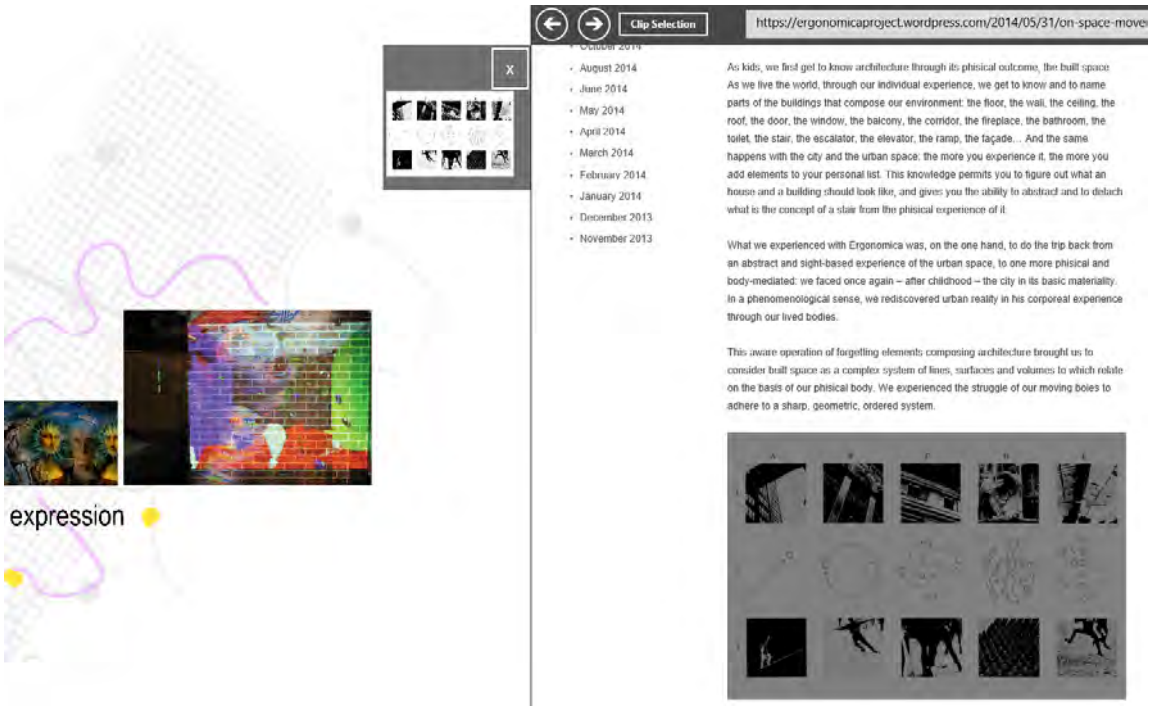


Figure 11: Snapshot of contextual pop-up (next to browser forward and back buttons) after selecting and clipping an element. The selected image (highlighted in gray) in the Integrated Web Browser was clipped by first selecting the image, then pressing the "Clip Selection" button. Then, the pop-up with the clipped image appeared.

lection) located outside the WebView control, which makes a JavaScript call to get the HTML contents of the current selection. We then generate an image or text clipping from that HTML, and present it in the contextual pop-up, where the user can use pen or touch to drag and drop the clipping into the diagramming canvas.

We developed several alternative designs for the clipping operation that could not be supported in the framework without building our own web browser from scratch. One design involved direct drag and drop, similar to how drag and drop of content works in a mouse-based web browser. Another design used bimanual interaction, where one hand (probably the right hand for precision) defines a selection of content on which to clip. While holding that selection, providing tension for the phrase, a touch down with the other

hand adds the clipped content into the diagram. Dragging that touch allows the clipped content to be translated to the desired position. Lifting the touch, adds the dragged element to the diagram at its current position.

The Integrated Web Browser is revealed and hidden using a button with an Earth globe icon at the top of the screen, on the user's right hand side. Pressing the button, toggles visibility of the web browser, which when made visible appears on the user's right hand side. The browser has a fixed width of 1100 pixels, and it always appears above the diagramming canvas, which means it can occlude content. However, the user can still pan and zoom the diagramming canvas in the region not occluded by the browser.

### III.4.2 Local Images

Not all information that designers work with while diagramming comes from the Web. Some content comes from other sources, such as design documents provided by a client or from photos taken at a site.

In Emmâ, the user can add images from the local file system, including USB thumb drives, which were commonly used by landscape architecture students in our investigation. Local images are added through the Local Image Browser, which functions similar to the Integrated Web Browser. The user presses a button, with a folder icon, near the top-right of the display to toggle visibility of the Local Image Browser. The Local Image Browser provides a navigable view of the local file system, which the user can use to find the desired images. The user can then drag and drop images from the Local Image Browser into the diagramming canvas.

Visual designers often work with high resolution images, such as photos taken with a large megapixel camera. When an image is added to the diagramming canvas, we check how large in pixels that image will appear given its size and the current zoom level. If the image's rendered size exceeds 1000 x 750 pixels, we apply scaling transforms to reduce the

size of the image to no larger than that size. This prevents large images from completing filling the current view on the diagramming canvas, occluding content and access to the diagramming canvas.

### III.4.3 Duplicate: Contextual Operation

Through discussions with users during our iterative design process, they reported wanting a way to make copies of existing elements. Designers often create duplicates of visual material to explore different design solutions with the same content. We developed a contextual technique for duplicating an element. Two touches on an element with the left hand phrase the technique. Then, the user places a single right-hand touch on the element. A translucent copy of the element appears, slightly offset above and to the right, while still overlapping the element (see Figure 12). Then, the user drags the translucent copy away from the original with the right-hand touch. While dragging with the right hand, the copy can be translated to a specific position. Lifting the right hand touch, drops the copy at its current position, adding it to the diagramming canvas. While keeping the two left-hand touches down, additional copies can be with repeated drag and drops with the right hand.

## III.5 Assembling Techniques

In Emmâ, the user assembles gathered content to express ideas and create meaning by applying visual transformations to elements. Translate, scale, and rotate are the primary transformation operations for assembling elements. These are contextual operations performed directly on an element or group of selected elements. Beyond these operations, we use quasimodes for: (1) selecting multiple elements to assemble with translate, scale, and rotate; (2) adjusting the translucence and blending of elements; and, (3) aligning elements in grids.

Figure 12: Example of translucent feedback when performing duplicate operation. Two left-hand touches (pink circles) and one right-hand touch (green circle) show touch points. Colored circles not present in Emmâ.

### III.5.1 Translate, Scale, Rotate

The most basic organizational operation is translation, in which an element's position is changed. Using spatial assemblage via translation, designers can connect and contrast ideas through juxtaposition. Placing elements near each other, particularly with some overlap creates visual grouping, connecting the elements and the ideas they represent.

In Emmâ, an element's position is adjusted with a contextual operation involving a singe touch drag on that element. There is a direct mapping in touch drag movement and element translation. Lifting up the drag ends translation. Multiple elements can be translated simultaneously using multiple touches.

In addition to translation, scale and rotation are basic organizational operations used in diagramming. Scale enables expressing relationships through contrast in size. Variations

in size support legibility of ideas, using larger elements to draw the viewer's attention to the most important points. In a zoomable space, scale allows for representing ideas at different levels of detail. For example, the more general ideas can be scaled large, so they are visible from a zoomed out view, showing the entire space. Then a user can zoom in to see more focused details, represented by smaller elements.

Through rotation, the orientation of an element is changed with respect to the x and y axes. Rotation can be used to promote legibility of relationships, grouping together a set of elements with a similar rotation. Rotating an element enables designers to explore different perspectives as the orientation changes.

The user simultaneously translates, scales, and rotates using the common two-finger stretchy gesture [Kurtenbach et al. 1997]. Translation is defined by the relative change in location of the midpoint between the two touches. The change in distance between the two touches defines the scale transformation. Rotation is defined by the change in angle between the initial line defined when the two touches first came into contact with the surface and the current line defined by the most recent position of the two touches. Users can perform this gesture with either two fingers on one hand, or one finger on each hand. We hypothesize that the user's choice in using one or two hands to perform stretchies is affected by the context. For example, if the user needs to make more fine-grained transformations, she will use two hands. With two hands, it is easier to hold one of the touch points in place, while moving the other one. Alternatively, the user may choose to use the most convenient hand. Such as when a user has just gathered an element with a single-touch drag and drop, the drag and drop hand is already near the element. The user would likely use that hand, by itself, for the stretchy.

### III.5.2 Quasimodes with the Hotpad

Emmâ has a number of visual design techniques to help assemble elements that involve repeatedly performing the same operation on different elements with different parameters. Visual design tools, such as Adobe Photoshop, use modal tools—activated through a tool palette—for performing these repeated operations. A modal tool remains in effect until a different mode is activated. This approach is prone to errors, as the user may forget which modal state she has activated [Sellen et al. 1992]. Instead, we use quasimodes, which activate temporarily, while the operation is performed, and then return back to the default modal state after the phrasing is ended. Quasimodes still require an initial activation before performing an operation.

In Emmâ, we provide an edge-constrained gesture area, called the *Hotpad*, for activating quasimodes with chorded touch gestures using the left (non-preferred) hand (see Figure 13). Edge-constrained gestures with the left hand support efficient modal switching [Hamilton et al. 2012; Matulic and Norrie 2013]. The Hotpad is a rectangular region located in the bottom corner of the display on the user's left hand side. Placing a single touch on the Hotpad activates the Selection quasimode. Placing two touches on the Hotpad activates the Translucence quasimode; and, placing three touches on the Hotpad activates the Edge Blending quasimode. Quasimodes can be quickly switched between by adding or lifting touches on the Hotpad. Lifting all touches returns to the default mode—inking with the pen and translate, scale, rotate with touch.

#### III.5.2.1 Selection

Once a designer is working with multiple elements, the need to select a subset of the elements for interaction is required. Emmâ supports selection of multiple elements via either lasso or crossing [Apitz and Guimbretière 2004] selection. Lasso selection allows for quickly selecting a group of elements bounded by a circular shape (see Figure 13). Cross-

ing selection allows selecting elements by drawing a selection line through the bounds of desired elements. This enables more precise selection than lasso, enabling selection of a subset of overlapping elements, rather than the entire group, as would be the case with lasso.

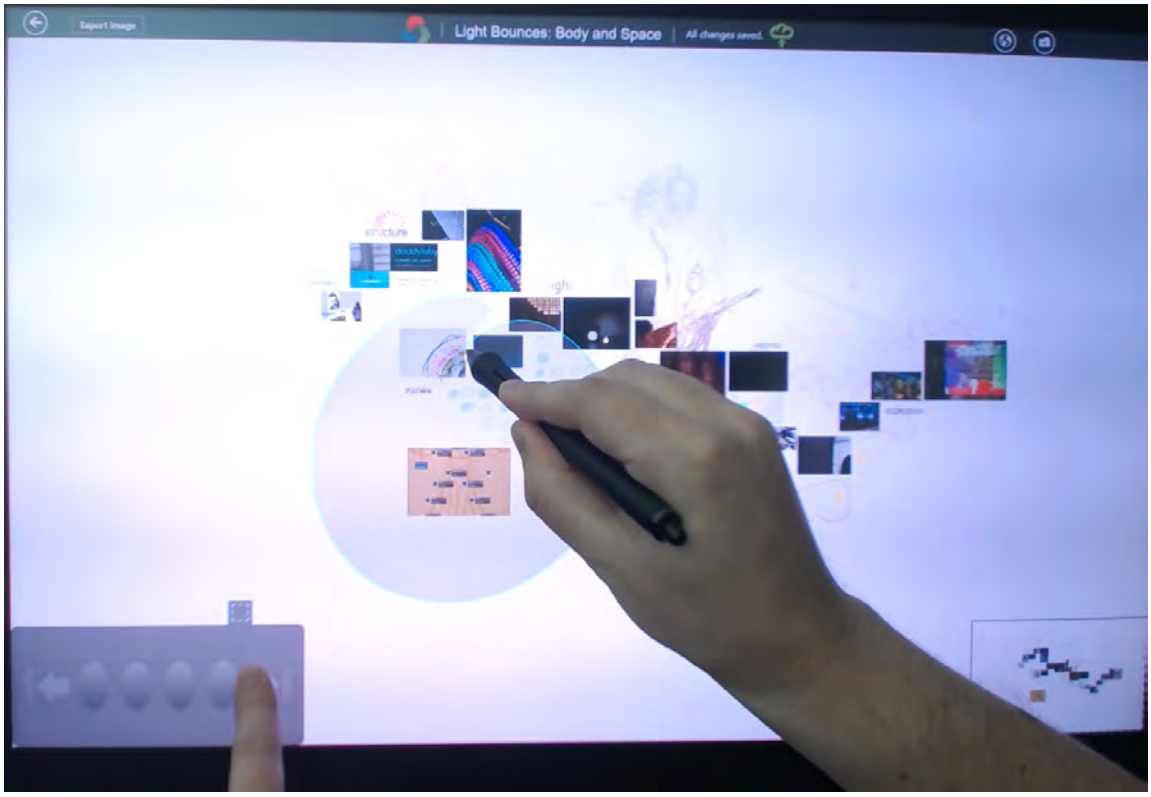


Figure 13: Example of lasso selection in Emmâ. Left hand touch on the Hotpad defines the quasimode, while the right hand with pen draws a lasso selection.

Selection in Emmâ is performed with a bimanual quasimode technique. A single left-hand touch held on the Hotpad, activates and maintains the selection quasimode. Then, using the right hand with either the pen or touch, the user can draw a selection region by either lassoing around the desired elements or crossing through the elements, one at a time. Lifting the pen or touch doing the selection, selects the desired elements, but the selection

quasimode remains active while the Hotpad touch is held. If the desired elements were not selected, the user can immediately repeat selection without having to activate selection again. Once the user has selected the desired elements, she can lift the Hotpad touch to deactivate the selection quasimode and end the phrase.

A group of selected elements can be moved with a single touch drag on any of the selected elements. The user can translate, scale, and rotate a group of selected elements with a two-finger stretchy gesture.

#### III.5.2.2 Translucence

Visual designers that we talked with expressed the importance of translucence as a common visual design tool. Designers use translucence for a number of functional reasons. One reason is to blend together overlapping content. Increasing the translucence of an element allows the elements behind it to become visible. Another reason is to reduce emphasis of an element, making it appear more distant. For example, landscape architects use increasing levels of translucence for elements that reside further in the background. This mimics what happens in human vision, where objects further in the distance become more fuzzy and less crisp.

In Emmâ, users can adjust the translucence of an element by activating a quasimode via the Hotpad. Two left-hand touches down on the Hotpad activates the translucence quasimode (see Figure 14). With the left-hand touches held down, translucence is applied to each element using the pen in the other hand. Dragging upward with the pen on an element increases the translucence for that element. In Laban Movement Analysis, an upward movement is associated with lightness. In a similar manner, we use upward movement to reduce the weight of the element through increasing its translucence. Dragging downward has the opposite effect, decreasing translucence and giving the element more weight as it becomes more opaque. When multiple elements are selected, the effect applies to all

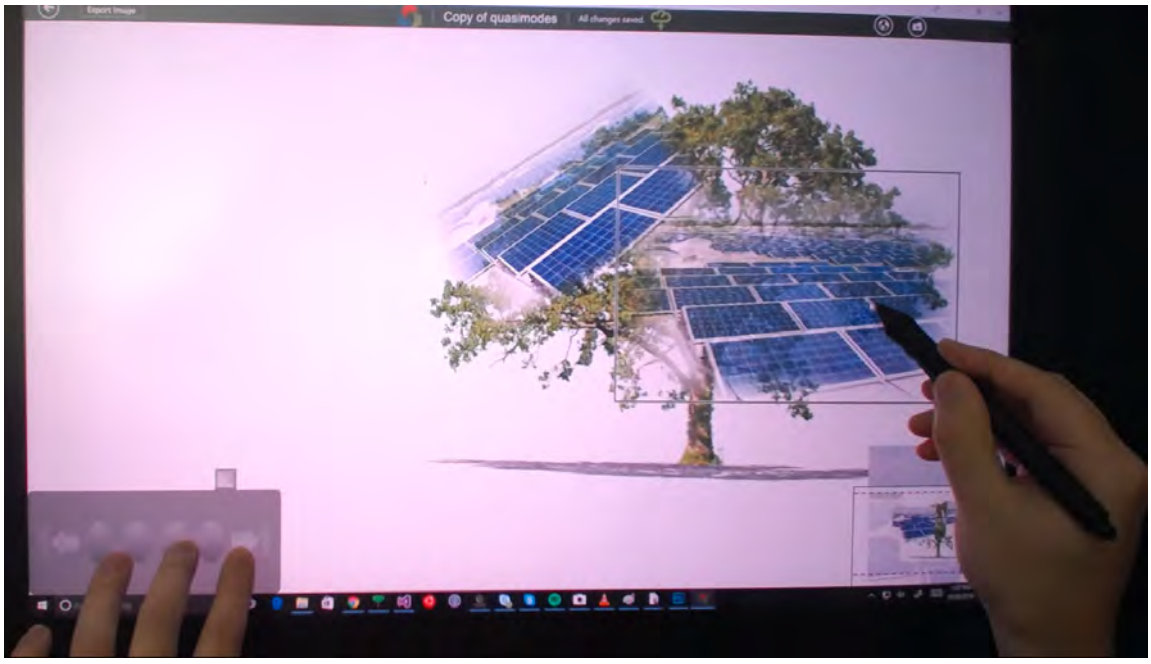


Figure 14: Example of translucence in Emmâ. Two left-hand touches on the Hotpad defines the quasimode, while the right hand with pen adjust translucence level.

elements.

#### III.5.2.3 Edge Blending

Designers use overlap to juxtapose and connect elements. The hard edges of visual imagery often provide strong contrast among overlapping elements, helping to separate them rather than connect them. In Emmâ, users can adjust the softness for the edges of image elements to help blend together overlapping content. We hypothesize that this will promote the formation of conceptual combinations—a cognitive process whereby two or more concepts are mentally synthesized to for a higher order concept, which may be a new, emergent idea [Finke et al. 1992].

Similar to selection and translucence, the user activates the edge blending quasimode via three touches on the Hotpad with the left hand. Using the pen or touch input with the right hand, the user drags from the edge of an image inward applying a soft gradient

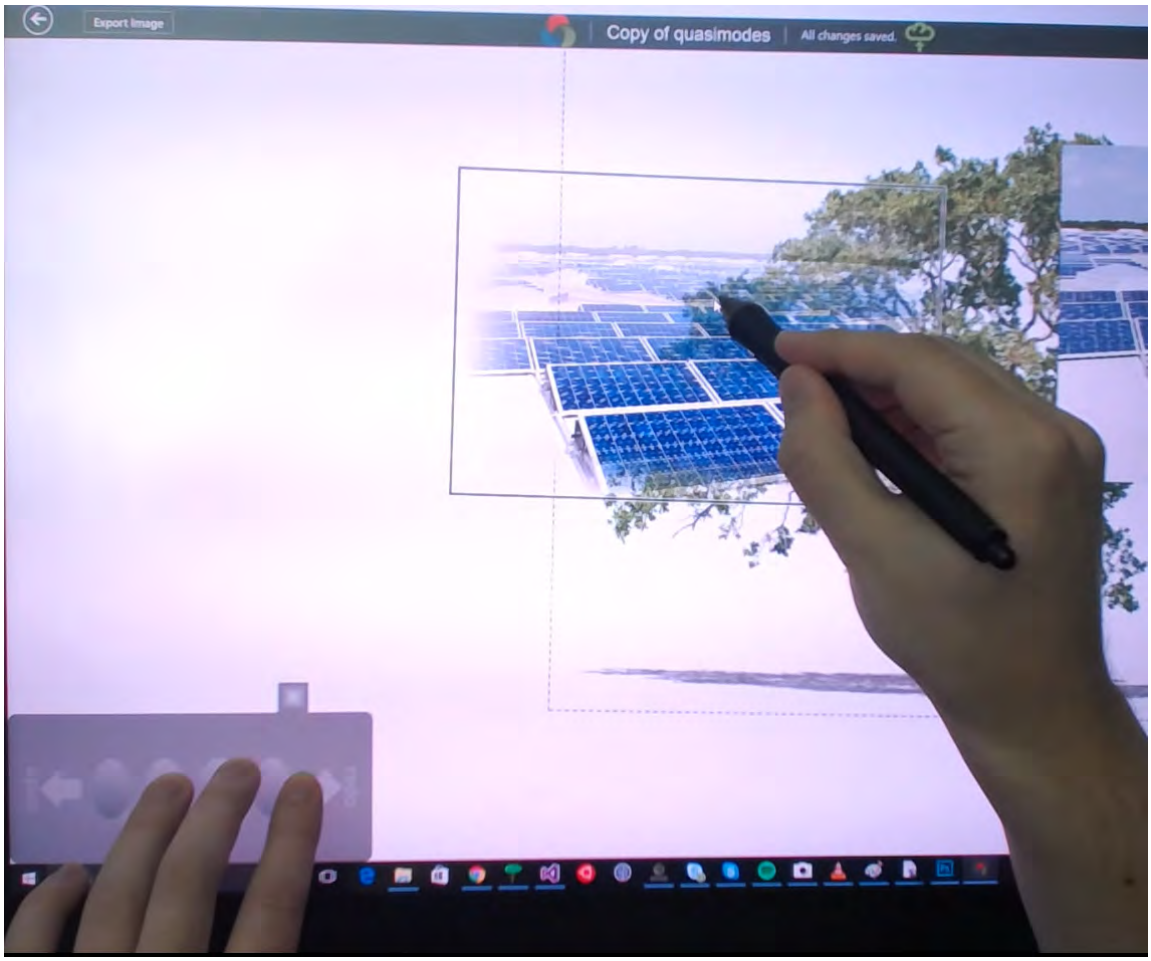


Figure 15: Example of edge blending in Emmâ. Three left-hand touches on the Hotpad defines the quasimode, while the right hand with pen adjust edge blend gradient lengths.

opacity mask that begins at the current drag position and ends with complete translucence at the edge (see Figure 15). The further inward dragged the wider the gradient. Dragging from a corner, will add gradients for the two edges that intersect in that corner. Tension from holding left-hand touches maintains the quasimode. This enables applying edge blending to multiple elements in series without having to activate the command repeatedly. We suspect that this is particularly valuable for edge blending where two or more elements require varying amounts of gradient on different edges.

#### III.5.2.4 Undo and Redo

In addition to activating quasimodes, single-touch swipes on the Hotpad perform undo or redo depending on which direction was swiped. Swiping to the left undoes the previous operation. Swiping to the right redoes the last undone operation.

### III.5.3 Alignment: Element-Defined Grids

Aligning content precisely is an important component of visual design. Alignment provides visual structure connecting and expressing relationships among aligned elements. Emmâ supports aligning elements in grids. A grid is a common visual design structure that divides a two-dimensional space into smaller regions, arranged in rows and columns [Müller-Brockmann 1996]. The smaller regions, called cells, are bounded by intersections of rows and columns and act as units of structure within a grid. Padding and visual boundaries for cells support legibility, helping the viewer distinguish groupings of content. Placing elements in adjacent cells connects those elements and the ideas they represent.

We developed Element-Defined Grids, a bimanual multi-touch technique for aligning elements in a grid structure without requiring users to define rows and columns. In Element-Defined Grids, the user begins by first positioning an element and activating a grid based upon that element's size and position. A grid is activated with a bimanual gesture. First, the left hand holds down a touch on the Grid Control, an edge-constrained region for adjusting parameters of Element-Defined Grids, located above the Hotpad. Holding this touch activates the Element-Defined Grids quasimode. Next, with the right hand, the user taps an element to activate its surrounding grid structure. A set of faint grid lines appear around the element (see Figure 16). The width and height of the element defines the width and height of each cell. The grid lines gradually fade out further away from the element. The user can continue to tap different elements to activate their grids. The user can place two touches with her left hand on the Grid Control and perform a two-finger

stretchy gesture to adjust the radius at which grid lines fade out from elements.

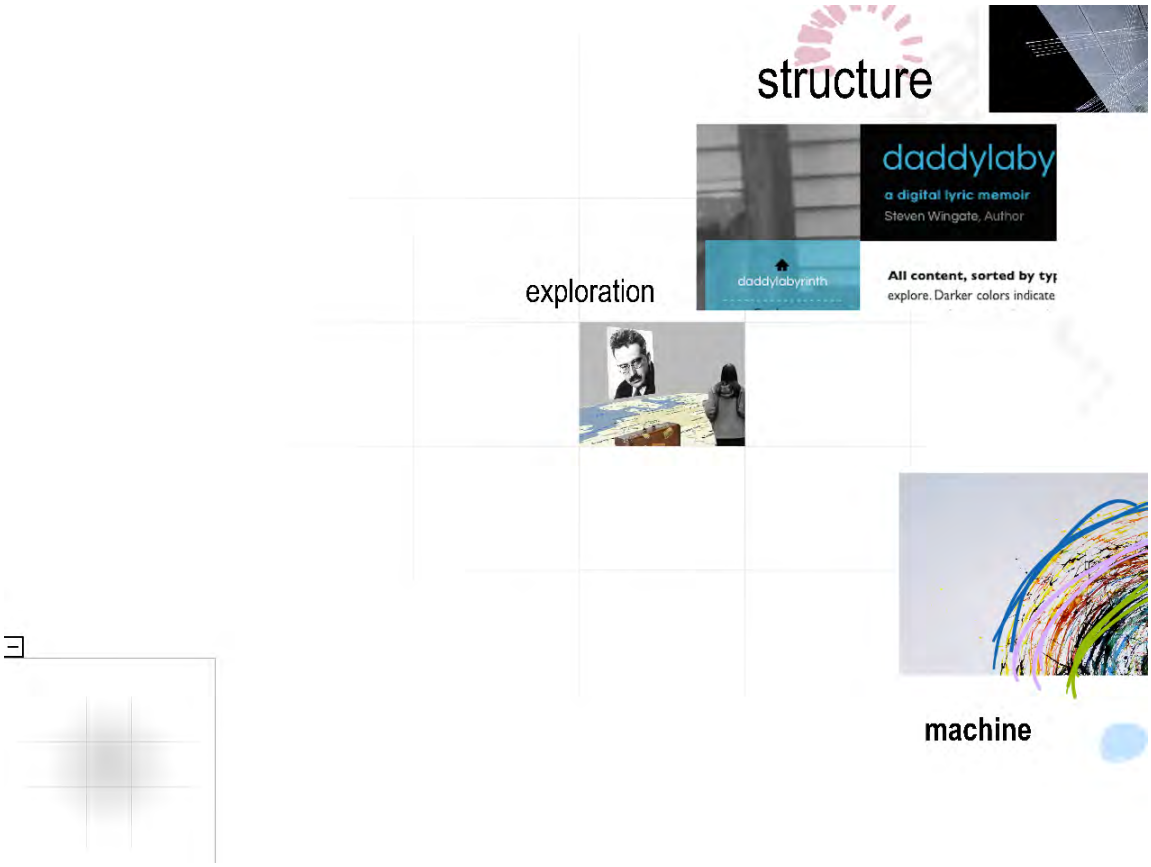


Figure 16: Example of Element-Defined Grids, where the user activated the quasimode by holding a left-hand touch on the Grid Control in the bottom-left corner, and tapping the image element in the center. A faint grid structure is rendered where the width and height of grid cells matches the element's dimensions.

Element-Defined Grids uses snapping and scaling to align elements in the defined grid structure. Coarse touch input makes precise alignment difficult [Holz and Baudisch 2011]. When the user drags an element near a grid, that element snaps and resizes to fit within the nearest grid cell. This snapping and resizing may cause the element to jump out from under the user's touch, which could cause confusion. When this happens, we keep the

selection outline that is always present when dragging an element at the current unsnapped position (see Figure 17). The element, itself, gets snapped and resized to fit in a grid cell. As the user continues to drag the element, the selection outline moves continuously with the drag, maintaining visual feedback about the position of the element. When the user lifts the touch, the selection outline disappears, and the element is inserted in its current grid cell. An element can be removed from a grid cell by dragging it out. Again, the selection outline will follow the drag, while the element remains snapped in the grid. If the user drags far enough away from the grid, the element will be removed from the grid and appear under the user's dragging touch.

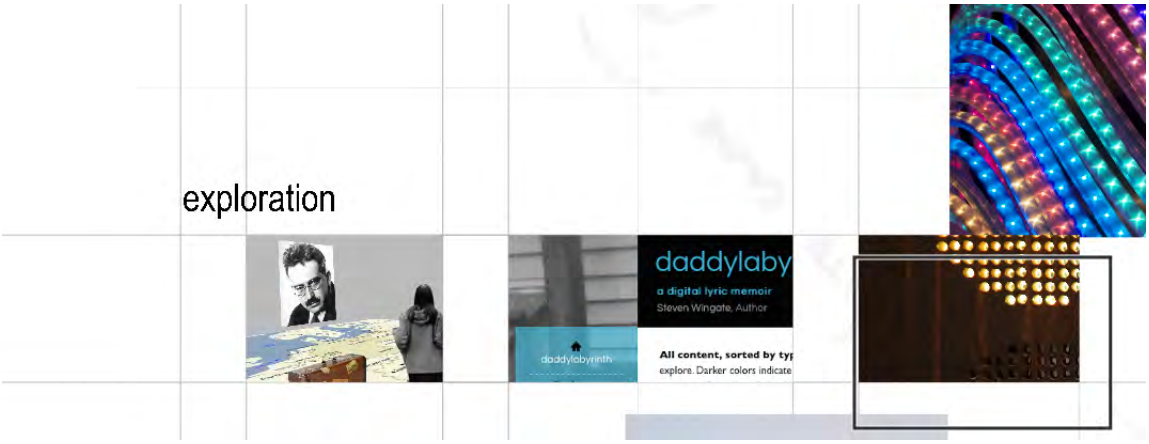


Figure 17: Example of offset created when a dragged element is snapped to a grid structure. The black selection rectangle remains where the element was before the snap, and continues to move as the touch is dragged. While, the visual contents of the element snap into the grid structure.

### III.5.4 Perspective Transform: Contextual Operation

Architects work with perspective, creating designs of 3D spaces on 2D mediums. Perspectives provide views of a site, as if standing at a specific point in space and time. They

help illustrate design solutions. Working with 3D models, designers can quickly create views for multiple perspectives. However, this requires tedious processes for constructing the models. In early-stage design, more rapid approaches, such as vanishing point sketching, enable designers to create fixed perspective views without 3D models. Further, landscape architects compose vignettes that mix perspective views of 3D models with 2D imagery of people, objects, and wildlife to create a scenes that show what a site could look with a specific design (see Figure 18).

Figure 18: Landscape architecture vignette of a residential courtyard created in Photoshop by Hejing Feng and Jiahe Bian.

In Emmâ, designers apply perspective transforms to elements using a contextual technique involving both pen and touch input. A touch on an element with the left hand defines a pivot point for the perspective transform. With the touch held, pressing the pen down

with the right hand, applies a 3D rotational force to the element around the pivot point (see Figure 19). The harder the pen is pressed; the greater the amount of rotational force is applied. If too much rotation is applied, pressing on the opposite side of the pivot point, begins to undo the previous rotation. The pen can also be dragged, moving the point where rotation is applied, enabling exploration.

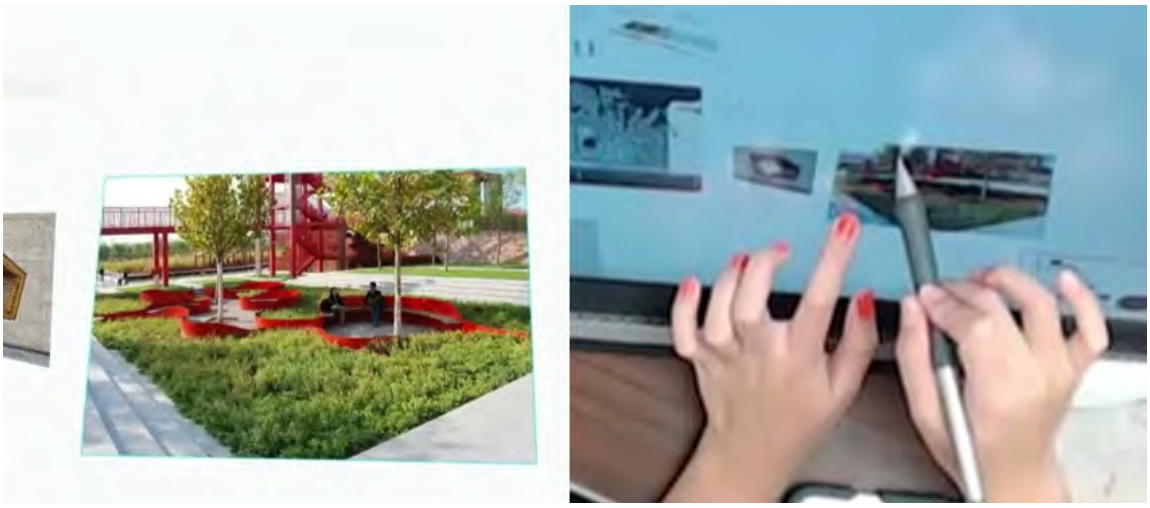

Figure 19: Example of perspective transform in Emmâ. Left side shows element being transformed. Right side shows users hands. Left hand defines the pivot point. Right hand with pen presses to apply perspective rotation.

We designed the gestural interaction to embody exploration of perspective transforms; not to create the most precise and efficient technique. Our goal is to support exploring a diverse set of ideas. The bimanual gesture allows the user to simultaneously manipulate several different parameters of perspective transform. The left hand touch defines the perspective rotational center. The pen position defines the axes of rotation, and pen pressure defines the amount of rotation.

### III.5.5 LayerFish: Combining Quasimodes, Transient Interfaces, and Contextual Operations

Free-form mediums allow for content to overlap. As the number of elements grows large, dealing with the visual stacking order—managing which elements need to appear above others—becomes increasingly difficult. We developed LayerFish, a bimanual technique for layering in-place with a fisheye visualization. LayerFish addresses issues of dealing with large numbers of overlapping elements through selection and the fisheye visualization. LayerFish employs quasimodes for selecting elements, a transient interface for layering, and contextual operations for translating, scaling, and rotating occluded content and hiding layers. The details of LayerFish and an evaluation of how effective it supports layering tasks with many elements are presented in Chapter V.

## III.6 Phrasing Bimanual Interactions with Quasimodes, Transient Interfaces, and Contextual Operations

We choreographed phrases of bimanual interaction in Emmâ using quasimodes (Space), transient interfaces (Shape), and contextual operations (Shape+Space). Using Guiard's principles of Left-Hand Precedence and Right-to-Left Spatial Reference, we design bimanual interactions that begin with the left (non-preferred hand) and define actions of the right (preferred hand) in context to the left. An interaction phrase is maintained through kinesthetic tension of the left hand.

We use the edge-constrained Hotpad as an area (Space) for activating quasimodes. By placing the Hotpad in a bottom corner, the user can quickly access it, while keeping touches with left hand out of the way of the right hand as it manipulates parameters for operations. The choreography involves movements of the left hand toward and away from the Hotpad as the user transitions between quasimodes and other forms of interaction. As we will see in Chapter IV, working in proximity to the Hotpad and hovering the left

hand near the surface can support effective modal switching among quasimodes, transient interfaces, and contextual operations. We use the number of touches on the Hotpad to determine which quasimode is activated. The user can quickly transition among quasimodes by adding or removing touches, without lifting up the whole hand. For example, a user could put one finger down on the Hotpad, and select a subset of elements. Then, put another finger down on the Hotpad, and adjust the translucence of the selected elements with a single pen drag using the right hand. The user can then lift one touch on the Hotpad, and select a new subset of elements. Then, put a second finger down again, and adjust the translucence of the new subset. This entire sequence becomes phrased together as a translucence task, with sub-phrases for the different selection and translucence operations.

Contextual operations phrase interactions around a spatial region (Space), the context, which could be the bounds of an element (e.g. Perspective Transform) or proximity to a left-hand touch (e.g. Straight Line Tool). They begin with left-hand touches, following Guiard's principle of Right-to-Left Spatial Reference, to perform gestures (Shape) specifying which contextual operation to enact. The choreography involves Impulsive gestures directed at specific regions (e.g., element bounds), in which the left hand places touches on an element to specify the operation to perform. Unlike quasimodes, in which the choreography directs the hands to one region relative to the display—the Hotpad, choreography for contextual operations direct movements towards spatial regions located all over the display. These left-hand touches can partially occlude the context of interaction or interfere with right-hand actions. They also can do more than just activate an operation, as with perspective transforms, where the left-hand touch also defines a parameter of the operation—the point around which to rotate for perspective transforms. While quasimodes define a global frame of reference for right-hand actions, contextual operations define an in-place frame of reference, centered around left-hand touches. We use quasimodes for interactions that must be applied to different elements repeatedly. We use contextual op-

erations for interactions that must be applied repeatedly to the same element.

Transient interfaces support phrasing bimanual interactions that occur in-place, around the user's point of focus, but unlike contextual operations, they are enacted in a global context, rather than on a specific element. Transient interfaces are activated with a left-hand gesture (Shape). The choreography for transient interfaces involves Impulsive gestures for initial activation, but unlike quasimodes or contextual operations, this gesture may be performed anywhere on the display. Thus, the left hand movements lack the quality of directed Space, which occurs with both Hotpad-activated quasimodes and contextual operations. The Ink Palette, as a transient interface, appears when the user performs the right-angle gesture with her left hand anywhere in the diagramming canvas. Like a contextual operation, the frame of reference for right hand interaction is confined to a specific region, the transient interface. Unlike a quasimode, other interactions with the right hand can be performed while the transient interface is active. However, we can restrict possible right-hand operations to ones associated with the transient interface or uni-manual input. By requiring that the user maintain a left hand gesture for a transient interface, the left hand is not free to activate quasimodes, without ending the transient interface phrase by lifting the left hand. For example, with the Ink Palette activated, users can still sketch with the pen and pan the camera, but cannot use selection, translucence, or edge blending.

By phrasing bimanual interactions with quasimodes, contextual operations, and transient interfaces, we support complex modal switching without the need for commonly used WIMP interfaces, such as menus or tool palettes. The use of modes in computing systems supports doing multiple things with one input device. While modes can support expert use, efficient mode switching can require remembering obscure key combination, such as those used in Emacs [Poller and Garter 1984]. Problems arise when users forget which mode they are in or become trapped in a mode, not sure how to escape [Tesler 1981]. Using Guiard's principles as guidance, we designed operations and modal states that con-

sistently use kinesthetic tension of the left hand to tie together various series of input as whole phrases. Removing the left hand from the interactive surface will return users to the default state. Thus, they can never be trapped, and the left-hand tension (combined with visual feedback) helps make clear the system's current state.

### III.7 IdeaMâché Platform

Emmâ is part of the larger IdeaMâché Platform. The IdeaMâché Platform consists of several clients that support diagramming and free-form web curation, of which Emmâ is one, and a cloud service for storing, retrieving, and sharing users' content created with any of the clients. Browser-based IdeaMâché[1], *Mâché* for short, is another client that runs in the Google Chrome web browser, allowing users to work from anywhere with Internet access. Diagrams created in Emmâ can be viewed and edited in Mâché, and vice versa. This cross-platform functionality allows participants in laboratory studies involving Emmâ or Mâché (e.g., Chapter IV) to take their creative work home with them. Participants can continue to explore their ideas, add new content, and make changes. If desired by experimenters, participants could work between laboratory sessions, producing more refined diagrams, as participants will have more time to engage with the software.

### III.8 Implementation

We implemented Emmâ in C# as a Windows 8.1 Store Application, using the Windows Runtime (WinRT) framework[2]. When Emmâ was initially developed, displays that supported pen+touch input where not commercially available. Thus, we developed Emmâ on specialized hardware. As pen+touch displays became available, issues—that were hidden by the specialized hardware—arose in the Windows OS for supporting simultaneous pen and multi-touch input.

[1] http://ideamache.ecologylab.net
[2] https://en.wikipedia.org/wiki/Windows_Runtime

### III.8.1 WinRT Framework

We selected the WinRT framework because of its ease of development for dealing with input events from different modalities, including pen, touch, and mouse, and its support for GPU-accelerated rendering. A prior version, running on the Windows Presentation Framework (WPF), had significant performance issues with text rendering.

#### III.8.1.1 Pointer Events

WinRT provides input event handling through pointer events. Whether input is performed via touch, pen, or mouse, the same event handlers are used supporting designing interactions that similarly when users have devices with different input modalities. However, pointer events provide information about which input modality raised the event, enabling us to design interactions where the pen does one action, while touch does another. For example, a single pen, by itself, draws ink. While, a single touch pans the camera or translates an element.

#### III.8.1.2 Win2D

During the iterative design of Emmâ, landscape architecture students created diagrams that pushed the limits of graphics performance with WinRT. They would draw sketches with hundreds of ink strokes, producing performance problems in interaction when panning and zooming the camera as these hundreds to thousands of ink strokes needed to be transformed and rendered. In order to gain increased control over the rendering cycle, we transitioned to using the Win2D library[3]. Win2D is a C# wrapper for Direct2D, allowing us to access Direct2D functionality in C# without needing a special C++ project.

[3]https://github.com/Microsoft/Win2D

### III.8.2 Hardware

Emmâ works with two versions of pen+touch hardware: (1) 21-24 inch Wacom Cintiq displays with ZeroTouch sensors; and (2) 22-27 inch Wacom Cintiq Touch displays. When we began development, high performance pen+touch devices at these sizes were not available. ZeroTouch is an in-house developed sensor that can add touch support to any display [Moeller and Kerne 2012]. Emmâ was originally developed using non-touch Wacom Cintiq displays (for pen input) that had a ZeroTouch sensor (for touch input). The Interface Ecology Lab developed a ZeroTouch driver for generating touch input events in Windows. This driver allowed us to use Pointer events in WinRT with the ZeroTouch. Later in the development cycle, Wacom released displays that support pen and multi-touch input. We obtained several of these Wacom Cintiq Touch displays, removing the need for a ZeroTouch sensor. However, due to limitations in Windows input event handling (see Section III.8.3), we still required specific hardware configurations for use.

### III.8.3 Supporting Simultaneous Pen and Touch

Not long into the development, we encountered a particularly troubling problem with Windows 8 input handling. It did not support simultaneous pen and touch input, which we considered an important requirement for our bimanual interaction design. As a form of naive palm detection in Windows, all touch input was canceled when the pen comes into range.

Our solution was to build our own touch event handlers for situations when the pen was also involved. The ZeroTouch sensor was generating touch events via the TUIO framework, a standard for sending touch input over UDP (the driver then converted TUIO into Windows touch). We implemented a listener that capture the TUIO messages. Then, based upon the position of the touches in the messages, we do hit testing to see what elements are under the touch, and fire custom event handlers on those elements.

In cases where the pen is not involved, we are still able to use Pointer events. In some cases where the pen can be involved, but may not necessarily, we have Pointer and TUIO event handlers calling the same code.

Windows 10 Anniversary Update added support for simultaneous pen and touch input. However, this was not available during the development of Emmâ.

# CHAPTER IV

# EVALUATION IN LANDSCAPE ARCHITECTURE DESIGN STUDIO

We developed Emmâ as an interactive system to support the design processes of creative visual thinkers, such as landscape architects, through embodied interactions with their ideas. Emmâ integrates various bimanual and unimanual interaction techniques, involving pen and multi-touch input, which together are designed to support designers engaging in creative processes of diagramming and curation. Our research question was to understand how this integration of bimanual and uni-manual techniques affects the design processes of creative visual thinkers. An evaluation addressing this question must investigate how participants intermix use of these interaction techniques and input modalities in design tasks. We argue that, when evaluating embodied interactions, it is important to not only observe participants' interactions with a system, but also the role of the body when performing those interactions and the body's relationship to any interactive spaces. For pen+touch interactions, this means observing the hands, fingers, and pen and their relations to a surface, including how the pen is tucked or stored when not in use.

Prior work has employed various methods for evaluating embodied interaction design. Human-computer interaction (HCI) researchers tend to adopt one of two methods for evaluating an interactive system: laboratory experiments [Bi et al. 2011; Hornecker et al. 2008; Matulic and Norrie 2012; Wilson et al. 2008; Wu and Balakrishnan 2003] or field studies [Antle et al. 2011; Benko et al. 2008; Hornecker 2008; Jacucci et al. 2010].

In laboratory experiments, HCI researchers invite participants to a controlled space, and ask them to perform tasks using interaction techniques. The tasks are defined by the researchers. For some experiments, participants perform short, repeated tasks that evaluate individual techniques in isolation [Bi et al. 2011; Matulic and Norrie 2012; Wilson et al. 2008]. In these tasks, participants are not stakeholders in what they are doing. Their

goal is only to complete the task. Data collection in these experiments primarily involves logging participant interactions, and computing metrics such as time and errors for comparison [Bi et al. 2011; Matulic and Norrie 2012; Wilson et al. 2008]. This form of data does not capture the relationships of the hands to interactive surfaces. In other laboratory experiments, participants perform longer tasks, using an entire system with multiple interaction techniques [Hornecker et al. 2008; Wu and Balakrishnan 2003]. These tasks tend to have some connection with the participants, to improve engagement and provide ecological validity, such as having members of a department plan seating arrangements for their department's move to a new building [Hornecker et al. 2008]. Data collection in these experiments involving observing what interactions participants perform and how they social interact if collaborating. Again, this does not capture the relationships of the hands to interactive surfaces.

In field studies, HCI researchers deploy systems in real-world environments to capture participant behavior "in the wild" [Brown et al. 2011]. The tasks are more open-ended than laboratory experiments, often involving walk-up-and-use scenarios, such as an exhibit in a natural history museum [Hornecker 2008]. Participants are not told how to interact or given a specific tasks. How the participants interact is part of the observational data collected. Analysis of observations tend to focus how participants understand the interface [Antle et al. 2011; Hornecker 2008], how they socially interact and collaborate with each other [Antle et al. 2011; Benko et al. 2008; Hornecker 2008; Jacucci et al. 2010], how engaged are participants [Hornecker 2008; Jacucci et al. 2010], and how well the sensing works [Antle et al. 2011; Benko et al. 2008; Hornecker 2008]. Once again, the data and analysis do not capture the relationships of the hands to interactive surfaces, except when the hands are in direct contact for interaction. In this case, Jacucci et al. [2010] made observations about which hands participants were using to interact with a surface.

Collecting observations for field studies can be challenging in certain contexts. Ex-

hibits have fixed locations and times when participants interact with them, allowing for premeditated arrangement of data capturing mechanisms, such as placing a camera to record participant interactions. Web-based interactive systems, such as Emmâ, are accessible from anywhere with Internet access. Capturing observational data about the body is challenging for these systems, since researchers have limited control over when and where participants use them. Researchers cannot insure data capturing mechanisms will be positioned correctly and activated for recording. Alternatively, controlled settings allow researchers to set up appropriate capture mechanisms, but may alter participants behaviors from real-world settings. The controlled tasks may not be of interest to all participants. In particular, designers performing arbitrary, albeit relevant, design tasks may employ strategies that require less effort, since the result will not directly aid design problems on which they are working.

We worked to define a study combining the advantages of controlled and field studies. That is, we wanted to capture all the rich data afforded by a controlled laboratory setting, while engendering the ecological validity of a field study. In concert with these to study approaches, we shift focus to how designers use their hands as they engage in pen+touch interaction.

Thus, we performed a hybrid investigation of Emmâ in landscape architecture design studio education. We invited participants to a controlled laboratory space in order to capture video data of their hands, synchronized with video of their creative work. We asked them to work on designs for course projects, providing a meaningful and real-world context. We did not give them a specific design task. We sought to observe how participants would make use of the suite of interaction techniques provided in Emmâ, and how they would engage in bimanual activity.

In this chapter, we first explain our evaluation methodology. Then, we present findings and discuss lessons learned. We conclude with implications for design for developing and

evaluating embodied interactions.

## IV.1 Methodology

We conducted a qualitative methods investigation of Emmâ in landscape architecture design studio education. We recruited participants from a graduate design studio course. They participated in laboratory sessions using Emmâ on course projects. We collected participating students' diagrams and video data of their interactions working with Emmâ. Afterwards, we conducted semi-structured interviews to understand their experiences. Using a visual grounded theory approach, we transcribed video data, developed codes, and identified emergent themes. We also engaged in qualitative analysis of the interview data, relating codes across study media. From our analysis, we derived implications for design of bimanual pen+touch interactions.

### IV.1.1 Participants and the Design Studio

We invited students in a landscape architecture design studio course, Professional Study, to work with Emmâ in our laboratory space. All three resulting participants (G1, G2, and G3) were female graduate students, majoring in landscape architecture. In this studio course, the students work on real-world design projects, which are initiated by industry sponsors. All participants were in the early stages of their design processes. They were still investigating design problems and potential solutions. While the participants had no prior experience working with Emmâ, they had extensive experience working with digital design tools, such as Adobe Illustrator and Photoshop. All participants had previously worked with surfaces supporting direct pen input, such as a Wacom Cintiq display or an iPad with capacitive pen.

### IV.1.2 Sessions

We conducted three sessions, each lasting from 60 to 90 minutes. G1, G2, and G3 participated in each session. Beforehand, we requested participants bring materials related to their projects with them. Participants worked individually, but were allowed to interact with each other (and they did), creating a studio-like setting. The first session began with a short tutorial that explained how to use Emmâ. Participants were then asked to use Emmâ to work on any suitable design studio project task. We did not define a specific task, preferring to let them define their own tasks, which would be productive for their design processes, and which would involve working with materials for their design studio projects. Participants were able to ask questions at any time, and free to leave whenever they wanted.

### IV.1.3 Laboratory Space

We created a laboratory space for evaluating Emmâ consisting of four workstations. Each workstation had a Wacom Cintiq display (for pen input) with touch sensing capabilities on an articulating arm. The arm allows the display to be oriented at different angles, like a drafting table. Workstations had high-end CPUs and graphics cards to support simultaneously running Emmâ and capturing data.

At each workstation, we positioned two cameras for data collection. One was located above, attached to the ceiling with an adjustable arm, providing a top-down view. The other was located to the side, attached to the desk with an adjustable arm, providing a profile view. Using two cameras insures that hands are not obstructing one another in at least one view. The adjustable arms allow us to easily re-position the cameras, if the participants need to change the display's position and orientation to make interacting more comfortable. Our video data capturing method uses Open Broadcaster Software[1] (OBS)

[1] http://obsproject.com

to integrate camera feeds, display capture of Emmâ, and touch sensor data into a single *embodied process recording* of a participant's session (see Figure 20).

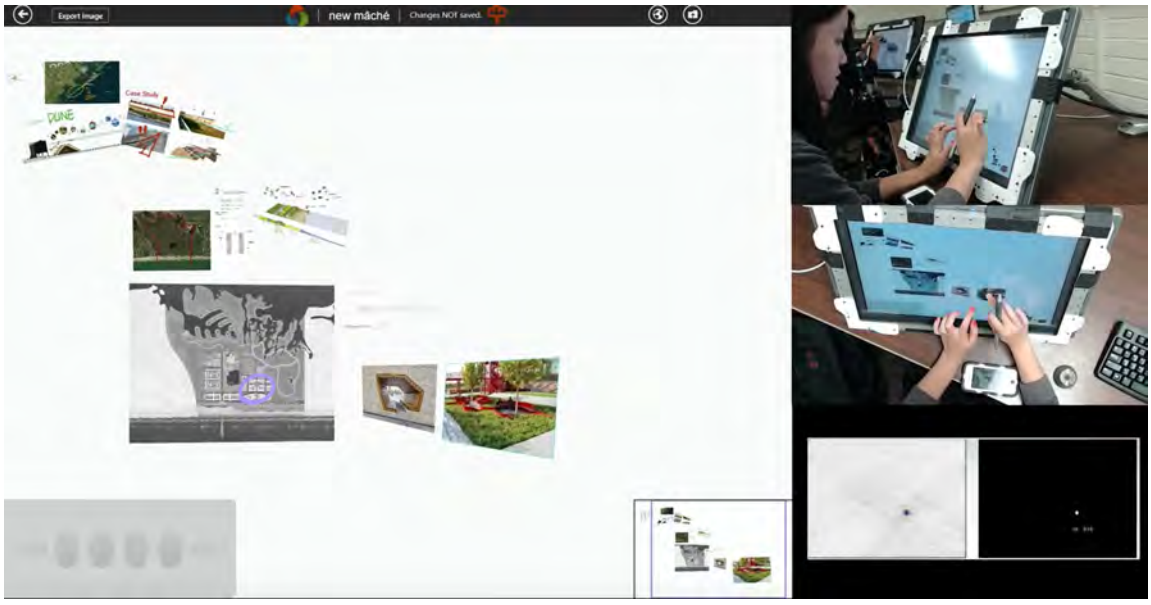


Figure 20: Snapshot from video data showing integrated display capture (left), two camera feeds (top-right), and touch sensor data (bottom-right).

### IV.1.4 Visual Grounded Theory Analysis

For each participant's session, we collected the diagram they created and the corresponding embodied process recording. We then used visual grounded theory methods [Konecki 2011; Lupfer et al. 2016] to analyze this collected data. Grounded theory is an empirical qualitative methodology, involving: gathering rich data, initial coding, theoretical sampling, focused coding, categorization, the incorporation of sensitizing concepts, and conceptual refinement [Charmaz 2014]. Rich data gathered for grounded theory primarily consists of field notes and interview transcriptions [Charmaz 2014; Strauss and Corbin 1998]. *Visual grounded theory* shifts the primary source from textual to visual

data [Konecki 2011]. Video data provides a visual version of an event, but resists, at first, reduction to categories and codes [Heath et al. 2010]. Yet, through selective and interpretive transcription, focusing on aspects related to research questions, we can apply visual grounded theory methods, like coding and categorization, to video data. However, the visual still remains primary as data, while transcripts compliment and help with analysis.

We first transcribed the embodied process recordings broadly, looking for phenomena related to our research question, of how integrated bimanual and uni-manual interaction affect design processes. From our transcription, we identified interesting uses of bimanual activity. We performed an initial open coding, which captured the actions performed, the hands used, and the resting position of the hands. We then performed a more focused coding, deriving 3 codes for processes of free-form web curation (gathering, assembling, and sketching) [Lupfer et al. 2016; Webb et al. 2016b], 3 codes for hands used (left, right, both), and 5 codes for left hand resting position (lap, hover, chin, float, desk). The transcription and initial coding were performed by two researchers. Meetings were held to discuss codes and phenomena. Once a specific set of codes was defined, one of the two researchers performed focused coding.

We also conducted semi-structured interviews with each participant, individually, a few days after the third session. Interview questions asked participants about their experiences using Emmâ and how it compares to other design tools they have used in the past. As needs and requirements gathering for iterative development, we asked participants what they felt was missing from Emmâ. During each interview, we displayed the participant's diagrams in Emmâ to stimulate discussion about their design processes. We used OBS to capture integrated audio recordings from a microphone and video capture of Emmâ with the diagrams being discussed. Interviews were transcribed and informally coded.

## IV.2 Findings

We develop findings by connecting analysis of three related forms of data: participants' diagrams, video of how they used their hands to create the diagrams, and subsequent semi-structured interviews. We look at each participant's diagram authoring processes. While each participant engaged in her own unique methods, we identify commonalities among participants' interactions with Emmâ. We observed several unexpected interactions by participants. From our coding, we further breakdown bimanual activity based upon processes of free-form web curation: gathering, assembling, and sketching. We note difference in how participants position their left hands to support their bimanual activity.

### IV.2.1 Diagram Authoring Process

Each participant created diagrams using Emmâ to ideate on design possibilities for her project. We present snapshot views of those diagrams, showcasing the roles of sketching over, translucence, and perspective in their design processes.

#### IV.2.1.1 G1

G1's first action in Emmâ was to sketch a diagram consisting of different shapes and patterns, including circles, rectangles, and arcs. She explored the expressiveness of making ink with the pen. In fact, the first act for all participants was to sketch. It quickly became clear that sketching is the most essential aspect of Emmâ for these participants. Sketching with the pen is familiar, expressive, and enjoyable. We suspect that directly sketching provides a familiar, inviting introduction to Emmâ, before jumping off into more complex interactions.

After sketching for a bit, G1 began composing a set of images related to her site in Galveston, Texas (see Figure 21). She had already gathered these images on a thumb drive, before coming in to participate. Using Emmâ's local image browser, she collected

some of these images into her diagram. She first sketched over the images, highlighting specific qualities related to design. She used zooming to magnify the images and make sketching over easier. She added arrows and handwritten annotations to exposit her ideas, such as labeling existing designs as good or bad.

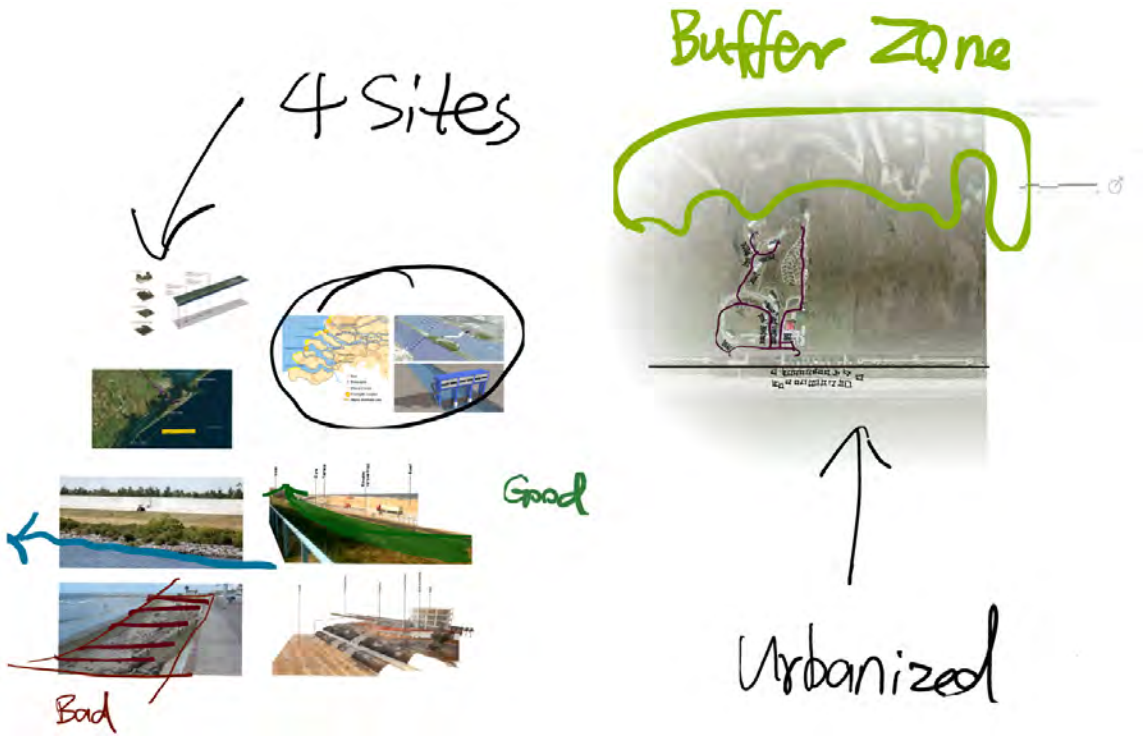


Figure 21: Part of diagram created by participant G1, where she gathered, assembled, and sketched over imagery. Handwritten annotations explain good and bad design solutions. Reprinted from [Lupfer et al. 2016].

In her second session, G1 gathered images from the web using the Integrated Web Browser. She used Google image search to find images of water plazas and Venice beach. After gathering an image, she would trace over that image, creating a contour drawing of the image's contents (see Figure 22). Lines extended beyond an image's bounds.

She would then drag the image away, leaving the drawing by itself. The tracing, which

Figure 22: Part of a diagram created by G1, where she traced over gathered images from the web. Over the left image, she outlines an interesting design. Over the right image, she traces contours and adds colored fills.

served as a quick way to sketch a scene, could then be modified and added to without the reference image. She selected different colors and a larger ink thickness. Then, she began filling in sections of the contour, exploring design ideas. She applied translucence to these fills.

In her third session, G1 continued sketching over images gathered from the Web. One gathered image presented a diagram for how vegetation around dunes on the coast can serve as protective barriers from strong winds and storms. She reconstructed this diagram using sketching and handwriting to text conversion. Text labels in the diagram were first hand written, then transformed to text. After reproducing the parts of the diagram, she pulled the reference image away, giving her a representation that she could, again, add to and modify.

#### IV.2.1.2 G2

G2's first action was to make ink, like G1. Again, sketching plays a fundamental role in the diagramming process. G2 experimented with different ink colors and thicknesses. In

one example, she created a dark region using a thick, black stroke. She then sketched over the dark region with thin, white strokes (see Figure 23), employing a reductive drawing technique [Aristides 2006], similar to working with charcoal, where one creates darkened regions with charcoal, then lightens them with an eraser.

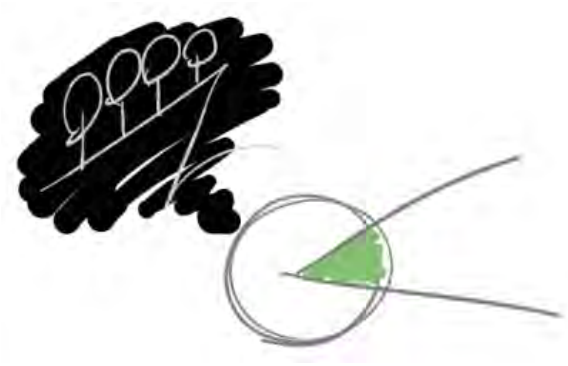

Figure 23: Example of reductive drawing by G2. She first sketched a dark region using a thick black stroke, then sketched over that region with a thin white stroke.

Next, G2 repeatedly gathered, assembled, and sketched over images, from her thumb drive, in Emmâ's diagramming space. She annotated a map of Galveston, highlighting the location for her site. She presented a case study by sketching over existing design solutions with teal and orange colored strokes, pointing out benefits (in teal) and problems (in orange) of each. G2 outlined the shape of the affected region for her site on an aerial photo that shows a transitional region (between areas of development) on Galveston Island.

In her second session, G2 brought in a site plan of a development, which is to be built in the transitional region that she outlined previously. She circled a specific section of the development for an industrial park. Then, she began gathering references images from the web of designs for seating, gathering, and offices. She applied similar perspective transforms to the images, assembling them so they appeared in the same plane. This

plane intersects with the site plan at the boundary of the region she outlined earlier (see Figure 24). G2 sketched over each image after applying a perspective transform to it. She traced out design possibilities, and annotated with handwriting and arrows. She drew a line connecting the tops of those images to the relevant part of the site plan. Below the perspective images, she drew a sectional view showing the arrangement of the dune, residences, office building, industrial labs, and wetland.


Figure 24: Diagram created by participant G2.

In her third session, G2 created a new diagram. She first gathered an image of an exemplary dune in the Netherlands. Then, she traced over that image, extracting out key features. Next, she dragged away the image, and began adding details to the tracing sketch. She used zooming to magnify, making it easier to add smaller details. She added a section view that connected with the perspective, showing different areas in the design (see Fig-

ure 25): development, detention, slopes, dune, beach, and sea. Following, G2 attempted to sketch human figures to provide a sense of scale. However, she was unsatisfied with her sketches, and opted to add an image found from the web.

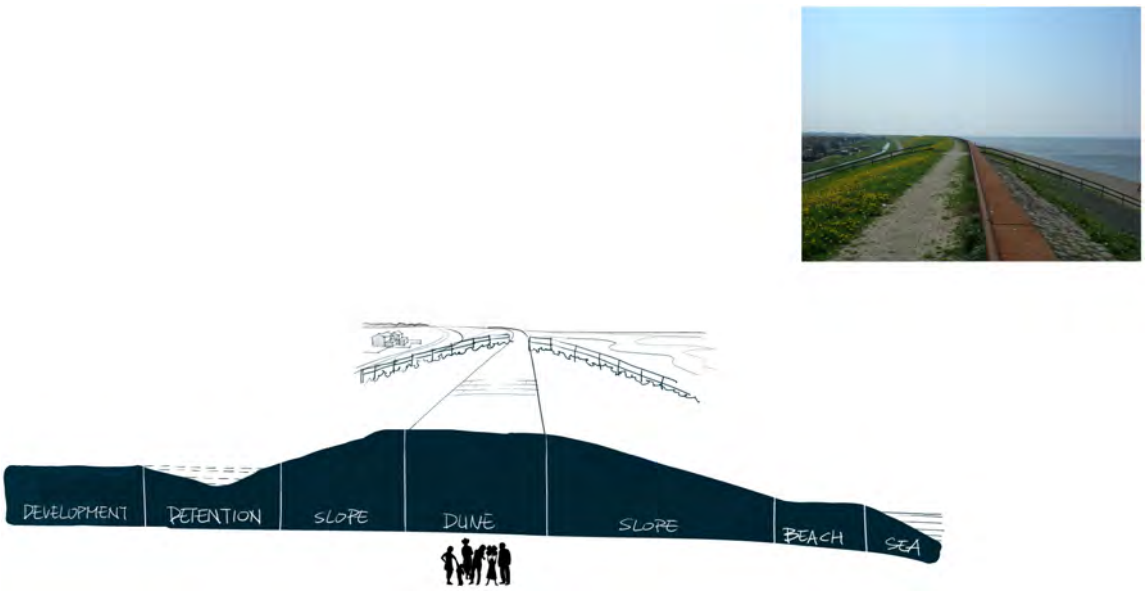


Figure 25: Diagram created by participant G2 in her third session. She traced over the image in the top-right, then added a sectional sketch connected with the tracing.

#### IV.2.1.3 G3

Just as G1 and G2 began with sketching, so did G3. However, G3 worked exclusively with sketching in her first session. She also made use of the handwriting to text conversion feature in the Ink Menu.

In her second session, G3 began by gathering (from her thumb drive) a diagram image related to her design problem, shrinking cities. She then began sketching handwritten annotations, using reductive drawing techniques, by placing some annotations in white over darker regions that were created with thicker ink strokes. She used opacity to improve

contrast between text and backgrounds. She created more explicit structure to convey relationships among concepts. G3 sketched arrows to connect related ideas. She assembled elements in columns with headings (see Figure 26). Her process was more focused on organization and planning than the other participants.

In the third session, G3 used the duplicate technique extensively to construct elements composed of a common element. She transformed the translation, scale, and rotation of duplicates. For example, the three concentric arcs connecting colored circles in Figure 26 were composed using one dashed arc, copied three times and scaled down for shorter radii.

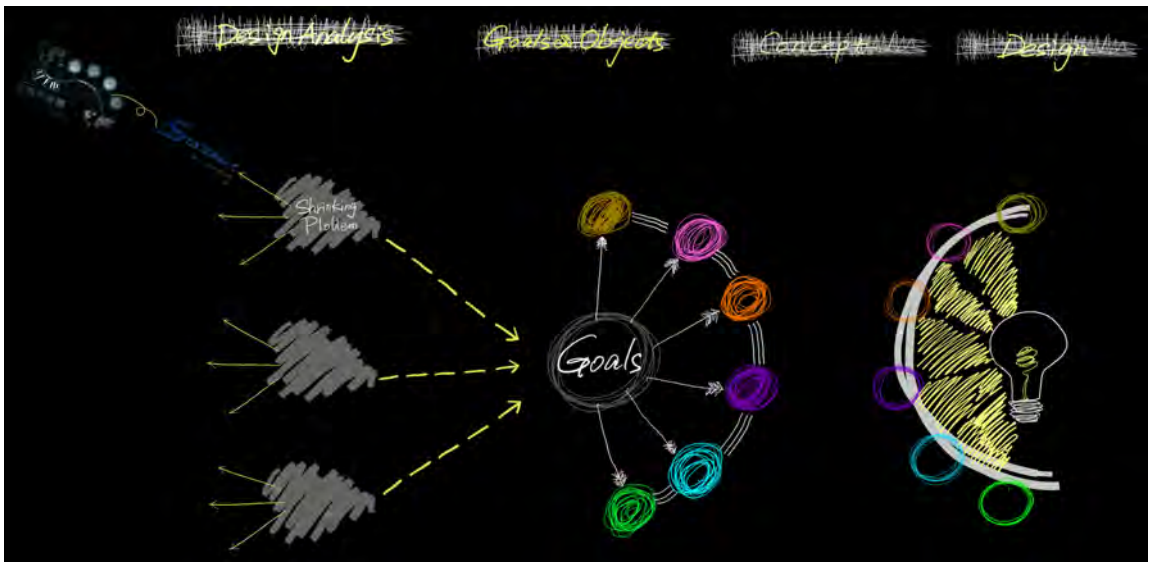


Figure 26: Diagram created by participant G3.

### IV.2.2 Common Phrases and Sequences of Interaction

Through our analysis of the video data, we identify common phrases and sequences of interaction that participants performed using Emmâ. All participants exhibited these common interactions, but with varying styles and effort.

#### IV.2.2.1 Sketching Over Imagery: Phrasing with Left-Hand Tension

One of the most common sequences was sketching over gathered imagery. A sketching over sequence begins with a phrase for first selecting the color and thickness of the ink using the Ink Palette. The left hand performs the right-angle gesture to activate the Ink Palette, and the right with pen interacts with the Ink Palette (see Figure 9). Next, the participant makes marks with the pen, typically sketching or tracing over parts of gathered imagery. For example, G1 marked out regions in photos of good and bad design, using different colored ink and stroke thicknesses (see Figure 21). G1 also constructed sketched representations of views for beach locations by tracing over imagery, and then removing that imagery (see Figure 22). The sketched representation could then be manipulated, possibly compositing it with photos taken from a site or sketching in new elements as part of design.

In some cases, participants would leave the Ink Palette activated in order to select different ink colors and thicknesses. Sketching sequences, phrased by holding the left hand gesture for Ink Palette, emerged as the participant would alternate between selecting ink color and thickness and making marks with the pen.

Other times, participants would lift their left hand after using the Ink Palette, but leave the left hand hovering in-place, ready to activate the Ink Palette again later. This hover state would phrase a larger sequence involving combinations of inking, using Ink Palette, and swiping the Hotpad for undo (see Section IV.2.3.2). Participants often used undo while sketching. In most cases, undo was used to remove the most recent stroke. Then, the participant immediately attempted a similar stroke. In rare cases, a participant would repeat cycles of creating a stroke and undoing that stroke until she created a suitable stroke.

#### IV.2.2.2 Two-Finger Stretchies: Bimanual or Unimanual

Emmâ supports using the well-known two-finger stretchy gesture to both translate-scale-rotate elements and zoom the diagramming space in and out. Users can perform the two-finger stretchy gesture with either one hand or both hands. Participants used a combination of unimanual and bimanual stretchies. However, the frequency at which they used one or two hands depended on the task being performed. For zooming, they often used both hands. While for transforming elements, they often used one hand, except when they wanted to perform a fine-grain transformation. For fine-grain transformations, particularly rotation, participants would used a bimanual stretchy. A bimanual stretchy allows for adjusting the two control points that define parameters of the transformation independently of each other. For unimanual stretchies, the muscular and skeletal connection between fingers on the same hand makes it difficult to move one without the other.

#### IV.2.2.3 Discarding Ink

When making ink, participants would sometimes discard ink using either undo with a swipe on the Hotpad to remove the previous ink stroke or pressing the Discard Ink button in the Inking Menu. Discarding with undo typically occurred immediately after an erroneous ink stroke was made. Participants would attempt to redraw a similar ink stroke that better represented their intended idea. Other times, the participant either had the wrong ink color or thickness. Undo removes just the last stroke, but can be repeated to remove all wet ink strokes (see Section III.3).

Alternatively, the Discard Ink button was typically used after making several marks. Discard Ink removes all wet ink strokes at once. Since the Discard Ink button appears in-place, it can be quicker to access than undo when the user's left hand is not near the display to swipe the Hotpad. Participants normally pressed the Discard Ink button with the pen. However, in one particularly interesting example, G2 tucked the pen, and tapped

the Discard Ink button with a touch. Then, she resumed sketching by untucking the pen. We suspect that tucking the pen could serve as another form of tension, phrasing alternate modes while inking. Others have used sensing pen orientation to support different interactions [Hinckley et al. 2014]. We discuss phrasing with pen tucking in more detail in Section IV.4.2.3 and Chapter VII.

#### IV.2.2.4 Pen Tucking

All participants exhibited a pen tucking behavior where the pen was stowed between fingers in the right so that touch interaction could be performed [Hamilton et al. 2012; Hinckley et al. 2014]. G1 almost never put the pen down, and the few cases where she did were the result of the application crashing. She typically kept the pen tucked in her right hand, when not using it. However, G1 did occasionally pass the pen to her left hand to make touch interactions easier with her right. She used a palm grip [Hinckley et al. 2014] to hold the pen in her left hand, wrapping her fingers around the barrel of the pen, instead of tucking the pen between her fingers. Participants would tuck the pen when doing both unimanual and bimanual stretchies (see Figure 20). For unimanual stretchies, participants most often used the index finger and thumb. However, G3 used a variety of different fingers with the pen tucked, including the middle and ring fingers.

### IV.2.3 Unexpected Interactions

We observed several unexpected interactions by participants. They used their multiple fingers to translate elements simultaneously without selection. G1 and G2 would hover their left hands above the surface when sketching to support quick modal switching. G3 appropriated the straight line tool as a straightedge for aligning elements.

#### IV.2.3.1 Translate Multiple Elements without Selection

Participants took advantage of multi-touch input to translate multiple elements simultaneously. Interestingly, we saw each participant translate multiple elements differently, using different combinations of hands and fingers.

Participants moved two elements using a single touch with each hand. For example, G1 was having problems using lasso selection for two overlapping elements. She instead selected the two elements simultaneously by placing a left-hand touch on one element and a right-hand touch on the other. Then, she dragged them simultaneously to the other side of the display. She made fine-grain adjustments to each, so that their relative positions matched their pre-drag states.

Participants moved multiple elements using different fingers from the same hand. For example, G2 used her middle and ring fingers on her right hand (with pen tucked) to move two images that overlap so that those images were aligned with other elements in her diagram. The elements were the most recently added and had not been precisely assembled. G2 used lateral movement of the hand with the fingers fixed to translate both elements. By keeping the her fingers fixed, she maintained the spatial relationship between the two elements while translating them both. Then, she used simultaneous movements of the fingers (similar to a two-finger stretchy) to change their relative positions to each other.

G3 combined bimanual and unimanual translation into one phrased interaction. Using the index, middle, and ring finger, she translated three elements with her left hand. Simultaneously, she used a touch with her right hand to translate another element (see Figure 27). Attempting to lasso this set of elements would likely result in the incorrect selection. Several elements resided between the three she moved with her left hand and the one she moved with her right, and would have been selected in a circular lasso.


Figure 27: G3 using multiple fingers and hands to simultaneous translate several elements.

#### IV.2.3.2 Fluid Mode Switching with Left Hand Hover

G2 demonstrated effective mode switching by hovering her left hand near the Hotpad (see Figure 28), allowing her to transition among undo, selection, and Ink Palette usage. By working near in the bottom half of the display, near the Hotpad, G2 entered what we identify as a fluid state, completing many different operations in a short period of time. This fluid state occurred when creating the sectional view for the industrial park. Her operations were centered around sketching. She used different ink colors and strokes. She undid strokes that did not match what she desired. By keeping her left hand hovering, she could quickly swipe the Hotpad or bring up the Ink Palette with the right-angle gesture.

We also observed G1 repeatedly hovering her left hand, particularly after using the Ink Palette. However, her left hand did not hover as close to the interactive surface as G2's did. After performing bimanual operations besides Ink Palette, she would often rest her hand on her lap or the desk. The act of sketching and the need to quickly access the Ink Palette seemed to cause her to leave her left hand hovering. She would also occasionally

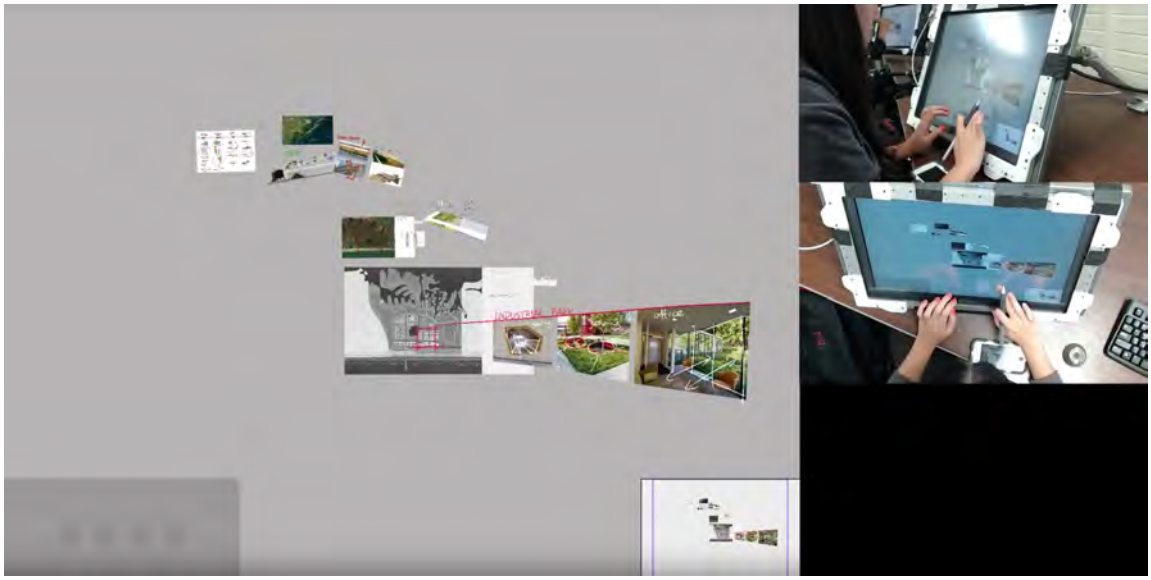

Figure 28: G2 engaging in fluid state switching between Hotpad undo and Ink Palette activation. Her left hand remained in a hovering state near the bottom-left corner of display.

transfer the pen from her right hand to her left, so that she could use touch input with her right hand. With her left hand hovering, this was a fast transition for her. Conversely, the other participants tucked the pen in the right hand to perform touch input.

#### IV.2.3.3 Appropriate Straight Line Tool for Alignment

G3 created a line with the Straight Line Tool to use as a guide for aligning content. The Element-defined Grids technique was not yet implemented in Emmâ. Pixel-precise alignment was not possible, as elements would not snap to the straight line. G3 was still able to appropriate a straight line object for alignment. She used a horizontal line to vertically align elements along their tops and bottoms (see Figure 29). She also used the straight line during element creation to help match sizes for her symmetric representation of circles. She first positioned a straight line so that it was aligned with the top of an element. Then, she began creating a new circle with ink, starting the ink stroke on the guideline.

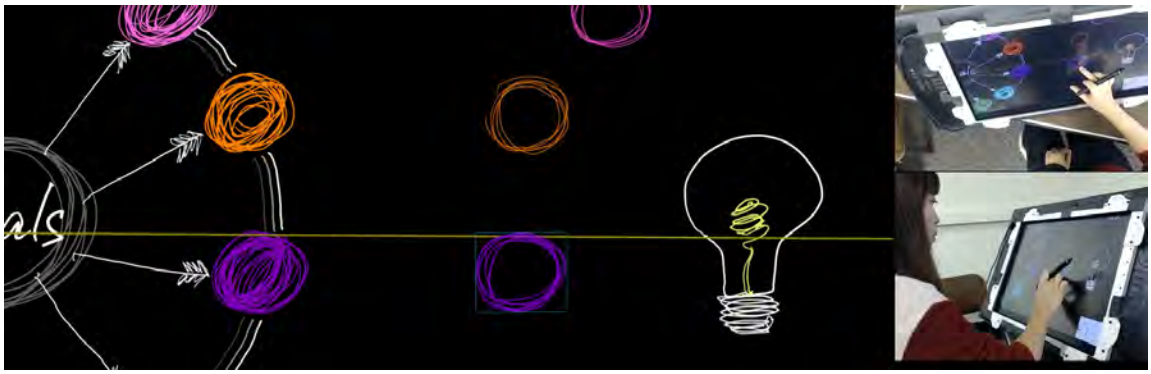

Figure 29: G3 used a straight line to aid in aligning elements. She created the horizontal yellow straight line. Then, used it to vertically align new elements with previously assembled ones. She positioned of new elements so their tops was along the yellow line. She continued to use the same yellow line for all alignments.

### IV.2.4 Bimanual Activity by Processes of Free-form Web Curation

We coded embodied process recordings for use of the hands in three free-form web curation processes: gathering, assembling, and sketching. Participants spent less time gathering. They spent most of their time sketching. Gathering interactions involved bringing in reference images. Assembling interactions involved organizing elements in space space, such as scaling and aligning elements into groups. Sketching interactions involved sketching ideas and tracing over imagery.

#### IV.2.4.1 Gathering

While gathering interactions, such as clipping an image, may be performed bimanually, participants primarily used one hand for these operations. Participants used their right hands to select, clip, and drag media from the Integrated Web Browser. Selection typically requires precision. Thus, participants used their preferred, right hands for these operations. We also observed rare instances in which participants would select media with the right hand, and then clip and drag with their left hand. A few times, G1 scrolled the web browser using her left hand and made selections using her right.

During gathering, the Integrated Web Browser and Local Image Browser occluded a significant area of diagramming space on the right-hand side. We suspect this may have influenced participants interactions. We observed examples where participants would pan and zoom the diagramming space with their left hand to orient the space for gathering content with their right hand.

The most common example of bimanual activity in gathering interactions was typing on the soft (touch-based) keyboard. Participants used the keyboard to perform search queries and to edit text annotations.

#### IV.2.4.2 Assembling

When assembling content, participants used both hands to perform operations almost as much as using one hand. The most common examples were applying two-touch manipulations to scale and rotate elements or zoom the camera in and out.

G3 in particular alternated among bimanual and unimanual activity when doing assemblage. She engaged in frequent zooming between micro and macro views. She used both hands to zoom in and out, then would translate and scale elements with different hands. For G3, the relative location elements in the current view seemed to influence which hand she used to perform assembling operations. She often used her left hand for elements on the left side of her view and her right hand for elements on the right side. In one particular example, she exclusively uses her left hand to zoom in and scale and align two elements located on the left side, zooming out to see the macro view, then zooming back in and manipulating the elements some more. She uses her index finger and thumb for zooming, and her middle and index fingers for scale and rotation of elements.

G1 spent the least amount of time assembling, but still often used bimanual interactions. She frequently used the bimanual two-finger gesture to zoom in and out. She also used both hands when doing fine-grain scale and rotation on elements.

In her first session, G2 primarily used one hand when assembling. However, in the other sessions, she began to use both hands when assembling. Her bimanual activity seemed to increase with the use operations that require both hands, such as lasso selection and perspective transform. She began to use both hands for scaling and rotating elements and zooming in and out the view.

#### IV.2.4.3 Sketching

For sketching interactions, participants used a mix of unimanual and bimanual interactions with the majority being unimanual. Making ink strokes with the pen—the most frequent operation that participants performed while sketching—is a unimanual operations. Other unimanual operations included undo and discard ink. Bimanual operations were either using the ink palette or zooming in or out. The frequency of bimanual activity depended on how often participants were changing ink styles or adjusting their view. For interactions in this free-form web curation process, we observed similar ratios of unimanual and bimanual activity among the three participants.

### IV.2.5 Left Hand Positioning

Participants exhibited differences in where they positioned their left hand when not performing bimanual interactions. Each participant had their own commonly used positions.

#### IV.2.5.1 G1: Lap and Hover

G1 primarily positioned her left hand on her lap, but also frequently hovered her left hand above the interactive surface. In periods of frequent bimanual activity, she would use the hover position, such as when frequently having to change the style of ink while sketching. In other periods, she would rest her hand on her lap. Yet, she continued to engage in bimanual interactions with her left hand, lifting it off her lap to perform operations and

returning to her lap afterwards.

#### IV.2.5.2 G2: Hover and Desk

G2 primarily positioned her left hand hovering above the interactive surface, but also frequently rested her left hand on the desk. Similar to G1, her left hand hovered when engaging in more bimanual activity, particularly when sketching. G2's hover position was often low, near the bottom-left corner of the display. Her forearm would at times rest against the desk to provide stability and reduce fatigue. As previously discussed, this hover position was near the Hotpad, enabling quick access to undo and lasso selection.

#### IV.2.5.3 G3: Lap

G3 primarily rested her left hand in her lap. Surprisingly, she exhibited a greater ratio of bimanual to unimanual activity than the other participants. We suspect this is, in part, due to her focus on sketching and assembling, while only doing a small amount of gathering—the process that was most unimanual among the participants. She also demonstrated greater ambidexterity than the other participants. She used combinations of unimanual interactions with either hand for translating and scaling elements and zooming in and out.

### IV.2.6 Interviews

From their interview responses, participants experienced Emmâ as helpful to their design processes. In particular, they felt that Emmâ made it easier for them to gather aspects of design, and sketch amidst those gathered elements, in comparison with prior design tools.

> G3: *[Emmâ] is better for us to combine diagramming and sketching, compared with traditional software, such as Illustrator and Photoshop.*
>
> G2: *I think this program is suitable for concept design, collecting all the*

*concepts. And, after we design, we need to produce the perspective. And, this one helps us to trace, like tracing paper, we can trace something on the graphic, so we can have a more accurate perspective.*

All the participants compared Emmâ with the role of tracing paper for their design processes with physical media. They use tracing paper to sketch over images that they find on the web. However, tracings on paper are less malleable than a digital representation. With Emmâ, they found that they were able quickly capture ideas from existing designs by tracing over images gathered from the web. They could then modify those tracings by adding in additional content. This enabled them to do before and after comparisons as they explored design possibilities. In contrast, tracings on physical paper become separated from the original source image, making it difficult to do comparison.

G1: *[In Emmâ,] I can trace an image, because sometimes we grab ideas from another design or image from a website. That is very useful thing. For example, I can make a perspective based on this image, and maybe make some changes. Like add more things, like more flowers, trees, something like that. So we can have before and after design comparison photos.*

G2: *The search for online graphics really helps. Because, if I search on computer and use tracing paper, it will be separate. But [with Emmâ], it is together. It makes the process easier and faster to produce.*

Participants valued the holistic diagramming space where they could gather their ideas in a continued process. By having it all together, they can look back over and reflect on their design process.

G2: *[Emmâ] will make it more continued. I can see what I have done from the site analysis to providing solutions to design process. It will be more continued, and I can see the process of my design, and it will be more clear.*

All participants requested a way to see and manipulate layers more directly. All the existing design tools they work have representations of layers that they can manipulate.

> G3: *I want to compare some layers. Make it as a group. If we can use our fingers to do that, it would be much better.*

We developed a technique for layering overlapping content with pen and touch in Emmâ, called LayerFish (see Chapter V), based upon this request from participants.

## IV.3 Discussion

The findings show that Emmâ supports early, conceptual stages of design, where sketching and tracing play pivotal roles. Participants effectively used our bimanual phrasing techniques, especially for sketching. However, we see methods for improving upon our use of phrasing for gathering operations. Our hybrid investigation provided interesting findings. By capturing embodied process recordings, we were able to discover variations in the roles of the hands for bimanual-symmetric and uni-manual interactions.

### IV.3.1 Sketching as Icebreaker

All participants began using Emmâ by sketching something. Sketching is familiar to the participants. The value of sketching in design is well documented [Buxton 2010; Goldschmidt 1991; Robbins and Cullinan 1994; Suwa et al. 2001; Verstijnen et al. 1998]. It provided an icebreaker, before delving deeper into the more complex operations supported in Emmâ. As suggested by Hinckley et al. [2010] for pen+touch interaction, the pen, by itself, always makes marks. This makes sketching very easy to perform in Emmâ. No guidance is required.

When sketching, stylized ink quickly becomes important for expressing ideas. Participants moved into using the Ink Palette as part of this early sketching icebreaker. The Ink Palette provided an introduction to our Guiard-abiding phrasing approach, where the

left-hand initiates interaction, and through sustained tension, frames the operation. Both G1 and G2 began to hover their left hand near the interactive surface, particularly when using Ink Palette. This hovering created tension for sketching, which could play a role in phrasing larger sequences, such as automatically aggregating elements that make up a larger sketch. The progression from sketching to Guiard-abiding phrasing with the Ink Palette draws users in and so helps them to learn to engage in bimanual interaction.

### IV.3.2 Tracing with Context

Participants reported value in being able to quickly trace over images. We observed participants frequently engaging in tracing over images gathered from the Web. This ability mimics existing practice with physical media, where architects use tracing paper to trace over printed out imagery or directly off a computer display [Laseau 2001]. An advantage of Emmâ, as reported by the participants, is the ability to keep the source and tracing connected. In Emmâ, gathered elements from the Web maintain referential links back to the source documents from which they were clipped. This linkage connects the user's diagram with ideas and materials on the Web. Currently, links between tracings and images are implicit through spatial relationships in the diagramming canvas. However, we could support more explicit connections, by creating referential links, like those created for gathered content, between ink strokes and the image that was traced. We then could provide ways to support visual comparison among the tracings, the originals, and the modified tracings.

### IV.3.3 Bimanual Phrasing

We designed bimanual interactions that tie together a series of inputs as a phrase through kinesthetic tension. We observed participants making effective use of our phrasing to perform complex modal switching. This benefit was particularly evident when performing sketching operations. For gathering techniques, we did not see significant bimanual

activity.

#### IV.3.3.1 Effectiveness in Sketching

Participants performed sketching the most among the free-form web curation processes that we coded. The common operations were make ink, use ink palette, use ink menu, and undo. Additionally, they occasionally zoomed briefly out and then back in to get a more holistic view while sketching out ideas. Both G1 and G2 hovered their left hands to support quicker modal switching. Sketching with pen and paper is a bimanual task. The left hand orients the paper for the right hand with a pen to make marks. We wonder if the physical experience of pen and paper played a factor in the observed behavior of participants to hover their left hands while sketching.

#### IV.3.3.2 Improving Gathering Techniques

We observed the least amount of bimanual activity for gathering sequences. We suspect that this is largely due to the lack of phrased bimanual interactions. Unfortunately, limitations in software APIs prevented us from designing more bimanual actions. Without these limitations, we envision phrasing gathering techniques, which the left hand defines contents for gathering, and the right hand places and transforms the gathered content in the diagramming canvas. Alternatively, we could use the duplicate operation for clipping content. In other words, two touches with the left hand on the selected content, followed by a touch drag with the right hand. This would prevent having to introduce any new gestures, and help reinforce the duplicate gesture as how to phrase copy operations. For either gathering approach, we could employ pen-based selection gestures for HTML content [Eichmann et al. 2016] to allow users to specify what parts of a web page to gather.

We did observe some division of labor between the hands, where the right hand clipped content from the Integrated Web Browser, and the left hand panned and zoomed the view. This was likely due to the clear division of the interactive surface with the Integrated Web

Browser on the right side and part of the diagramming canvas on the left.

### IV.3.4 Hybrid Investigation

Our investigation combined aspects of laboratory experiments and field studies. We needed a controlled setting to capture embodied process recordings, but wanted to observe participants using Emmâ on "real-world" design tasks. The tasks performed by participants in a study affect the results and findings. Picking appropriate tasks can be challenging, particularly for open-ended creative tasks—such as those associated with design—where engagement [Carroll et al. 2009] and personal motivation [Amabile 1985] directly influence participants' creativity. Design involves a personal lens, in which a designer draws on her past experiences, knowledge, and interests to help come up with new ideas [Daly et al. 2012]. Thus, we chose to have participants choose their own tasks, with only a restriction that they work on materials related to their design studio projects.

We argue that this approach to evaluating embodied interactions provides ecological validity for studies where the participants are stakeholders in the outcomes beyond the study itself. What participants create in this type of study continues to have value after the study has ended. By letting participants choose their tasks, we promote their personal motivation and engagement. At the same time, we collect observations on what aspects of a system are most valuable to the participants and what processes it supports. A potential risk of giving participants choice is that they may exhibit different behaviors, making it difficult to draw conclusions. In our study, we recruited from a single design studio to ensure that data gathered would contain some commonality among the participants for comparison. While participants were each working on individual projects, they shared common goals as landscape architecture students in the same design studio course.

### IV.3.5 Embodied Process Recordings

Capturing video of both the participant's design process and their body movements resulted in findings that would not be possible with logged interactions alone. Integrating these different sources into a single video enabled us to employ visual grounded theory methods on participants' embodied processes. Researchers used the capture of Emmâ to transcribe participants' actions without the occlusion of the hands, while simultaneously transcribing the activities of the hands on and around the surface. We derived codes about the use of hands in relation to different processes of free-form web curation and the positions of hands when not interacting with the surface. We observed interesting variations in the use of hands for performing two-finger stretchies.

Our participants exhibited both common and unique interaction styles. Participants often used two hands to zoom and one to translate-scale-rotate elements. The exception is when performing more precise transforms on elements. In this case, participants used both hands, as they needed to manipulate only one of the control points, which is difficult with one hand. They used their hands and fingers in different ways. They demonstrated different states of flow, guided by left-hand position and free-form web curation processes. Using only logging data, these characteristics of their interactions would be lost.

## IV.4 Implications for Design

We derive implications for design of bimanual interactions based upon findings from our evaluation in landscape architecture design studio education. First, fluid phrasing of bimanual interactions through quasimodes, transient interfaces, and contextual operations can support effective post-WIMP interaction design. Second, we can further phrase bimanual interactions using contextual information provided by sensing fingers, hands, and pen orientations. Next, phrasing macro and micro view transitions in zoomable user interfaces can support design processes involving reflection on the design as a whole in tandem

with specific aspects. Finally, hybrid studies with embodied process recordings enable evaluations of embodied interactions for creative tasks.

### IV.4.1 Quasimodes, Transient Interfaces, and Contextual Operations

Interaction designers would benefit from considering how different forms of phrasing can be combined into more complex sequences required to perform compound tasks, such as creating a sketch element with different ink styles. Participants exhibited fluid states where they quickly transitioned among quasimodes, transient interfaces, and contextual operations. For example, G2 hovered her left hand near the Hotpad, allowing her to switch between activating the transient Ink Palette and swiping and touching the Hotpad. We did not originally consider this in our design of the techniques. Just as choreographers must consider how phrases combine into sequences and how that will affect dancers, so to must interaction designers consider how users might compose phrased interactions into sequences.

The spatial constraints of the Hotpad allowed participants to quickly switch among its quasimodes and other phrased interactions involving transient interfaces or contextual operations. However, this required that participants work near the Hotpad for quick transitions. Quasimodes activate a global state, where all interactions are within a specific mode. By using an edge-constrained region for activating quasimodes, we seek to prevent the left hand from occluding or interfering with actions of the right. An alternative strategy would be to combine alternate sensing methods (see Section IV.4.2) to support phrasing quasimodes contextually, such as through in-air gestures or different postures for holding the pen [e.g. Hinckley et al. 2014].

### IV.4.2 Additional Sensing to Support Phrasing

Interaction designers would benefit from taking advantage of additional sensing, such as identifying the hands or fingers, to support phrasing. Participants demonstrated use of

different hands or fingers when performing two-finger stretchies to either zoom the space or scale and rotate elements. The position of the hands and pen when not being used also could provide contextual details to phrase interactions.

#### IV.4.2.1 Identify the Hands

Participants often used both hands to zoom in and out, but only one hand to translate, scale, and rotate elements. Currently in Emmâ, the user must touch empty areas of the diagramming space to zoom in and out. If the user touches an element when attempting to zoom, that element will be affected instead. With the hands identified, two-finger stretchies could be designed to zoom—even when touching elements—when two hands are used. This can be particularly useful when the user scales an image so large that it hides any empty space at the current zoom level, making it impossible to zoom without first shrinking the large element to reveal empty space. We present a method for identifying the hands on multi-touch surfaces using wearable sensing in Chapter VI.

#### IV.4.2.2 Identify the Fingers

When fingers are identifiable, interaction designers can phrase interactions based upon which finger is touching. For unimanual interactions, G3 used her index finger and thumb for zooming and her middle and index fingers for scaling and rotating elements. These contextual operations are phrased by spatial constraints, but could be phrased by kinesthetic constraints of which fingers are used.

Identifying the fingers could more easily support chorded gestures [Ghomi et al. 2013] for activating transient interfaces and contextual operations. For examples, recognition of the right-angle gesture for activating the Ink Palette could be made more robust and less complicated if the fingers were identifiable for touch input.

#### IV.4.2.3 Pen Tucking

Interactions designers would benefit from making use of pen tucking to support phrasing interactions. For example, in Emmâ, tucking the pen could identify the end of sketch, activating the Ink Menu. Currently Emmâ uses a timeout for activating the Ink Menu. The time between ink strokes varies, so a timeout is not ideal and can cause interaction issues. In the worst case, the Ink Menu appears just under where a user is about to make another ink stroke. Timeouts do provide kinesthetic tension either. Tucking and untucking the pen could show and hide the Ink Menu.

Participants would not always hold the pen. Thus, pen tucking as a phrasing technique requires designing around interactions that use the pen, such as sketching. Otherwise, designers need to support two methods for phrasing a single operations, one that requires the pen and one that does not. Certainly, providing greater flexibility in how operations are performed will better support the unique ways of interacting in which our participants demonstrated. The downside is added complexity in interaction design.

### IV.4.3 Phrasing Zooms between Macro and Micro Views

Zoomable user interfaces for design environments would benefit from supporting user-defined transitions between macro and micro views. All our participants worked across scales, zooming in, then out, then back in again. While Emmâ supports this interaction through two-finger stretchy zooms, it requires that the user specify repeated zoom amounts for each zoom action. One possible improvement would be to support phrasing spring-loaded zoom modes, where participants could zoom in or out temporarily, afterwards reverting to the previous zoom level. This could be phrased using sustained kinesthetic tension. We can use the same two-finger stretchy, but we phrase a spring-loaded zoom when the user holds the touches down after she stops zooming. In a normal zoom, the touches would be lifted almost immediately upon zoom completion. When the touches

are lifted during spring-loaded zooming, the tension is removed and the zoom level is reverted to the previous level with an animated transition.

### IV.4.4 Hybrid Studies and Embodied Process Recordings

Evaluations of embodied creativity support environments, such as Emmâ, would benefit from performing hybrid studies. A controlled laboratory settings allows for capturing embodied process recordings, while giving participants choice in what tasks to perform supports ”in the wild” use similar to field studies, improving ecological validity. Thanks to IdeaMâché’s web-based platform, participants were able to take what they created in our study with them as they continued their work elsewhere (see Chapter III, Section **??**). This aspect is key to supporting participant engagement for long-term processes, such as design, especially when early-stage representations are often referred back to in later steps.

We argue that the evaluation of embodied interactions requires observing relationships of the body to the interactive space—a surface for pen+touch. The embodied process recordings allow for visual grounded theory analysis of these relationships between body and space. Our findings would not be possible without this integrated data collection method. We envision using human motion sensing, via depth cameras or inertial measurement units (IMUs), as an additional component of embodied process recordings (see Chapter VII).

# CHAPTER V

## LAYERFISH: BIMANUAL LAYERING WITH A FISHEYE IN-PLACE*

Designers work with many visual elements, from 10s to 100s, in large 2D spaces. Elements overlap, requiring reordering of the visual stack, or *layering*, to place some elements in front of or behind others. As the number of overlapping elements increases in scale, the complexity of layering also increases. This requires more expressive forms of interaction than the common "bring to front" and "send to back" contextual menu options provided in tools such as PowerPoint.

Design tools, such as Adobe Illustrator, support complex layering through a *scene index*—an ordered list showing the visual stacking order of elements. Each element in the 2D design space is represented in the scene index by a *correspondent* thumbnail. Correspondents can be reordered to adjust the visual stacking of elements. They can also be selected to constrain direct manipulation to only corresponding elements, even those occluded by other elements. As the number of elements increases, the scene index must afford scrolling. Scrolling becomes tedious when working with hundreds of elements. Further, the index is often located out of the user's visual focus, requiring her to split attention between the design space and the index.

Prior work developed alternative techniques that provide *in-place* layering, addressing the split attention shortcomings of the scene index [Hinckley et al. 2013; Ramos et al. 2006]. However, these techniques are designed for working with a small number of elements, and will not scale well in design spaces with hundreds of elements.

*LayerFish* is a pen+touch interaction technique to support layering and manipulating

*Edited reprint with permission from "LayerFish: Bimanual Layering with a Fisheye In-Place" by Andrew M. Webb, Andruid Kerne, Zach Brown, Jun-Hyun Kim, and Elizabeth Kellogg, 2016. In *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces*, 189-198, DOI: `https://doi.org/10.1145/2992154.2992171`. Copyright 2016 by Webb, Kerne, Brown, Kim, and Kelloggx. Publication rights licensed to ACM.

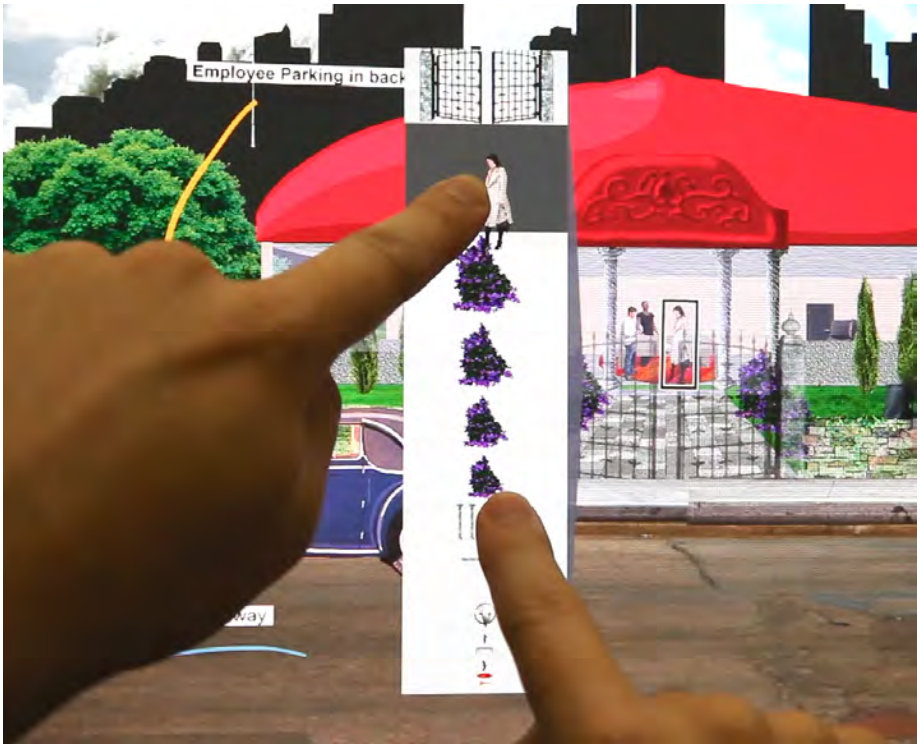


Figure 30: Example of bimanual interaction in LayerFish. The left hand touch has a correspondent selected and held in place, while the right hand drags to scroll the fisheye scene index.

overlapping content in a 2D design space on desktop and large collaborative surfaces (see Figure 30). LayerFish addresses issues of scene index scale and split attention through a fisheye visualization [Furnas 1986] and in-place positioning of the scene index at the user's point of focus. The fisheye distorts the visual space, decreasing the sizes of correspondents away from a focus element, and so enables more to be visible than a typical scene index. However, interaction issues arise, as the spatial distortion disrupts layering and scrolling operations. We develop bimanual techniques to address these issues, which keep the focus element fixed while the scene index reorders layers and scrolls. We hypothesize that the fisheye will reduce time to find desired correspondents, since the spatial distortion would help scrolling over large distances.

Furthermore, LayerFish advances bimanual interaction design for manipulating overlapping content. Overlap produces occlusion that interferes with manipulation. This is exacerbated by fat finger and Midas touch issues, which arise with touch input [Holz and Baudisch 2011]. In LayerFish, an occluded element is activated by touching its correspondent, with the non-preferred hand. Once activated, only that element can be directly manipulated in the design space, with the preferred hand.

This chapter begins with a discussion of prior work. Next, we present LayerFish. We follow with an evaluation comparing performance of LayerFish—when working with a large number of elements—to a traditional scene index. We discuss findings and derive implications for design.

## V.1 Related Work

Various layering techniques exist. Commercial design tools typically use either pop-up menus or scene indexes. HCI researchers have developed alternatives, investigating how spatial arrangement [Grossman et al. 2009; Herrlich et al. 2011; Leithinger and Haller 2007; Ramos et al. 2006] and sensing modalities [Davidson and Han 2008; Hinckley et al. 2013] can support interacting with dense and occluded content.

### V.1.1 Commercial Design Tools

Many commercial design tools employ a contextual menu (e.g. Microsoft PowerPoint). The contextual menu includes several commands, not all of which are for accessing occluded content. In mouse-based interfaces, the contextual menu is activated by right clicking an element. The user must navigate a hierarchical menu and find the command to bring an element forward or backward. In touch-based interfaces, the contextual menu is activated by touching an element. Commands appear in a menu bar. The contextual menu approach requires the user to perform input on an element, which may be occluded, making it challenging to activate.

Alternatively, a scene index provides an ordered list of visual layers, which can be rearranged to adjust the z-ordering of elements. As the number of layers grows, the index becomes large. It can become difficult to find a layer. Thumbnails for each layer are used to support recognition. The thumbnails are all the same size, making it difficult to differentiate similar elements of different sizes [Ramos et al. 2006]. LayerFish addresses issues of scale through a fisheye visualization and user selection.

### V.1.2 HCI Researchers

Ramos et al. explored techniques that maintain the shape and size of elements, while creating spatial separation between layers, enabling novice users to interact with occluded content [2006]. LayerFish addresses issues of scale while also maintaining element shape.

Herrlich et al. developed a touch technique for selecting occluded content [2011]. The user resolves selection ambiguity by touching proxies arranged radially around the point of interaction. Handle Flags is an in-place technique for selecting overlapping ink strokes without complex lasso selections [Grossman et al. 2009]. For LayerFish, the user needs to not only select the element to be layered, but also nearby element(s) that it will be layered above or below. Therefore, we use lasso for selecting a region rather than one of these precise selection approaches.

Pen and multi-touch sensing technologies support additional input parameters that can be used to access and manipulate occluded content. Davidson and Han used touch pressure and points of contact to define layering gestures [2008]. Hinckley et al. combined touch to select an element with pen tilt to reveal occluded elements [2013]. These techniques are designed for working with a few overlapping elements.

Leithinger and Haller address occlusion issues of context menus on cluttered tabletops through user-drawn arrangements of menu items [2007]. LayerFish's in-place scene index raises occlusion issues that could be mitigated through user-drawn arrangement. However,

new interaction issues arise as scrolling the scene index and re-ordering correspondents involves two dimensions instead of one.

### V.1.3 Kinematic Chains

Guiard developed a model of bimanual interaction, the kinematic chain, to describe the asymmetric division of labor between the hands [1987]. In a *kinematic chain*, the left hand acts to define a reference frame for the actions of the right hand. When drawing on paper, this is equivalent to the left hand positioning the paper in conjunction with the right hand making marks with a pencil. Hinckley et al. derived new pen+touch interaction based upon kinematic chains: the left hand gestures to define the operation performed by the right hand with a pen [2010]. We build upon Guiard and Hinckley et al., by designing new kinematic chain gestures that specify elements to be manipulated in the design space with the right hand via touching correspondent representations in a scene index with the left hand.

## V.2 LayerFish

LayerFish is a bimanual interaction technique designed for desktop and large surfaces, including tabletops, where both hands are free to interact on the surface. We found, through discussions with visual designers, that these form-factors are commonly used when working with many layers. Smaller form factors are also used, but tended to involve fewer layers. The LayerFish technique consists of a selection and activation gesture (quasimode), a fisheye scene index (transient interface), and interactions for layering elements and manipulating occluded content (contextual operations).

### V.2.1 Selection and Activation

Using LayerFish begins with selecting elements on which to operate, followed by an activation gesture. Selection specifies the elements active in LayerFish. Unlike a tra-

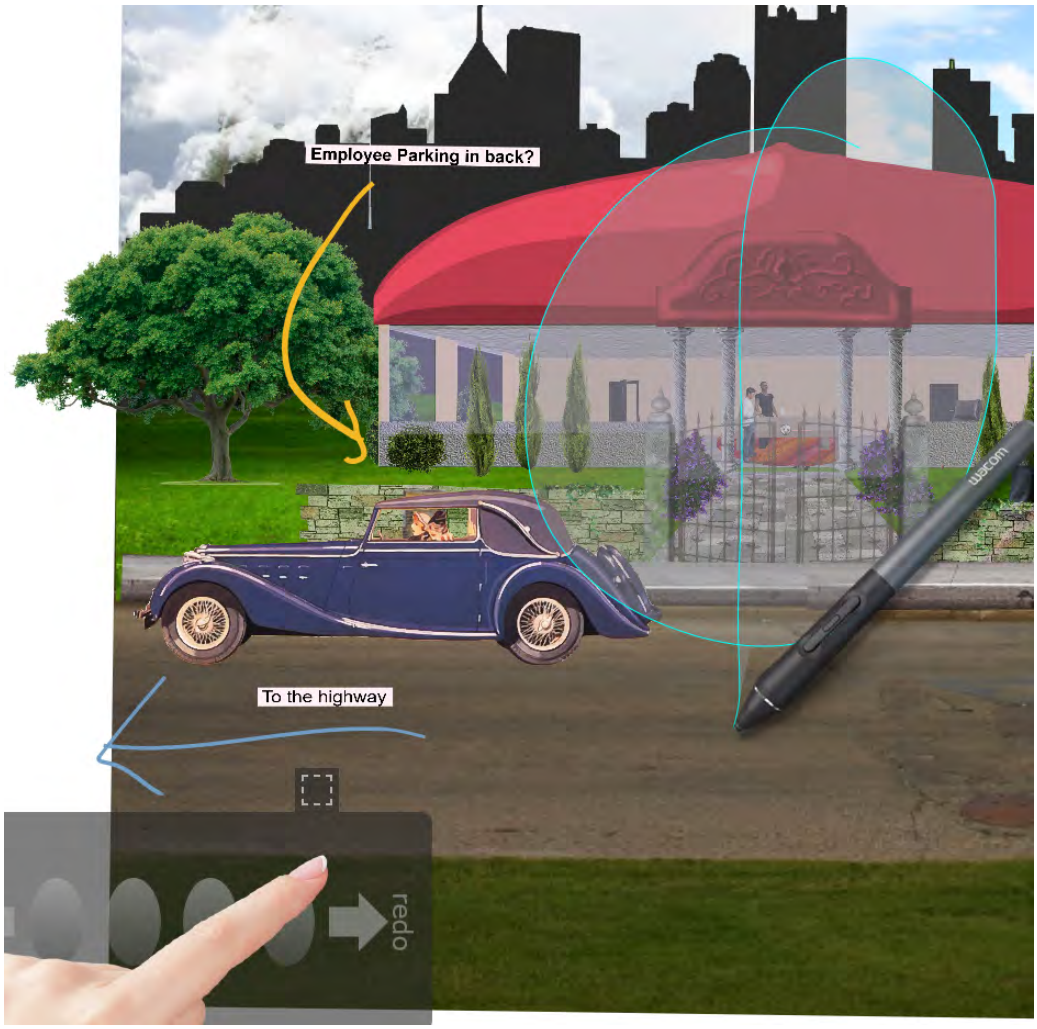


Figure 31: While holding a left-hand touch on the Hotpad, the user performs a Lasso'n'Cross gesture with a pen, selecting elements with a lasso, followed by a vertical slash to activate LayerFish.

ditional scene index that contains all elements, selection enables the user to work with a subset of elements, and so addresses scaling issues for human attention. To distinguish between normal selection for manipulating a group of elements versus selection for LayerFish, we use an activation gesture. LayerFish supports two bimanual input modalities for selection and activation, pen+touch and multi-touch.

#### V.2.1.1 Pen+Touch

When using the pen, LayerFish supports lasso selection of elements. LayerFish is activated with a Lasso'n'Cross gesture [Agarawala and Balakrishnan 2006]. A lasso selection with the pen is followed by a vertical slash across the center of the circled selection (see Figure 31). Lasso selection is performed using the selection quasimode, by placing one touch down on the Hotpad. After the user performs a Lasso'n'Cross gesture with a vertical slice, LayerFish's fisheye scene index, a transient interface, appears adjacent to the bounding box of the selection area.

#### V.2.1.2 Multi-touch

Touch input is less precise than pen due to the fat finger problem [Holz and Baudisch 2011]. We suspected this imprecision would make lasso selection involving dense overlapping content difficult with touch. We instead support rectangular selection with touch. Again, a left hand touch on the Hotpad defines right-hand actions as selection. With a right-hand touch, the user drags a rectangular selection area that defines what elements LayerFish will select. Once the desired selection area is defined, with the left hand touch still down, the user lifts up the right hand and makes a vertical slash through the rectangular selection. The slash activates LayerFish's fisheye scene index with the elements within the selection bounds.

### V.2.2 Fisheye Scene Index of Layered Elements

Upon activation, correspondents, representing the subset of selected elements, appear in an ordered list. Correspondents are ordered from top-most selected element to bottom-most.

Figure 32: Example of fisheye scene index in LayerFish. The focus of the fisheye is on the correspondent of the woman in a coat. Correspondents get smaller in size further from the focus.

#### V.2.2.1 Fisheye Visualization

A fisheye visualization is used to represent the list of correspondents (see Figure 32). Fisheyes are a focus+context technique that use spatial distortion to give larger visual emphasize to a focus while providing a smaller peripheral view of contextual details. In the case of LayerFish, the fisheye allows the user to see more correspondents at once than with a traditional scene index. The spatial distortion also makes the effect of scrolling non-

linear, enabling direct-touch scrolling over larger distances than with a traditional scene index.

The fisheye visualization defines a focus, initially the top correspondent in LayerFish. The focus is given a fixed size (120 x 80 pixels at 94 dpi). Correspondents get progressively smaller as you move away from the focus. Pressing down with pen or touch on a correspondent makes it the focus of the fisheye. Through pen or touch drags, the scene index is scrolled, keeping the focus positioned under the drag input.

LayerFish supports inertial scrolling, a common technique used in touch interfaces for browsing large lists. With a quick flick, the fisheye scrolls based upon the velocity and direction of the flick, decelerating with time. We employ a form of speed-coupled flattening during inertial scrolling to address target acquisition issues [Gutwin 2002]. That is, when the velocity of inertial scrolling exceeds a threshold, we flatten the fisheye scene index, making all correspondents equal in size. As the velocity falls back below the threshold, we redraw the fisheye with a focus on the top-most or bottom-most visible correspondent in the direction of the scroll.

### V.2.3 Layering Interactions

The user adjusts an element's position in the visual stack by first selecting and then dragging and dropping, with pen or touch, its correspondent up or down the scene index. A correspondent is selected by briefly pressing down on it. The selected correspondent enlarges slightly, giving the effect of popping out of LayerFish. A rectangular placeholder highlights the current position of a selected correspondent in the scene index (Figure 33). As the selected correspondent is dragged up or down, the placeholder moves up and down the list accordingly, swapping positions with correspondents. The fisheye redraws, maintaining focus on the placeholder. When the user drops the selected correspondent, it is placed back in the scene index in the position of the placeholder.

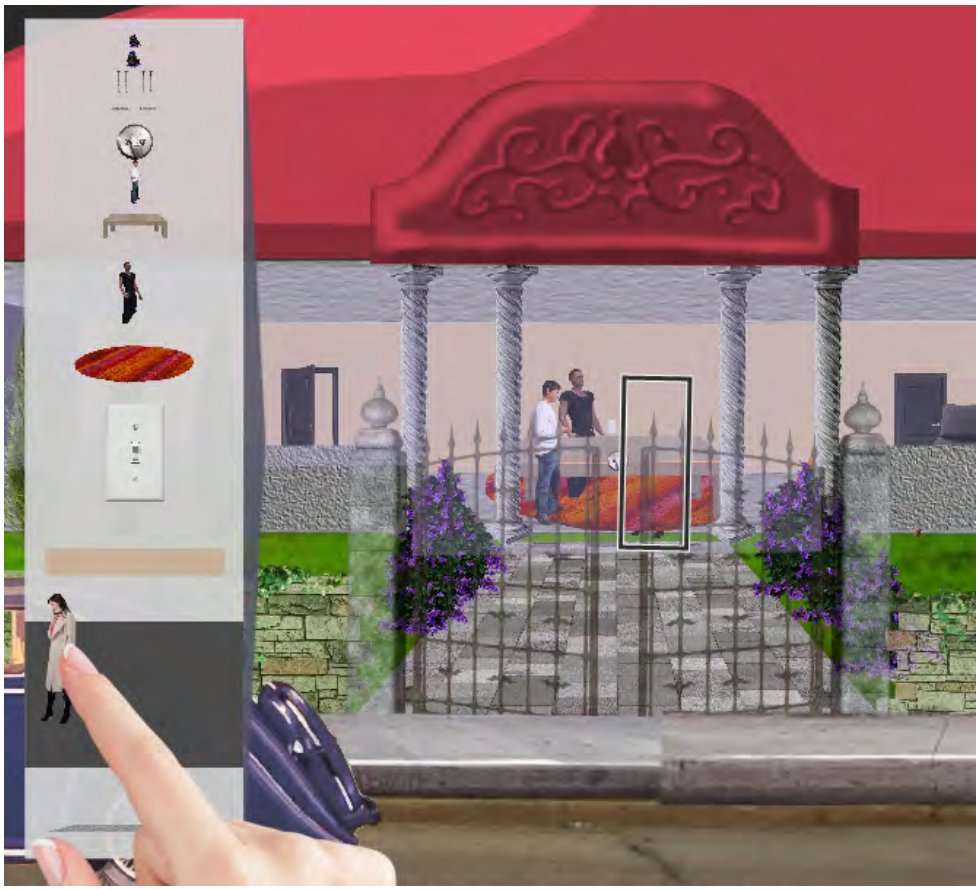

Figure 33: Example demonstrating selection of a correspondent thumbnail in LayerFish. The correspondent for the woman in a coat is selected. A gray rectangle indicates the current position of the selected correspondent in the scene index. A rectangular outline in the design space shows the location of the element represented by the selected correspondent, even though the element is hidden behind the beige wall (the correspondent directly above in the scene index).

#### V.2.3.1 Bimanual Scrolling

Drag and drop layering only supports layering above or below what is currently visible in the fisheye. As user the drags a selected correspondent, the focus changes, providing less space to visualize correspondents in the direction of the drag. To overcome this issue, the user can scroll the scene index without changing the focus, using a bimanual gesture. With a correspondent selected using the right hand, the user can drag the scene index using

the left hand, causing the fisheye to scroll. The selected correspondent remains fixed below the right hand, while its position in the scene index changes as correspondents above or below it are scrolled up or down.

The user can scroll the fisheye scene index at a faster rate by dragging up or down with a left hand touch, beyond the bounds of LayerFish. This initiates automatic scrolling. The further away from LayerFish that the user drags the left hand touch, the faster the automatic scrolling happens.

#### V.2.3.2 Bring to Top / Send to Bottom

LayerFish supports quickly moving an element to the top or bottom of the visual stack. With a correspondent selected, the user quickly drags it above LayerFish's fisheye scene index to bring the represented element to the top, while a quick drag below sends the element to the bottom. We use drag acceleration to identify how the user is moving the correspondent toward the top or bottom. If acceleration exceeds a threshold, then the drag movement is considered a bring to top or bottom action. The placeholder remains fixed until the acceleration decreases below the threshold. If this occurs when the user is above or below LayerFish, the correspondent and its represented element is moved to the top or bottom.

If the user stops dragging before reaching the top or bottom, then the action is considered a normal layering operation. The placeholder is repositioned under the dragged correspondent, and the fisheye is refocused and redrawn around the placeholder's position.

### V.2.4 Bimanual Interaction with Occluded Content

Occlusion issues make it difficult to access and manipulate overlapping content. The coarseness of touch input further compounds the problem. LayerFish addresses these issues through bimanual interaction, in which kinematic chains connect the scene index and

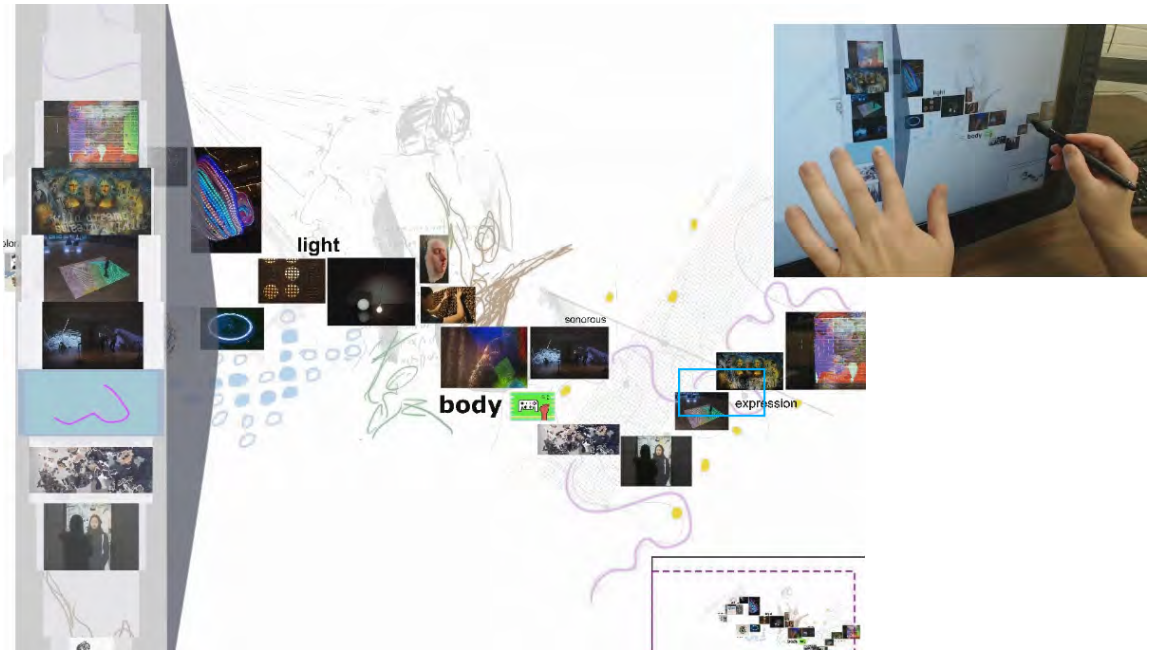


Figure 34: A user manipulates an occluded sketch element that is overlapped by images and text. A kinematic chain is formed by touching the element's correspondent representation in the fisheye scene index with the non-preferred hand, and repositioning the element in the design space with the pen in the right hand.

design space, to enable users to directly interact with specific elements.

The user activates elements for manipulation by touching their correspondent in the scene index with the left hand. Manipulation with pen or touch by the right hand in the design space is selectively limited to left hand activated elements (see Figure 34). Selected elements are visually outlined, in the design space, to indicate that only they afford manipulation. When the left hand lifts touches from the scene index correspondents, the outlines disappear in the design space. The user again can now interact with all elements.

### V.2.5 Visibility

The visibility in the design space of elements and groups can be hidden by tapping correspondents in the scene index with the eraser end of the pen. They remain in the scene index, faded out to indicate hidden status. Tapping a hidden element in the scene index

with the eraser makes it visible again.

## V.3 Evaluation Design

To evaluate the effectiveness of LayerFish for adjusting the visual stacking order of overlapping content, we designed a repeated measures (within-subjects) evaluation comparing LayerFish with a traditional scene index, which we call *Sidebar*. We compare differences in task time between LayerFish and Sidebar. We vary several independent variables, including the number of elements and overlap density, to investigate in which situations LayerFish performs better than Sidebar.

### V.3.1 Apparatus

The study apparatus consisted of a pre-questionnaire, two tutorial videos, 12 training tasks, and 48 study tasks. Participants began by answering the pre-questionnaire, which collected demographic information and details about their experiences using visual design tools and layering. Then, a tutorial video explained the task that participants would perform and demonstrated how to use either LayerFish or Sidebar. Next, participants performed 6 training tasks, followed by 24 study tasks using either LayerFish or Sidebar, depending on which tutorial video was shown. After completing the tasks, a second tutorial video demonstrated how to use the other technique, either LayerFish or Sidebar, that the participant had not yet used. Again, the participants performed 6 training tasks, followed by 24 study tasks. The study concluded with a post-questionnaire. Study sessions lasted approximately 30 minutes. Participants were given a $15 Amazon gift card.

#### V.3.1.1 Task

Each task consisted of a visual arrangement of overlapping elements (see Figures 36 and 37). Each element was a dancing figure from the artworks of Keith Haring [Deitch et al. 2014], providing playfully engaging visuals with low complexity. The number of

elements and how much they overlap was specified by independent variables (see *Independent Variables* section). The objective of each task was to layer a *task element* so that it resides between two goal elements in the z-dimension. The task and goal elements were uniquely colored with special fill patterns to make them easily distinguishable from the other elements (see Figure 35). The two goal elements always resided near the top of the visual stack, while the goal element resided near the middle or bottom. Performing a task consisted of three stages: (1) accessing the layering technique; (2) finding the task element; and (3) layering the task element between the two goal elements.

Figure 35: Yellow-gridded task element in the center, and two goal state elements (blue-dotted and red-striped) on left and right.

#### V.3.1.2 Sidebar

Sidebar represents a typical scene index used in visual design tools. In the study, Sidebar appears on the preferred hand side of the screen and contains correspondents for all elements (see Figure 36). Using a single touch drag, Sidebar is scrolled. The participant adjusts the layering of an element by dragging out the correspondent horizontally. The correspondent is removed from the scene index and appears under the participants dragging touch. Dragging the correspondent above or below the presented scene index will cause scroll the scene index in the dragged direction, beyond the initially visible subset of elements. Dropping the dragged correspondent back onto the scene index enables the

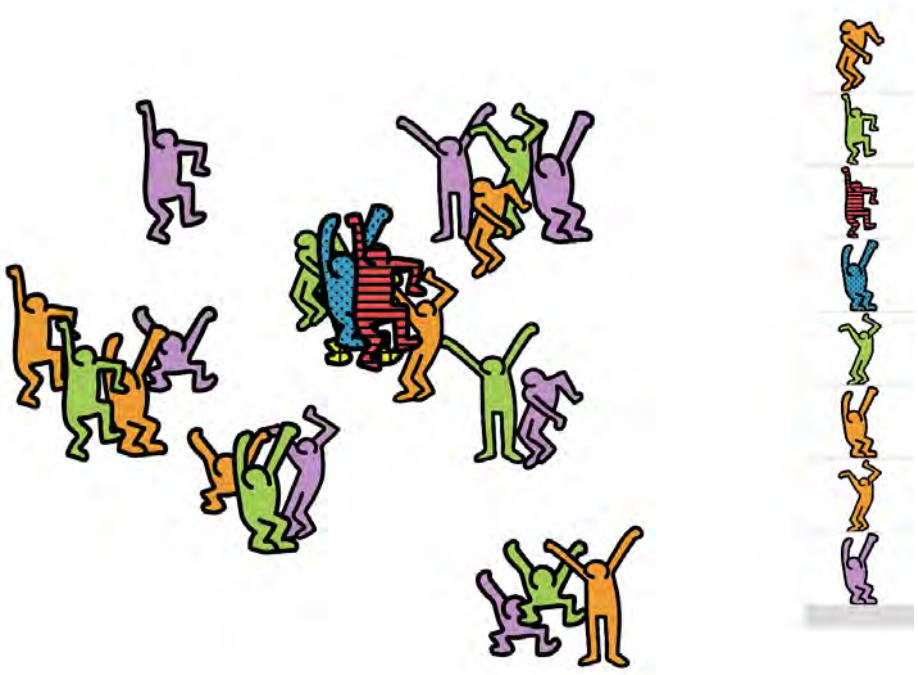

Figure 36: Study task example with Sidebar and Low Overlap Density.

participant to layer an element higher or lower in the visual stack. Dropping the correspondent outside the scene index cancels the operation, returning the correspondent to its original position.

#### V.3.1.3 Hardware Setup

The study workstation consisted of a Windows 8.1 PC with an Intel i7-5960X processor, 16GB RAM, and an NVIDIA GeForce GTX 680 Ti graphics card. Pen and touch input was supported through a Wacom Cintiq 24HD Touch display (1920 x 1200 pixels). Two cameras, one overhead and one on the right, captured participants' gestural interactions.

### V.3.2 Independent Variables

Tasks involved 4 independent variables: Technique, Overlap Density, Number of Elements, and Layering Distance.

#### V.3.2.1 Technique

Technique was either LayerFish or Sidebar.

#### V.3.2.2 Overlap Density

Overlap Density is either *Low* or *High*. For Low Overlap Density, the task and goal elements only overlap with a few other elements (see Figure 37). This represents common layering tasks, such as when designing slides for a PowerPoint presentation. For High Overlap Density, all elements overlap. This represents the worst possible case for layering tasks.

High Overlap Density focuses investigation on how LayerFish's fisheye visualization compares with a traditional scene index when the numbers of correspondents are the same. In Low Overlap Density tasks, LayerFish can be used to select a subset of elements. In High Overlap Density tasks, all elements overlap, so the user is unable to select a subset using LayerFish. As a result, both techniques present an equal number of correspondents, where the primary difference is how the correspondents are presented, either with fisheye or in a traditional scene index.

#### V.3.2.3 Number of Elements

The Number of Elements is either *25* or *100*. Through informal needs and requirements gathering sessions, Landscape Architecture students reported often creating design representations with around 100 layers. In the most extreme cases, all these layers overlapped. Thus, the condition with 100 elements and High overlap, represents a worst possible case scenario. While, 25 elements and High overlap represents the more common problematic scenario. One of our principle hypotheses is that LayerFish will scale better than Sidebar, as the number of elements increases.

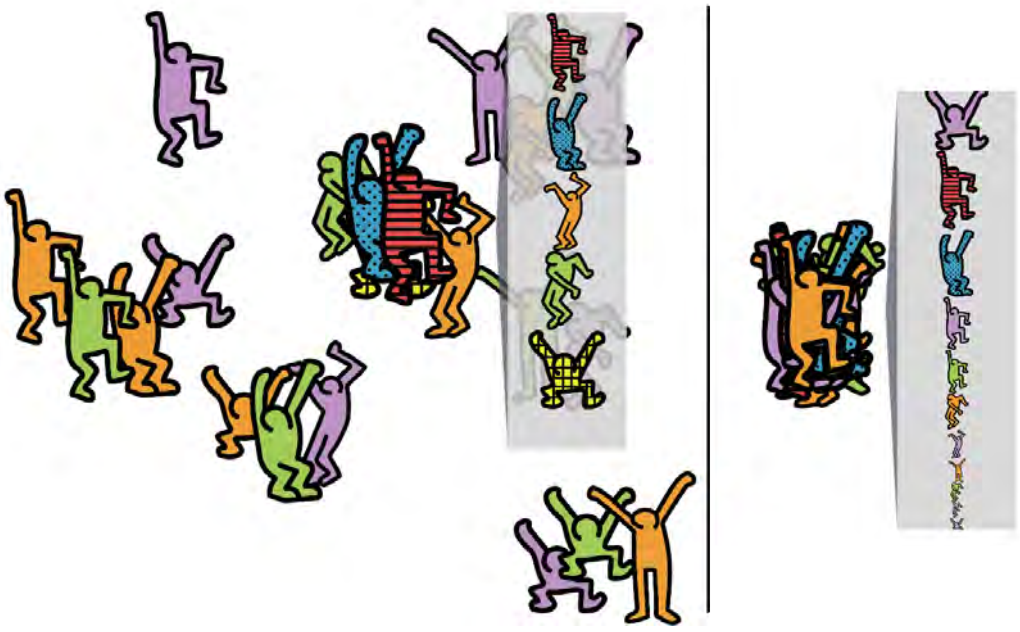

Figure 37: Low Overlap vs. High Overlap Density with LayerFish.

#### V.3.2.4 Layering Distance (Relative)

Relative Layering Distance is the number of elements between the task element and the two goal elements at the start of the task. We use two possibilities for Layering Distance, *Medium* and *Far*. The absolute magnitude of these distances, in elements, varies directly with the Number of Elements. When Number of Elements is *25*, *Medium* is 7 and *Far* is 19. When Number of Elements is *100*, *Medium* is 44 and *Far* is 94. We vary Layering Distances so that participants do not know where the task element is located in each trial. Using two fixed distances for each Number of Elements reduces variability, and so facilitates comparison across techniques.

#### V.3.2.5 Tasks: Independent Variable Conditions

The 4 independent variables, each with 2 levels, produce 16 different conditions. Participants performed a total of 60 tasks, of which 12 were training and 48 were timed

trials. For the timed trials, each condition was performed by each participant 3 times ($16 \times 3 = 48$).

### V.3.3 Dependent Variables: Time Metrics

Layering tasks with scene indexes involve first finding a desired correspondent, and then adjusting that correspondent's position in the scene index. We wondered how a fish-eye scene index would compare with a traditional scene index for this components of a layering task. Thus, we derive three time metrics for analyzing participants performance: Total Task Time, Find Time, and Layering Time. *Total Task Time* measures how long it took participants to perform the task in entirety. *Find Time* measures how long it took participants to find the task element that must be layered. Find Time is calculated as the time between the first touch down on the scene index (to scroll) to the first touch down on the task element. *Layering Time* measures how long it took participants upon finding the task element to reorder it to the goal state. Layering Time is calculated as the time after selection until the task element is positioned between the two goal elements.

We note that Find Time + Layering Time $\neq$ Total Task Time. Total Task Time includes additional time for selecting and activating LayerFish and for accessing Sidebar. We do not include a separate comparison of this additional time since LayerFish is expected to always be slower, because selection and activation requires a longer input sequence than the single touch required for accessing Sidebar.

## V.4 Participants and Results

We recruited 47 participants (12 female) between 19 and 33 years old (22 mean). Participants were university students, primarily from Computer Science. We specifically invited participants from design-centric courses in Computer Science and Landscape Architecture to recruit participants with extensive experience working with layers. A majority (37) had worked with design tools at least once. Before investigating with expert visual

designers, we sought to evaluate LayerFish with a more accessible population, as an initial usability study. Future work will examine expert visual designers.

We conducted a 2 (Technique) x 2 (Overlap Density) x 2 (Number of Elements) x 2 (Layering Distance) x 3 (trials) repeated measures analysis of variance (RM-ANOVA) on Total Task Time, Find Time, and Layering Time metrics for *All Participants*, the set of 47 participants. We performed pairwise comparison for each condition (see Table 5). Further, we examine a subset of All Participants identified as Experienced Layering Participants. We observe no order effects. Since a task was only completed when the task element was layered between the two goal elements, there is no measure of errors. A post-questionnaire provides subjective experience report data.

### V.4.1 Experienced Layering Participants

Through observations of participants during study sessions and variances in the data, we suspected that the participants more experienced with layering would performed tasks faster. From the All Participants set, we identify 12 Experienced Layering Participants based upon responses to two questions in the pre-questionnaires:

- How often do you use visual design tools?
- On average, how many layers do your projects contain?

These 12 participants responded that they often or frequently use visual design tools and typically work with at least 10 layers. The other 37 participants rarely used visual design tools, and on average, worked with 5 or less layers. For each time metric, we specifically compare times for these experienced users between the two techniques.

### V.4.2 Total Task Time: All Participants

Total Task Time begins when a participant first touches and ends when the task is completed correctly. For LayerFish, this includes selection and activation time. Total Task

| | | | | *Total Task Time (s)* | | | *Find Time (s)* | | | *Layering Time (s)* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Overlap Density** | **Num Elements** | **Layer Distance** | **Layer Fish** | **Side-bar** | $\frac{p <}{F =}$ | **Layer Fish** | **Side-bar** | $\frac{p <}{F =}$ | **Layer Fish** | **Side-bar** | $\frac{p <}{F =}$ |
| A1 | Low | 25 | Medium | 4.6 | 6.5 | $\frac{.001}{98.5}$ | .036 | 2.3 | $\frac{.001}{810.8}$ | 1.61 | 4.22 | $\frac{.001}{145.7}$ |
| E1 | Low | 25 | Medium | 4.01 | 6.35 | $\frac{.001}{25.44}$ | 0 | 2.1 | $\frac{.001}{187.3}$ | 1.08 | 4.00 | $\frac{.001}{60.62}$ |
| A2 | Low | 25 | Far | 4.2 | 7.2 | $\frac{.001}{53.7}$ | .081 | 2.03 | $\frac{.001}{1036}$ | 1.48 | 5.21 | $\frac{.001}{189.8}$ |
| E2 | Low | 25 | Far | 4.12 | 7.18 | $\frac{.001}{28.42}$ | 0 | 2.09 | $\frac{.001}{155.6}$ | 1.14 | 5.10 | $\frac{.001}{69.75}$ |
| A3 | Low | 100 | Medium | 4.1 | 14.3 | $\frac{.001}{283}$ | .03 | 6.5 | $\frac{.001}{397.1}$ | 1.43 | 7.79 | $\frac{.001}{163.7}$ |
| E3 | Low | 100 | Medium | 3.84 | 14.39 | $\frac{.001}{71.66}$ | .03 | 5.43 | $\frac{.001}{201.8}$ | 1.03 | 8.03 | $\frac{.001}{59.55}$ |
| A4 | Low | 100 | Far | 5.5 | 19.7 | $\frac{.001}{797}$ | .10 | 7.98 | $\frac{.001}{1858}$ | 2.23 | 11.75 | $\frac{.001}{556.1}$ |
| E4 | Low | 100 | Far | 4.80 | 19.16 | $\frac{.001}{388.8}$ | .07 | 7.38 | $\frac{.001}{437.6}$ | 1.24 | 11.36 | $\frac{.001}{414.7}$ |
| A5 | High | 25 | Medium | 8.7 | 6.6 | $\frac{.001}{50.2}$ | 2.33 | 2.34 | $\frac{.97}{.002}$ | 3.77 | 4.25 | $\frac{.20}{5.07}$ |
| E5 | High | 25 | Medium | 7.6 | 6.54 | $\frac{.10}{3.25}$ | 2.2 | 2.35 | $\frac{.65}{.212}$ | 3.22 | 4.19 | $\frac{.05}{3.63}$ |
| A6 | High | 25 | Far | 9.6 | 7.8 | $\frac{.01}{7.51}$ | 1.8 | 2.0 | $\frac{.02}{5.67}$ | 5.10 | 5.83 | $\frac{.11}{2.59}$ |
| E6 | High | 25 | Far | 9.28 | 7.23 | $\frac{.001}{25.44}$ | 1.9 | 1.9 | $\frac{.83}{.05}$ | 3.97 | 5.34 | $\frac{.047}{4.15}$ |
| A7 | High | 100 | Medium | 16.9 | 12.9 | $\frac{.001}{20.2}$ | 5.53 | 5.71 | $\frac{.54}{.39}$ | 8.41 | 7.16 | $\frac{.24}{1.44}$ |
| E7 | High | 100 | Medium | 15.6 | 13.17 | $\frac{.05}{4.22}$ | 4.6 | 5.9 | $\frac{.10}{2.49}$ | 7.12 | 7.5 | $\frac{.70}{.209}$ |
| A8 | High | 100 | Far | 23.2 | 19.9 | $\frac{.001}{23.3}$ | 7.46 | 8.22 | $\frac{.03}{4.1}$ | 13.39 | 11.64 | $\frac{.044}{4.10}$ |
| E8 | High | 100 | Far | 21.39 | 20.0 | $\frac{.50}{.477}$ | 5.86 | 8.25 | $\frac{.017}{6.18}$ | 12. 69 | 11.78 | $\frac{.61}{.268}$ |

Table 5: Pairwise comparison of Total Task Time, Find Time, and Layering Time between the two Techniques for each combination of Overlap Density, Number of Elements, Layering Distance. Rows A1-A8 (white background) show mean times with statistical significance for All Participants. Rows E1-E8 (gray background) show mean times with statistical significance for Experienced Layering Participants, a subset of All Participants.

Time using LayerFish was less than Sidebar with Low Overlap Density, but greater than with High Overlap Density (see Figure 38). Differences in means were statistically significant in all conditions (Table 5, A1-8). We observed a main effect for each independent variable: Technique ($F_{1,34} = 95.862$, $p < 0.001$), Overlap Density ($F_{1,34} = 918.01$,

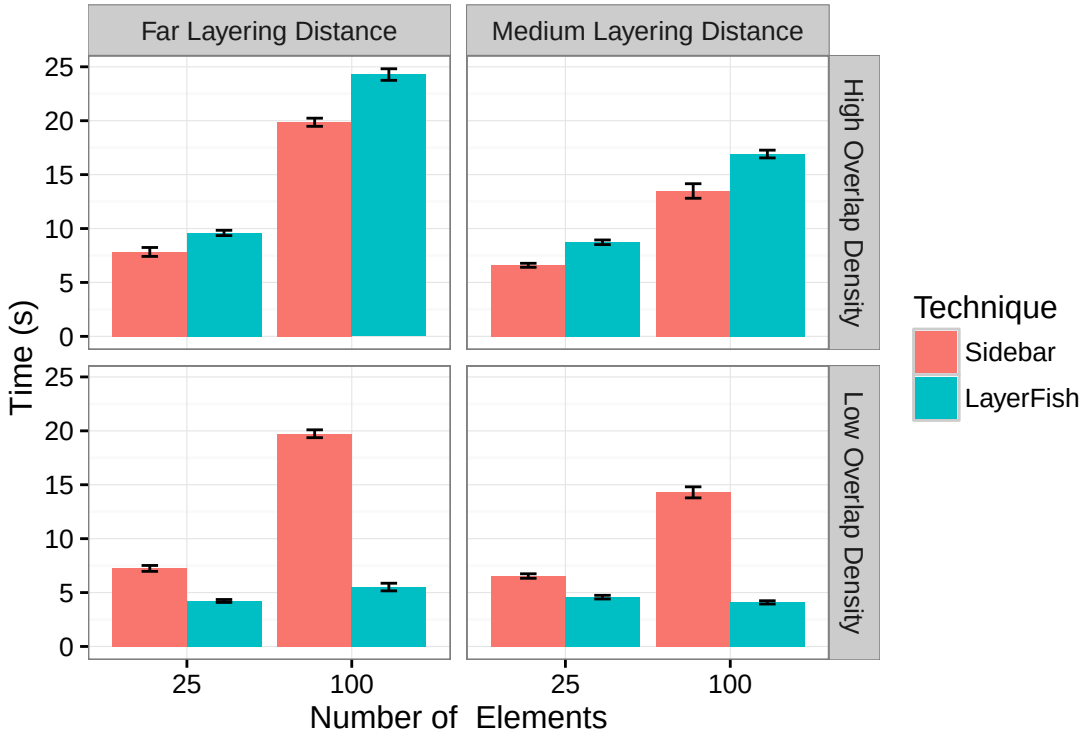


Figure 38: Mean Total Task Time for All Participants.

$p < 0.001$), Number of Elements ($F_{1,34} = 1938.316$, $p < 0.001$), and Layering Distance ($F_{1,34} = 154.541$, $p < 0.001$).

### V.4.3 Total Task Time: Experienced Layer Participants

Experienced Layering Participants were on average faster for Total Task Time when compared with All Participants, particularly for LayerFish. In Low Overlap tasks, these participants were faster with LayerFish (see Figure 39). In High Overlap tasks, these participants were faster with Sidebar, but only some cases were statistically significant (Table 5, E6 and E7).

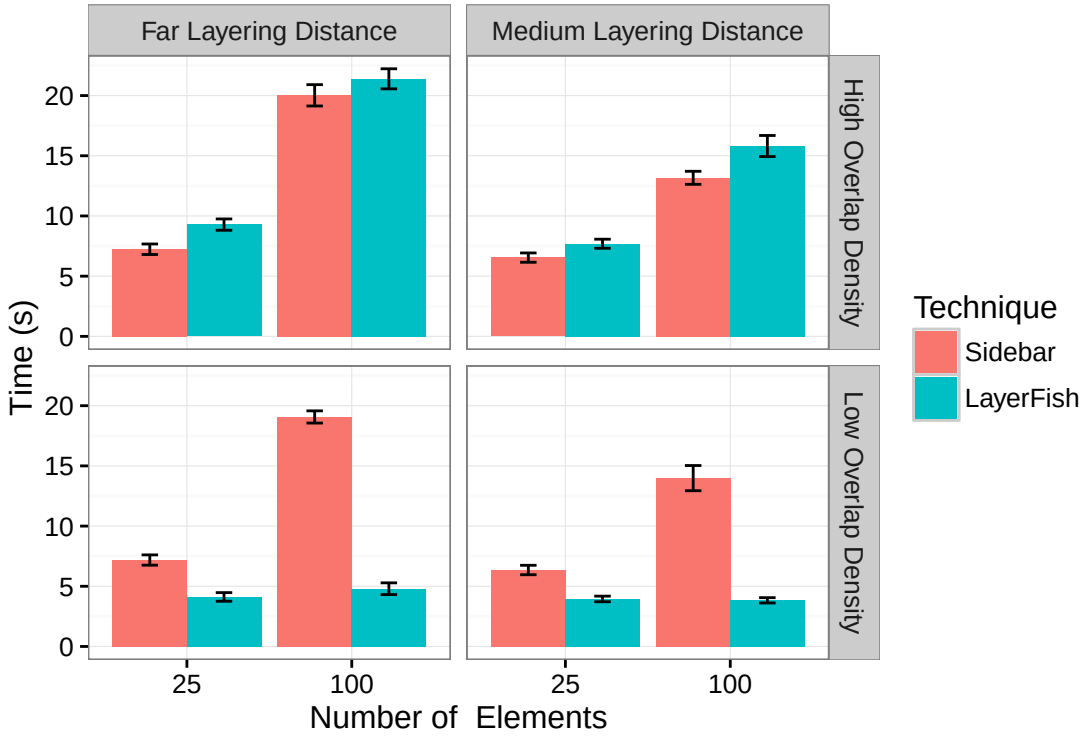


Figure 39: Mean Total Task Time for Experienced Layering Participants.

### V.4.4 Find Time: All Participants

In Low Overlap Density tasks, participants were faster at finding the task element with LayerFish (Figure 40, Table 5 A1-4). In High Overlap Density tasks, participants averaged faster Find Time with LayerFish in most conditions. These results were statistically significant when Layering Distance is Far (Table 5, A6 and A8). We observed a main effect for all factors: Technique ($F_{1,34} = 1412.692$, $p < 0.001$), Overlap Density ($F_{1,34} = 416.6$, $p < 0.001$), Number of Elements ($F_{1,34} = 1175.654$, $p < 0.001$), and Layering Distance ($F_{1,34} = 31.753$, $p < 0.001$).

In Low Overlap Density tasks, Find Time for LayerFish can be zero seconds (Table 5, A1-4, E1-4). Find Time is intended to measure the scrolling time required to find the task element. Find Time is measured as the time between the first touch down within a

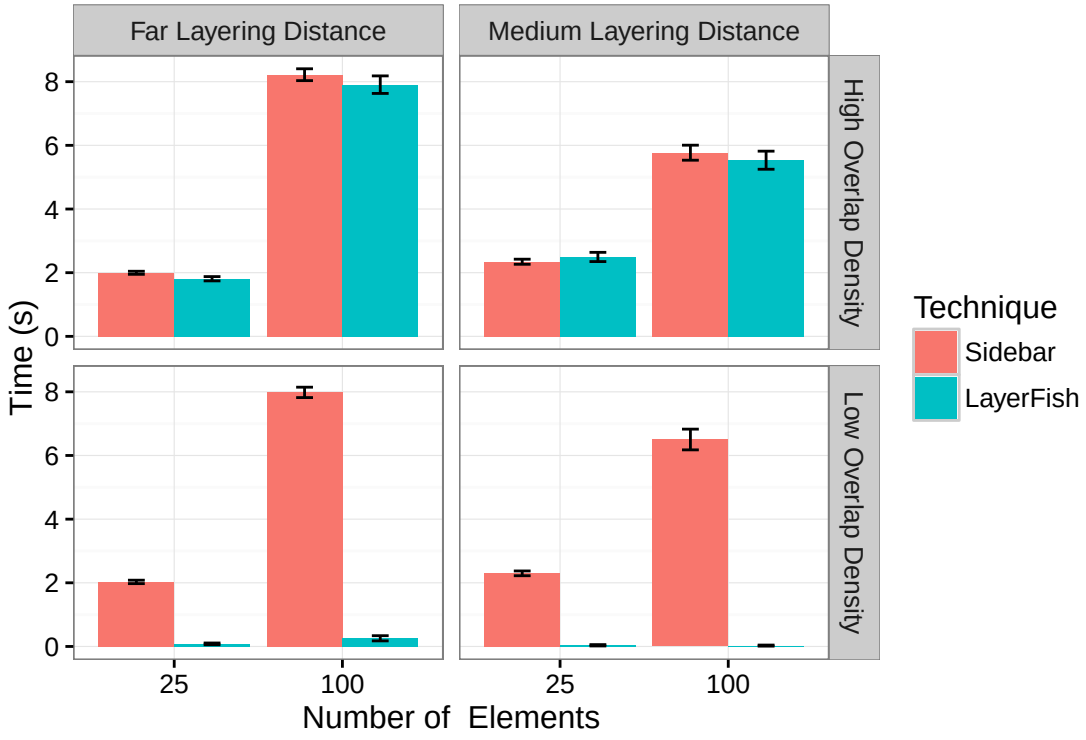


Figure 40: Mean Find Time for All Participants.

technique and the first touch down on the task element. With LayerFish, these two touches may be the same touch, as selection allows the participant to see the task element without scrolling.

### V.4.5 Find Time: Experienced Layering Participants

The Experienced Layering Participants were faster at finding the task element on average using LayerFish than Sidebar (see Figure 41). This result was statistically significant for all Low Overlap Density tasks and when Number of Elements was 100 and Layering Distance was Bottom for High Overlap Density tasks (Table 5, E1-4, E8). These participants also were on average faster at finding the task element than the average for All Participants.

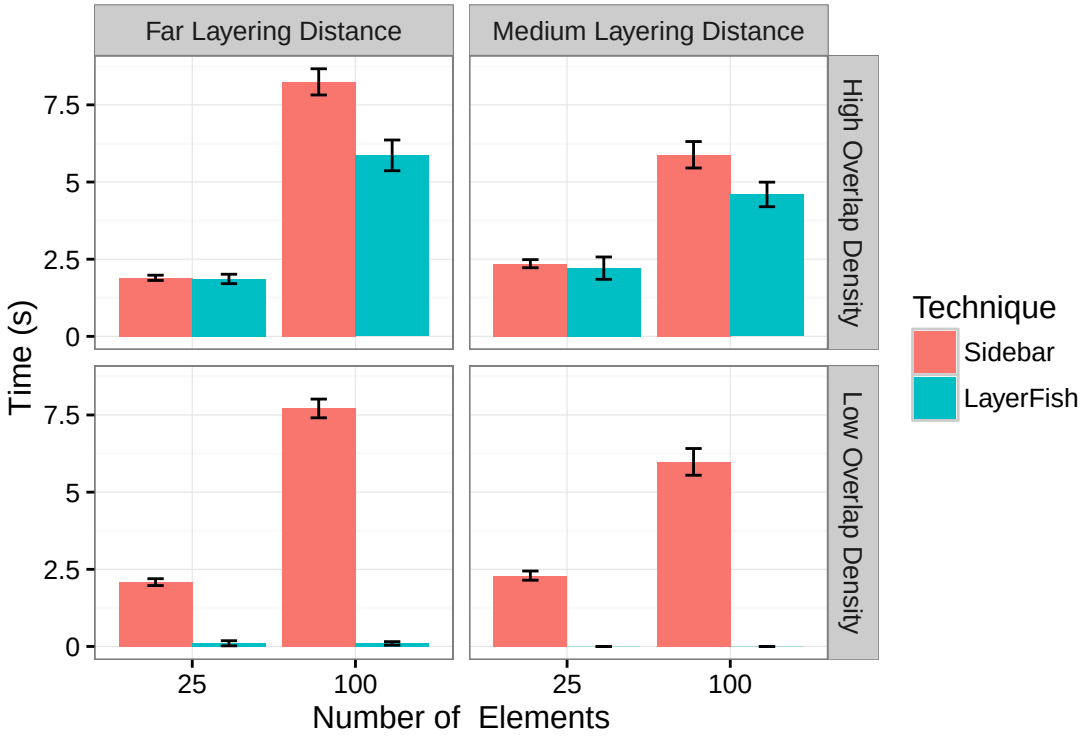


Figure 41: Mean Find Time for Experienced Layering Participants.

### V.4.6 Layering Time: All Participants

Layering Time measures long it takes to adjust the visual stack position of the task element so that it is between the two goal elements. The metric is derived by calculating the difference in time between the selection of the task element and the finishing of the task. In Low Overlap Density tasks, participants had faster Layering Times when using LayerFish (see Figure 42). This result is statistically significant for all Low Overlap Density tasks (Table 5, A1-4). In High Overlap Density tasks, participants were faster with LayerFish when the Number of Elements was 25, but slower with LayerFish when the Number of Elements was 100. This result was statistically significant only in tasks with 100 elements and the Layering Distance is Far (Table 5, A8). We observed a main effect for all factors: Technique ($F_{1,34} = 247.403$, $p < 0.001$), Overlap Density ($F_{1,34} = 471.168$,

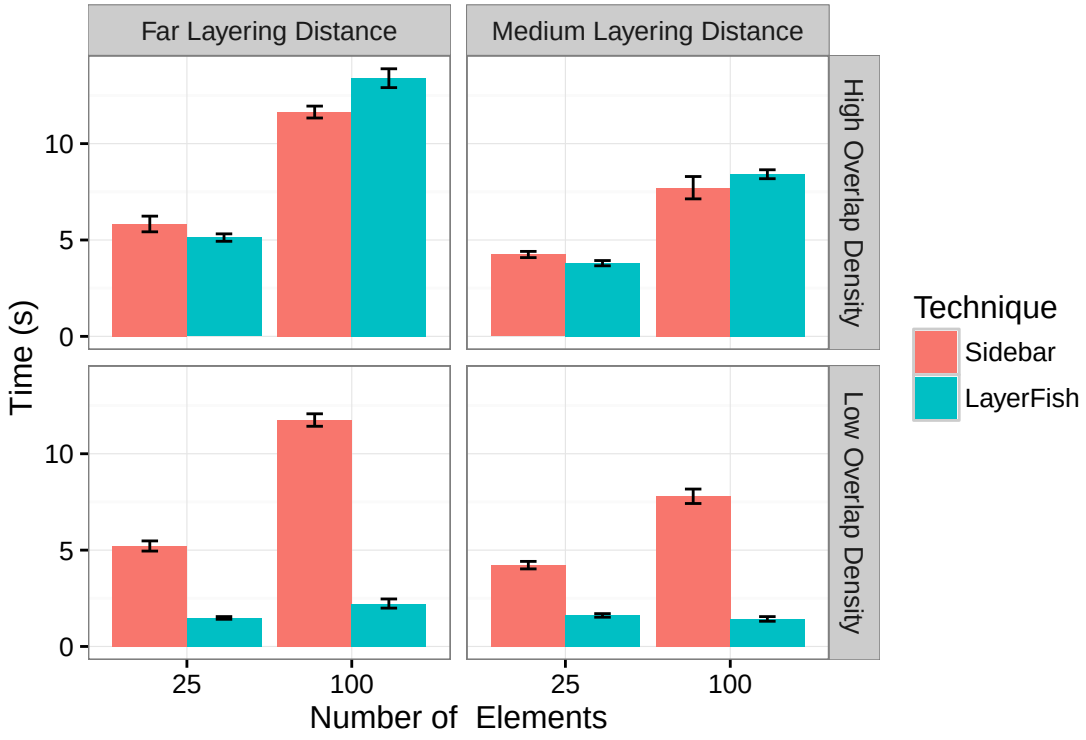


Figure 42: Mean Layering Time for All Participants.

$p < 0.001$), Number of Elements ($F_{1,34} = 1105.93$, $p < 0.001$), and Layering Distance ($F_{1,34} = 140.914$, $p < 0.001$).

### V.4.7 Layering Time: Experienced Layering Participants

Experienced Layering Participants had faster Layering Times for Low Overlap Density tasks when using LayerFish (Figure 43). These results were statistically significant (Table 5, A1-4). In High Overlap Density tasks, these participants were faster with LayerFish in all cases, except when the Number of Elements was 100 and the Layering Distance was Far (E8). The results were statistically significant only when the Number of Elements was 25 (E5-6).

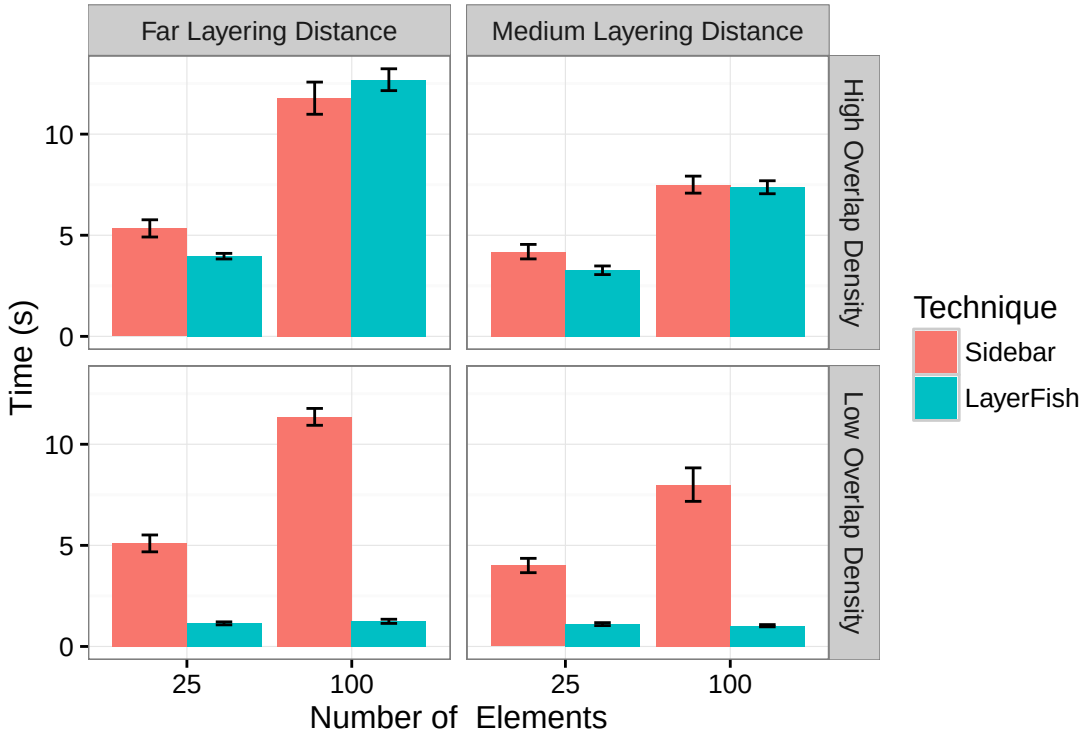


Figure 43: Mean Layering Time for Experienced Layering Participants.

### V.4.8 Experience Reports

Post-questionnaire responses provide reports of subjective experiences (see Figure 44). A majority of participants agreed that LayerFish was easy to use ($\chi^2 = 30.13$, $p < 0.001$), and they felt faster with LayerFish for both a few elements ($\chi^2 = 21.62$, $p < 0.003$) and many elements ($\chi^2 = 10.55$, $p < 0.05$). Participants agreed that LayerFish required less effort with only a few elements ($\chi^2 = 9.49$, $p < 0.05$). However, with many elements, only a slight majority agreed that LayerFish required less effort. This result was not statistically significant ($\chi^2 = 6.94$, $p < 0.14$). Participants agreed that they would use LayerFish in visual design tools like Adobe Photoshop ($\chi^2 = 20.55$, $p < 0.004$).

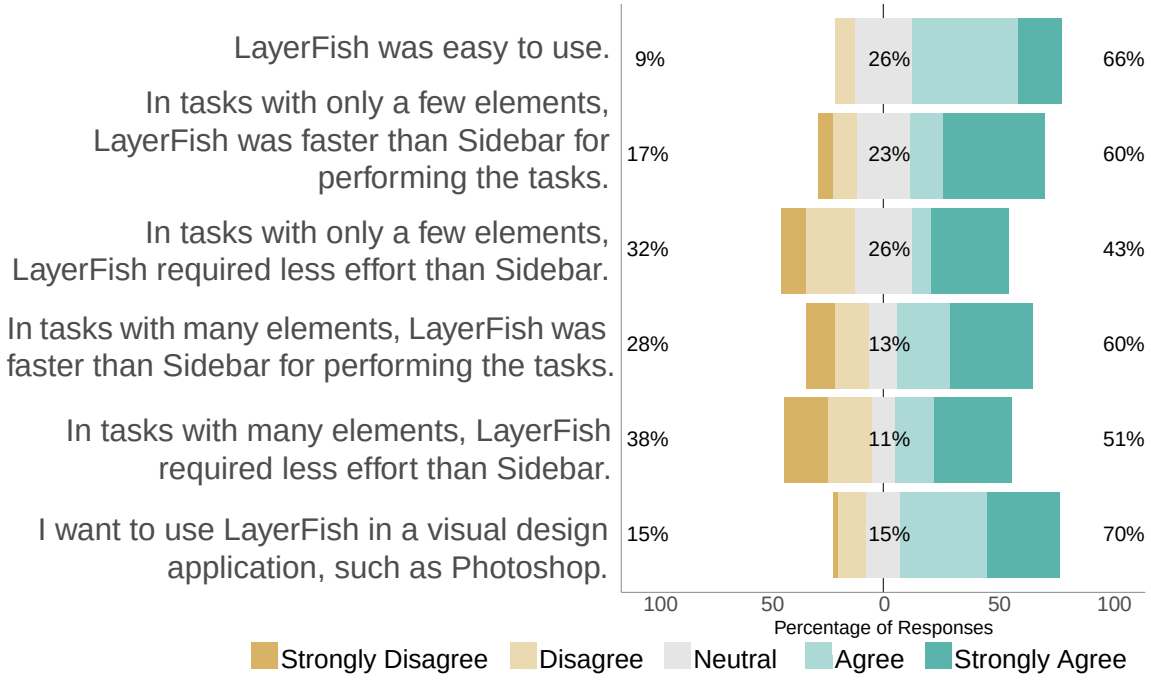


Figure 44: Post-questionnaire responses. Percentages show breakdowns participants' agreement (right), neutrality (middle), or disagreement (left) for each statement.

### V.5 Discussion

We hypothesized that our independent variables, which differentiate situations that users are likely to experience when working with overlapping content, would yield interesting findings. We observe significant differences between the two techniques for Overlap Density, Number of Elements, and Layering Distance. The most significant differences occur with Overlap Density. LayerFish was faster than Sidebar for Low Overlap tasks and slower in High Overlap tasks. We suspect that LayerFish requires additional learning time, as participants gain understanding of how interactions affect the fisheye representation. We also observed task performance differences for Experienced Layer Participants that indicate benefits of LayerFish for more expert users.

### V.5.1 Low Overlap Density

In Low Overlap Density tasks, mean Total Task Time, Find Time, and Layering Time with LayerFish were all shorter than with Sidebar. Low Overlap Density means that the task element and goal elements only overlap with a few other elements. Selection in LayerFish allows the participant to engage only those few elements that overlap, unlike with Sidebar, where all elements are present in the scene index. Not surprisingly, this capability improves layering performance. After selection with LayerFish, the task element was immediately visible and could be quickly layered between the two goal elements, which were also visible. LayerFish does not require scrolling in this case, while Sidebar does.

LayerFish does not present a fisheye visualization in these low overlap tasks, instead using the common scene index representation. Ramos et al. suggested (but did not investigate) that an in-place scene index palette would improve layering times [2006]. We have shown that this is true with at least 25 elements. We suspect that with fewer elements, the performance of a global scene index, like Sidebar, will be comparable to a contextual technique, like LayerFish. There exists a threshold in the number of elements, below which the selection and activation time for a contextual technique exceeds the time to move to the side to access a global technique.

### V.5.2 High Overlap Density

We sought to understand the effects of the fisheye visualization on layering performance, by including a task condition, High Overlap, in which both LayerFish and Sidebar include a similar number of elements.

The Total Task Time results show that LayerFish is slower than Sidebar in high overlap tasks. LayerFish requires a selection gesture to activate. High Overlap tasks require that participants select (nearly) all elements using LayerFish. In this case, the selection takes

time but doesn't help the user. When we further breakdown task time, for Find Time, we find that LayerFish is faster than Sidebar when there are 100 elements and Layering Distance is Far (Table 5, A8, E8). We suspect the fisheye's spatial distortion allows participants to scroll longer distances in a single touch gesture. As Layering Distance increases, so too does the difference in Find Time between LayerFish and Sidebar.

Meanwhile, for Layering Time with All Participants, LayerFish was slower when Number of Elements was 100 and Layering Distance was Far (Table 5, A8). When we look just at Experienced Layering Participants, LayerFish was faster when the Number of Elements was 25 (E5-6). We observe a larger variance in Layering Times when Layering Distance is Far, particularly for LayerFish. Bimanual scrolling was new to participants. We suspect that it may require more training for users to become comfortable at using it effectively.

### V.5.3 Experienced Layer Participants

We observe a difference in the performance of the Experienced Layering Participants. Using LayerFish, these participants were faster than the average for All Participants. Yet, while using Sidebar, they performed, on average, similarly to All Participants. These performance differences for LayerFish are most evident in the Find Time metric with High Overlap tasks, particularly with 100 elements. In these cases, Experienced Layering Participants' mean Find Time is almost two seconds faster than the mean time for All Participants. The faster performance exhibited by Experienced Layering Participants with LayerFish indicates that LayerFish has performance benefits for visual designers and others who work with many layers of overlapping elements. These participants seem more adept at learning to use LayerFish effectively. Novice layer users may also see improved performance with greater training and experience with layering.

The other significant difference was in Layering Time for High Overlap tasks with 25

elements. In these tasks, Experienced Layering Participants were faster using LayerFish than Sidebar. This matches the observed results for All Participants, except that, unlike for all, the differences for Experienced Layering Participants are statistically significant. All participants used bimanual scrolling in LayerFish to complete tasks with dense overlap. Again, bimanual scrolling is a new interaction, different from what participants are use to doing with touch devices. These results indicate that Experienced Layering Participants were among the quickest to learn how to effectively use the technique, emphasizing the benefits of LayerFish for visual designers who work with many layers.

### V.5.4 Participant Experiences

Participants reported positive experiences using LayerFish. Interestingly, despite that participants were actually slower (in Total Task Time) using LayerFish with many elements, many participants felt that they were faster. When we exclude selection time for LayerFish, most participants were faster using LayerFish. This may have been part of participants' consideration when responding. Participants were not in agreement that LayerFish requires less effort than Sidebar when working with many elements. Since LayerFish requires additional touches for selection and bimanual scrolling, it is not surprising that participants felt it didn't require less effort.

## V.6 Implications for Design

We use the findings from the evaluation to derive implications for the design of pen+touch and multi-touch design tools that support layering operations.

Design tools with a scene index would benefit from transitioning to a fisheye visualization as the number of elements in the design space grows large. We saw benefits with LayerFish when the design space contained at least 25 elements, even when all elements overlapped. LayerFish was faster than Sidebar for Find Time in High Overlap tasks when Layering Distance was Far. We suspect that the spatial distortion of the fisheye allowed

users to scroll longer distances more quickly. Prior work has used semantic zooming [Perlin and Fox 1993] as a form of spatial scaling to support speed-dependent scrolling rates in document browsing tasks [Igarashi and Hinckley 2000]. We have shown that a fisheye view, as an alternate form of spatial scaling, can also improve scrolling performance in layering tasks with a scene index.

Design tools should support both in-place (i.e. LayerFish) and always-present (i.e. Sidebar) techniques for layering overlapping content. As the number of elements in a design grows large, the traditional scene index requires more effort and attention to effectively layer elements. An in-place technique, which enables selection to filter elements the user wishes to layer, can improve layering task performance time. However, when there are many elements that highly overlap, the user is unable to effectively select only the desired elements, reducing the benefits of selection. Additionally, when there are only a few elements, all are directly visible in the scene index, and scrolling is not required. In these cases, we recommend a traditional scene index that transitions to a fisheye when the number of elements grows large (at least 25).

Multi-touch scene indexes should support bimanual scrolling, particularly for tools in which users typically create 10 or more layers. Bimanual interactions have been shown to improve performance of compound tasks, in which each hand can adjust different parameters of interaction [Kabbash et al. 1994]. In LayerFish, the right hand layers a selected correspondent, while the left hand scrolls the fisheye scene index. Unimanual approaches require dragging the selected correspondent to the top or bottom of the scene index, requiring more total touch distance travel than with our bimanual technique.

## V.7 Conclusion

We present LayerFish, a new bimanual interaction technique for layering overlapping content. With a bimanual gesture, the user selects a subset of elements within a larger

design space. When filtering out elements with selection, layering operations take less time than with a traditional scene index. A fisheye scene index increases the number of visible correspondents, while reducing time to find a desired element. Bimanual scrolling allows a selected correspondent, manipulated with the right hand, to remain as the focus of the fisheye while the left hand scrolls the view.

We apply a fisheye visualization to a traditional scene index. Prior work has investigated various advantages and disadvantages of fisheyes in numerous contexts [Cockburn et al. 2009]. Our findings show that fisheyes can be faster for finding the desired correspondent in a large scene index. One of the primary challenges with fisheyes is target acquisition, since targets are continuously moving due to changes in the fisheye focus. One solution is to use speed-coupled flattening to remove the fisheye effects as panning or scrolling velocities increase [Gutwin 2002]. We employ speed-coupled flattening with inertial scrolling in LayerFish. For target acquisition with hierarchical structures of text, Fisheye Menus [Bederson 2000] are slower than traditional hierarchical menus [Hornbæk and Hertzum 2007]. Our present investigation focuses on flat scene index structures with thumbnail images. Future work can investigate hierarchical scene indexes.

Specifying the focus is a key interaction component for fisheyes. In LayerFish, the focus is always under the user's first touch. The focus can be repositioned with drags, or fixed in place, using bimanual scrolling. An alternative would be to always have the focus in the center of LayerFish. On the one hand, this may support effective browsing of the list of visual media using touch, similar to Apple's Cover Flow[1]. However, a shortcoming is that attempting to layer a selected correspondent would become more challenging further from the focus, as the size of correspondents shrink.

In Guiard's bimanual model, the left hand physically positions elements for action by the right hand. Prior Guiard-abiding techniques have applied this idea literally, where

[1] https://en.wikipedia.org/wiki/Cover_Flow

both hands interact simultaneously with the same element [Hinckley et al. 2010]. We extend kinematic chain interaction design, by providing correspondents for the left hand to activate. The right hand's direct operations are limited to the activated elements. This design avoids physical interference by the left hand and the occlusion of overlapping elements. We can support additional operations, beyond translate and scale, such as crop and translucence, through additional gestures with the left hand on correspondents to activate different operations.

In order to focus investigation on layering performance, participants were not able to zoom and pan the design space. The High Density Overlap tasks where all elements overlap completely, while representing extreme scenarios, would gain little benefit from zooming and panning. However, we presently use LayerFish in a design environment with an infinite, zoomable canvas. Future work will look at how a zoomable space affects selection and layering performance with LayerFish.

Designers are accustomed to working with their hands. While working with physical media, visual designers generate new ideas by exploring different arrangements of elements, moving them around and sketching over and amidst these elements. Future work will investigate LayerFish in architecture contexts, where designers create complex visual representations with overlapping content. We hypothesize that visual designers would benefit from interactive tools that allow them to directly gesture with their hands to explore a space of ideas.

# CHAPTER VI

## WEARABLES AS CONTEXT FOR GUIARD-ABIDING BIMANUAL TOUCH*

HCI is witnessing a proliferation of very large and very small form-factors in an ecosystem (or "society") of devices that complement one another. Whiteboards and tabletops offer multi-user interaction, while wearables such as watches [Apple Watch 2016; Chen et al. 2014a], fitness bands [Fitbit 2016; Microsoft Band 2015], and rings [Strange 2014] promise personal interaction that is contextual in a fine-grained way.

In particular, during multi-user, multi-touch interaction on large displays, standard touchscreens rarely distinguish the contribution of individual users (much less the preferred vs. non-preferred hands) from one another. The problem is that all touches trigger the same events, generic "contacts" that seem identical without further context [Dietz and Leigh 2001; Kharrufa et al. 2015; Marquardt et al. 2010]—namely, who touched the display, and which hand produced the event. This missing context is especially problematic for bimanual interaction on large displays, as such a setting compounds these ambiguities. And while wearables are just one possible approach to sense such context, they do offer an expedient solution that rides on existing trends towards a society of devices where users carry, wear, or encounter many inter-connected devices—a mobile approach that scales to situated displays (rather than requiring new sensors at each location).

We demonstrate that augmenting touch with such context supports phrasing together [Buxton 1986] inputs in a manner consistent with bimanual asymmetry as set forth by Guiard [1987]. Our key point is that this added context—who touches and with which hand—enables design of interactive dialogues that can flexibly assign appropriate roles to

*Edited reprint with permission from "Wearables as Context for Guiard-abiding Bimanual Touch" by Andrew M. Webb, Michel Pahud, Ken Hinckley, and Bill Buxton, 2016. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, ACM, New York, NY, USA, 287-300. DOI: `https://doi.org/10.1145/2984511.2984564`. 

each hand, including symmetric role assignments where appropriate. Because of the arbitrary limitations and ambiguities inherent in standard multi-touch technology, most previous systems exploring bimanual touch—including efforts motivated by Guiard's insights—have not been able to fully support this design goal.

In this chapter, we support our contribution:

> âĂęthe use of wearables to identify which user, and distinguish what hand, is touching a large display—and to thereby **phrase together** direct-touch interactions in a manner consistent with the **bimanual-asymmetric assignment of roles** to the hands—right vs. left, preferred vs. non-preferred—as set forth by GuiardâĂę

...as follows: First, we review related work. Second, we detail the design of Affinity, which uses a single wearable in combination with phrasing to support bimanual-asymmetric interactions. We describe a few details of how we use wearables to identify users and sense hands (details largely explored already by others, and which are therefore not our key contribution). Third, we discuss MoodBoard, which uses two wearables to explore further possibilities. And fourth, we offer an informal evaluation and discuss some implications of our work.

## VI.1 Related Work

Our work builds on insights from bimanual input, wearables (and other sensors) plus touch, and large-display interaction.

### VI.1.1 Asymmetric (and Symmetric) Bimanual Interaction

Guiard's work has motivated many techniques [Hamilton et al. 2012; Hinckley et al. 2010; Kurtenbach et al. 1997]. Yet the difficulty inherent in unambiguously supporting proper lateral preferences for standard direct-touch input has not (to our knowledge) been

previously articulated.

#### VI.1.1.1 Direct vs. Indirect, Mechanical Intermediaries, Pen vs. Touch

It is sometimes possible to design around hand ambiguity by favoring symmetric bimanual interactions [Hinrichs and Carpendale 2011; Kurtenbach et al. 1997; Leganchuk et al. 1998; Wu and Balakrishnan 2003], with interchangeable roles of the hands. But, it is telling that many previous bimanual-asymmetric techniques do not use direct touch but instead rely on mechanical intermediaries such as pens [Brandl et al. 2008; Hinckley et al. 2010] (or indirect input devices including trackballs [Bier et al. 1993], or pucks [Kurtenbach et al. 1997; Leganchuk et al. 1998], or touchpads [Buxton 1986]) that distinguish the contribution of the left hand from that of the right. However, even predominately asymmetric designs such as Toolglass often interleave symmetric interactions [Bier et al. 1993; Kabbash et al. 1994], so flexibly mixing both styles (symmetric and asymmetric) is important to support as well.

Pen + touch together [Brandl et al. 2008; Hamilton et al. 2012; Hinckley et al. 2010; Matulic and Norrie 2013] is an interesting test case. Here, direct-touch is present, but typically only strokes articulated by the pen—and not touches with the pen temporarily "tucked" in hand—can definitively be associated with the preferred hand [Hinckley et al. 2014]. Conté [Vogel and Casiez 2011] offers a rich example of this style of interaction, but it must initiate many of its interactions via the digital Conté crayon in the preferred hand to do so. Thus, many pen+touch systems still face ambiguities in terms of Left-Hand Precedence and Right-to-Left Spatial Reference for touch, and in general it has not been recognized that the inability to distinguish left vs. right limits the interaction fidelity vis-á-vis Guiard.

#### VI.1.1.2 Symmetric Interaction

Through interchangeable roles of the hands, symmetric bimanual interactions afford parallel action. This can take the form of two independent movements, such as dragging a pair of objects to two different locations [Chatty 1994], or the movements may comprise elements of a compound task, as in two-handed stretchies [Bier et al. 1993; Kabbash et al. 1994; Kurtenbach et al. 1997], or spline curve manipulation [Latulipe et al. 2006]. Task difficulty, divided attention, and visual integration can all influence the degree of parallelism attainable [Balakrishnan and Hinckley 2000]. By distinguishing the hands, our work demonstrates how both asymmetric and symmetric interactions—potentially with either strict or flexible assignment of roles to the hands—can be richly supported through appropriate context sensing.

### VI.1.2 Wearables (and Other Sensors) Plus Touch

Although our insights are not particularly wedded to wearable devices, a number of previous systems do leverage personally-worn devices to augment touch and multi-touch interactions. Wrist-worn devices on either the preferred or non-preferred hand (but not both) have been used in combination with touch on phones [Chen et al. 2014a] and to authenticate users [Kharrufa et al. 2015; Mare et al. 2014].

Vision-based techniques [Ewerling et al. 2012; Ramakers et al. 2012; Zhang et al. 2012] or proximity sensors [Annett et al. 2011] can sense which hand approaches a touchscreen. However, it remains difficult to bridge the "identity gap" of associating tracked objects (blobs) with a specific individual. Biometric properties inherent in touch [Blažica et al. 2013], capacitive coupling [Dietz and Leigh 2001], and combining inertial sensors on mobile devices with vision-based approaches [Rofouei et al. 2012] offer other possibilities to associate touches with an individual. Fingerprint sensing [Holz and Baudisch 2013] or fiduciary-tagged gloves [Marquardt et al. 2010] can distinguish the hands and

even individual fingers, as can instrumenting fingertips with optical sensors [Yang et al. 2012], but these efforts do not unpack what this means for bimanual interaction in depth.

Identifying, articulating, and exploring the implications of this added context (rather than the particular manner in which it is sensed) is the focus of our contribution.

### VI.1.3 Interaction on Large Displays

Large displays afford workspaces for collaboration and creative work [Ardito et al. 2015]. Their scale affords use of both hands by multiple users, but previous work has not remarked upon the inherent ambiguities in multi-touch with respect to bimanual input. For large public displays, breakdowns in personal space may occur, where users will reach across each other to interact with objects, causing sensing and interaction design issues [Azad et al. 2012]. Affinity primarily relies on phrasing [Buxton 1986] to avoid this, where personal spaces are delineated by activating a tool sheet with the non-preferred hand (see Figure 45); the system assumes users do not reach into one another's tool sheets. In MoodBoard, we avoid the issue completely by leveraging two wearables, one on each hand, to disambiguate touches.

#### VI.1.3.1 Electronic Whiteboard

Interactive wall-sized displays can bridge the gap between informal whiteboard use and the rigid data manipulations of the desktop [Guimbretière et al. 2001]. Tivoli extends the functionality of physical whiteboards with digital affordances, while preserving an informal feel [Pedersen et al. 1993]. Flatland provides functionality for managing space, with flexibility for different applications [Mynatt et al. 1999]. Range uses implicit interactions to transition the whiteboard among different modes based on proximity [Ju et al. 2008]. Our work, by contrast, focuses on the role of bimanual direct-touch (and pen) interaction for design activities.

#### VI.1.3.2 Design and Mood Boards

We explore our techniques in the context of mood boards—assemblages of images, sketches, and other visual media that designers use in studio environments for both problem-finding and problem-solving [Garner and McDonagh-Philp 2001; Lucero 2012] by drawing connections among ideas through visual juxtaposition [Buxton 2010]. Portfolio Wall, for example, enables designers to drag images from personal computers to a wall-mounted display [Buxton et al. 2000]. Funky Wall [Lucero et al. 2008] and MessyBoard [Fass et al. 2002] offer other examples, but our exploration of Affinity and MoodBoard offers the only example that leverages identified touches for bimanual input.

#### VI.1.3.3 Tabletops

Although our focus is on vertical displays, and electronic whiteboards in particular, our work is heavily informed by related techniques drawn from tabletops. For example, bimanual interaction has been extensively explored in this context, including whole hand gestures [Wu and Balakrishnan 2003] and cooperative gestures [Morris et al. 2006]. Some systems can differentiate the hands, for example using different parts of the hands to perform different operations [Marquardt et al. 2010], or moving individual elements with the preferred hand and groups of elements with the non-preferred hand [Annett et al. 2011]. However, these previous systems do not address the implications for bimanual touch as we do.

### VI.1.4 Implications

As should now be clear, despite many previous efforts that have addressed bimanual interaction, direct touch, and touch augmented with various forms of sensing, to our knowledge no previous system has put all of these elements in play simultaneously. In particular, our work is the first to articulate the underlying ambiguities that direct-touch

presents with respect to lateral preference, i.e., the proper assignment of distinct roles to the hands during asymmetric bimanual interaction. Without a clear understanding and enumeration of these issues, it is difficult to design systems that adhere to the principles of asymmetry as articulated by Guiard—much less to contemplate designs that explore other mappings, or that (intentionally) diverge from the Guiard-abiding perspective in service of other design goals.

In the following sections, our techniques demonstrate various design considerations and trade-offs for such interactions, including the interleaving of bimanual-symmetric inputs. And to be absolutely clear, the contributions of the techniques are in their illustration, exploration, and bracketing of various design decisions in these regards, rather than in the underlying gestures themselves (taps, drags, chops, and so forth). Collectively, these interactions show in some detail why it is problematic to design bimanual interactions that incorporate Guiard's well-known principles without sufficient information on which user and which hand—information that is sorely lacking on almost all touch displays in common use.

### VI.2 Affinity: Instrumenting Left Hand

Our investigation began with designing a user experience for bimanual-asymmetric interaction involving one wearable per user. We developed Affinity, a prototype environment for a pair of users to author affinity diagrams [Beyer and Holtzblatt 1997], where the left (non-preferred) hand of each user is instrumented with a wearable sensor, the Microsoft Band [Microsoft Band 2015]. Affinity diagrams involve recording ideas on post-it notes, reflecting on the relationship among ideas, organizing ideas into groups, and drawing lines and shapes to convey connections [Beyer and Holtzblatt 1997].

### VI.2.1 Bimanual-Asymmetric Interaction

In Affinity, a user creates digital post-it notes and shapes with bimanual-asymmetric gestures through an in-place tool sheet—a transient interface. The tool sheet is activated with a touch from their left hand (see Figure 45, top-left), following Guiard's principle of Left-Hand Precedence. The wearable identifies the specific user that touched, presenting that user's name above the tool sheet and a personalized set of commands (see Figure 45, bottom). The tool sheet can be moved by dragging the left hand touch. The user selects a colored note or shape command in the tool sheet with their right (preferred) hand, following Guiard's principle of Right-to-Left Spatial Reference. A new note or shape appears next to the tool sheet.

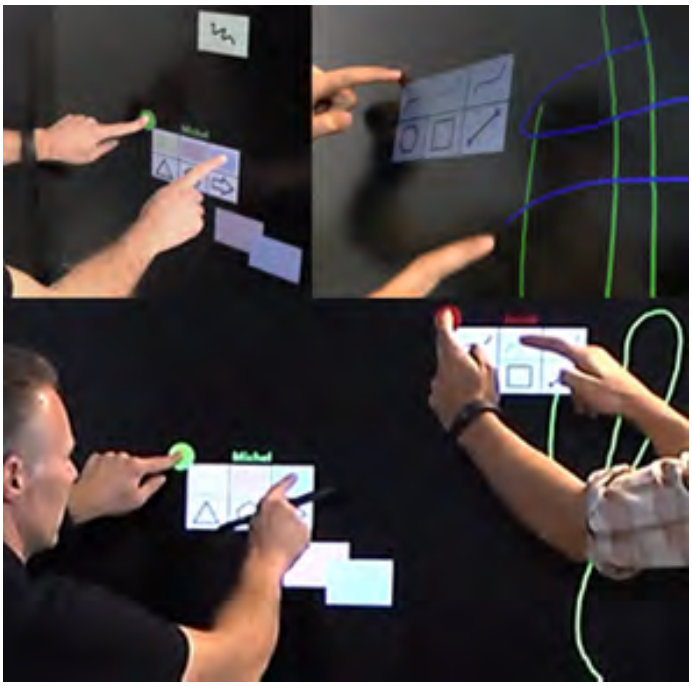

Figure 45: Snapshots of Affinity. Touches with the left hand identify the user and present that user's unique tool sheet, while touches with the right hand select tools, draw, and move notes. User's name and color are shown above each tool sheet.

### VI.2.2 Combining Asymmetric and Symmetric Interactions

Command selection is asymmetric, where the left hand activates a tool sheet and the right hand selects a command, such as Draw Rectangle. In Affinity, once a command is selected, the two hands function symmetrically to manipulate parameters of the command in parallel, such as scaling and positioning rectangles [Leganchuk et al. 1998], lines, or circles.

### VI.2.3 Unimanual Interaction: Right Hand

Notes and shapes can be freely arranged to organize ideas by dragging them (via touch) with the right hand. The pen is used to draw or write ideas inside of notes. The user can also draw colored strokes to connect notes using the pen or a touch on an empty area of the space with their right hand.

### VI.2.4 Wearable + Phrasing = Guiard-abiding Bimanual Touch

In Affinity, we employ only one wearable. This level of sensing is sufficient to identify and differentiate any interactions initiated by the left hand. And the design of the interactions is such that the left hand leads all bimanual-asymmetric transactions (Left-Hand Precedence).

Although the right hand remains uninstrumented, Affinity's design limits the potential of this to cause problems. Given users' reluctance to invade the personal space of one another [Greenberg et al. 2011], once the left hand establishes a frame of reference (in the form of the in-place tool sheet), it is reasonable to assume that an uninstrumented touch in the tool sheet is most likely the right hand of the same user. Thus, the role of the right hand can be distinguished (within reasonable assumptions) for such transactions even though it is not directly sensed.

Note that this also leverages Buxton's notion of phrasing to connect the activity of the

right hand to the reference frame established by the left hand. As long as the left hand remains in contact with the display, the "phrase" remains open, allowing the right hand to continue the transaction (i.e. by dragging "through" the tool sheet called up by the left hand.)

As noted above the design also supports unimanual actions by the right, uninstrumented hand, such as moving around the virtual post-it notes on the display. The system knows that an uninstrumented touch is not performed by the left hand, and therefore it can unambiguously interpret the input using the unimanual, preferred-hand role. Touching with the left hand would trigger a bimanual-asymmetric transaction, i.e. bring up the tool sheet, instead.

Of course, inputs articulated by any walk-up users who are completely uninstrumented would remain ambiguous. However, the fallback of the system here—namely, to interpret all such transactions as unimanual inputs—offers a reasonable fallback, and indeed there would seem to be some virtue in only allowing authenticated users to access the full functionality of the application as manifest in the tool sheets.

Thus, with careful attention to detail in the design, even a single wearable can support a significant range of interactions that adhere to Guiard's principles.

### VI.2.5 Hardware and Signal Processing

Affinity consists of the following hardware components: (1) a large 55-inch pen and multi-touch display on a vertical stand; (2) two Microsoft Bands; and (3) a computer with Bluetooth for processing band sensing data and running our affinity diagram application.

The Microsoft Band is a fitness band worn around the wrist. Its sensors include a 3-axis accelerometer and gyroscope. The Band communicates wirelessly with other devices via Bluetooth, and has a sampling rate of 60 Hz. We use accelerometer signals from the band to detect the hard-contact forces (spikes) that result from the hand coming into contact

with the display. We identify touches with the left hand by detecting spikes of a particular size and width in the band's accelerometer signal and associate those spikes with touch-down events. These spikes are larger and quicker (narrower pulse width) than normal hand movements, which exhibit more gradual changes in acceleration.

We look at the x-axis of the accelerometer, which is the axis parallel to the orientation of the forearm when the device is worn. Similar to the spike detection in [Hinckley et al. 2014; Li et al. 2011], we detect spikes by sampling for a bump (above a threshold) in the second order finite difference with a window of 100 ms (see Figure 46). If a spike is within the 100 ms window following a touch-down event, we associate that touch to the left hand. If a touch-down event occurs, but there is no spike within 100 ms, we associate that touch to the right hand.

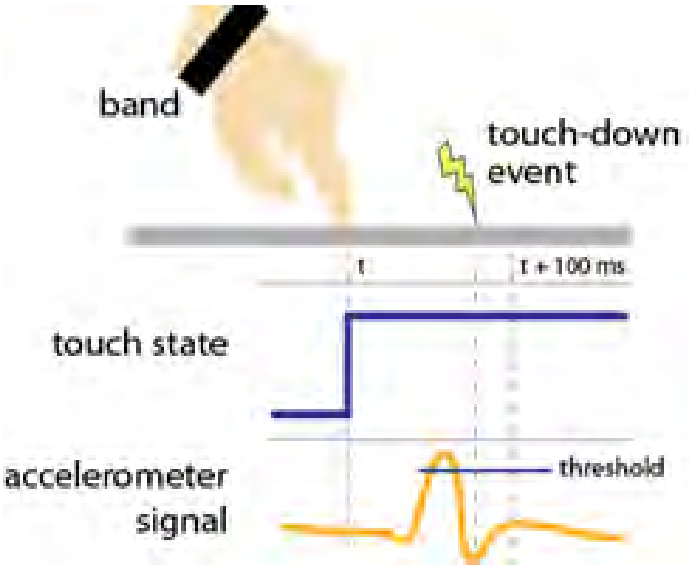


Figure 46: Diagram of signal processing for identifying touches based upon spikes in accelerometer signal.

### VI.3 Moodboard: Instrumenting Both Hands

We further demonstrate how wearables—this time one on each hand—support Guiard-abiding bimanual-asymmetric interactions on large displays. We present MoodBoard, a prototype environment for authoring mood boards with a single user or pair of users.

As with Affinity, we instrument the left (non-preferred) hand with a Microsoft Band. We instrument the right (preferred) hand with a different wearable, an in-house designed ring, placed on the index finger. We intentionally use two types of wearables. In everyday life, it seems unlikely that a user would choose to wear two watches, or two fitness bands, but it may be more plausible that a user would own two different (and complimentary) wearables, such as a fitness band on the non-preferred hand and a "smart ring" type of device [Kienzle and Hinckley 2014; Ogata et al. 2012] on the index finger of the preferred hand.

In MoodBoard, we use the orientation of devices as an additional parameter for interaction [Wilkinson et al. 2016]. We detect three orientation states: normal, side, and back (see Figure 47). Normal orientation is the standard touch orientation when touching with the tips of fingers. Side orientation is when the pinky finger side of the hand is used. Back orientation is when the back of a finger is pressed against the display. Details of our orientation detection methodology are presented later.

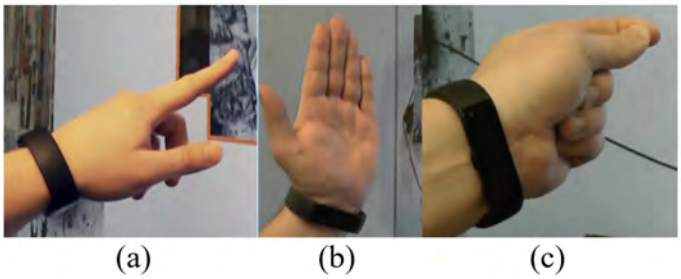


Figure 47: Detected orientations: (a) normal, (b) side, (c) back.

We designed a suite of interaction techniques for creating and organizing content in a mood board. The techniques demonstrate various considerations and trade-offs for Guiard-abiding bimanual touch interaction on large displays. Our core interactions are well-known gestures drawn from prior work; it is the way we augment them with the added context of which user and which hand—and how we use this to illustrate various design choices—that is new. These techniques further demonstrate how additional context from wearables, including orientation of the hand—in conjunction with Guiard's principles—can support phrasing together rich, bimanual-asymmetric interactions. Adding a wearable on the right hand addresses ambiguity issues for lateral preference, removing the need for any assumptions about what an uninstrumented hand represents, and thereby affording a richer design space that can assign a variety of distinct, complementary roles to each hand.

In the sections that follow, we present a series of interactions that we prototyped within MoodBoard, with the intent of exploring this design space and the various possibilities that it offers designers. Our intent is not to argue that any one particular technique represents the epitome of our approach, but rather to cover enough of the design space to compare and contrast various design issues, as well as to illustrate that many novel bimanual-asymmetric possibilities arise.

### VI.3.1 ChopZ

Elements in a mood board can overlap. The user needs techniques for adjusting the visual stacking order (or z-index), so that the desired elements can appear on top of or below other elements.

Design tools, such as Adobe Illustrator [Adobe Illustrator 2016], use an ordered list to represent visual stacking order. In the list, elements are organized top to bottom where elements higher up in the list appear above elements lower in the list. Visual stacking order is changed by reordering the list using drag and drop. The list appears in a floating palette

near the display edge. With large displays, with multiple users, a floating palette near the display edge is not easily accessible.

Contextual menus, such as those in Microsoft PowerPoint, can allow adjusting visual stacking order of an element in discrete steps: bringing it up one step, down one step, all the way to the top, or all the way to the bottom. Getting the desired stacking order with this approach can be tedious as the discrete changes may require multiple executions.

In-place techniques allow adjusting visual stacking order with mouse or pen input [Ramos et al. 2006]. Touch pressure has been used to adjust visual stacking order between two elements [Davidson and Han 2008].

ChopZ is a bimanual-asymmetric transient interface for adjusting the visual stacking order of elements. It allows for reordering of elements using drag and drop, while supporting in-place activation and selection of elements. ChopZ enables multiple users to perform the operation simultaneously.

ChopZ is activated with a side-of-hand "chopping" gesture [Matulic and Norrie 2013; Wu and Balakrishnan 2003] using the left hand (see Figure 48). The chop gesture phrases the interaction. While the chop is held down, any right hand actions are input for ChopZ. Releasing the chop, immediately deactivates ChopZ and ends the interaction. This follows Guiard's principles of Left-Hand Precedence and Right-to-Left Spatial Reference. The left hand begins the interaction with a chop gesture. The right hand defines the bounds of selection in reference to the left hand chop.

After chopping, the user touches down with their right hand to define a scope of elements on which to interact. The selection area is visually represented by a gray rectangle, vertically bounded by the top and bottom of the display and horizontally bounded by the two hands (see Figure 48, left). Moving either hand adjusts the horizontal bounds of the selection area. The user may perform a bimanual-symmetric gesture by moving both hands together. A flick gesture with the right hand away from the left hand will automatically

Figure 48: *Left*: Using right hand to select scope after chop with left hand. *Right*: Using right hand to move individual elements to change visual stacking order.

select all elements in the direction of the flick. We suspect that flick selection will be particularly useful on large displays where the reach of a user is smaller than the display size. Lifting up the right hand touch, finalizes the selection.

Upon finalizing selection, an animated transition (500 ms) occurs, rendering a view of the selected elements as if seen from the side of the board. In other words, the x-axis is re-mapped to the z-axis, and a perspective transformation is applied to the elements to convey a side view (see Figure 48, right). As a result of the re-mapping, elements are ordered in the x-dimension according to their visual stacking order. Elements at the top of the stack appear on the left-most side, and elements at the bottom appear on the right-most side.

With the right hand, users can drag elements to adjust their visual stacking order. Moving elements to the left, moves them on top of elements to their right.

After the user is finished adjusting elements stacking order, they can lift up their left hand, deactivating ChopZ. An animated transition (500 ms) returns the selected elements to their original positions, although with a new stacking order based upon the user's changes.

### VI.3.2 Straightedge Tool

A straightedge is a valuable tool in visual design, enabling designers to draw straight lines that create visual structure, such as showing connections among elements or producing geometric shapes on which to align elements around. Straightedges also enable aligning elements along a line.

We present a bimanual-asymmetric quasimode for using a straightedge that follows Guiard's Right-to-Left Spatial Reference principle. The left hand gestures to create a movable virtual straightedge. Then, using the right hand, a user can draw straight lines using a pen, or move elements parallel to the straightedge using touch.

The straightedge tool is activated using a two touch gesture with the index finger and thumb of the left hand (see Figure 49). This gesture phrases the straightedge operation for the user. While maintaining this gesture with the left hand, any touches or pen input with the right hand by this user will parameters for the straightedge operation.

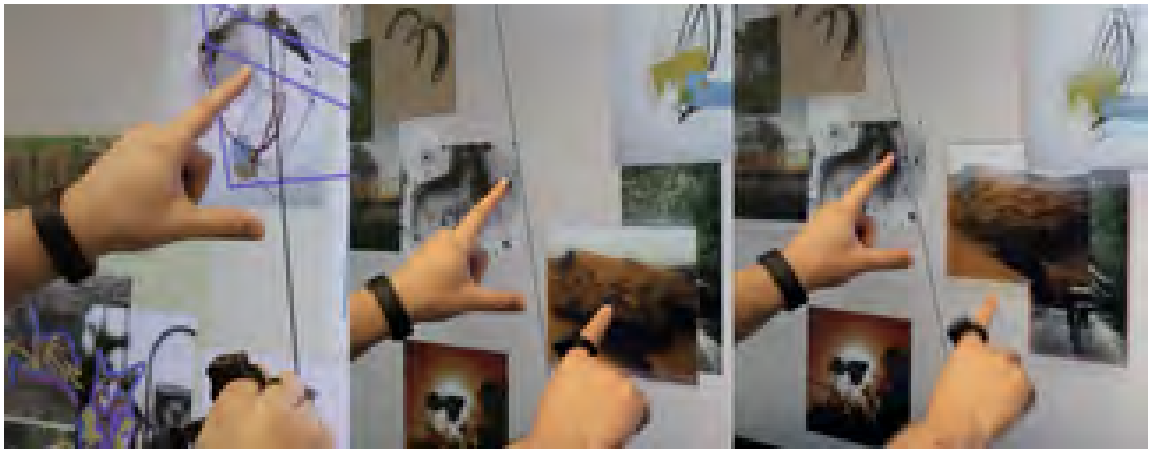

Figure 49: Straightedge tool activated with left hand. Input modality with right hand and context specify operations: (a) using pen draws a straight line; (b) using touch on an image moves that image along the line; (c) using touch on the background moves the entire space on that side of the line.

Upon activation, a thin line representing the straightedge appears near the left hand. Moving the left hand translates and orients the straightedge line, just as if holding a physical straightedge tool. The line extends from one edge of the display to another. The slope of the line is defined by the two touch points. We offset the straightedge line by 40 pixels to avoid occlusion with the hand. The direction of the offset is determined by the orientation of the band to ensure that the line appears away from the hand instead of under it. This shows how implicit parameters of the Band enrich the interaction. When gravitational forces on the y-axis of the accelerometer read greater than 0.5 or less than -0.5, we position the line to the left of the hand; otherwise, we position the line to the right.

With the straightedge activated, the user can draw ink with the pen (in the right hand). The ink is snapped to the straightedge line, allowing for the creation of straight lines (see Figure 49a). Alternatively, the user can use touch with the right hand to select a single element and move that element along a line parallel to the virtual straightedge (see Figure 49b). If the user touches on an empty area of the mood board, they can move all elements on that side of the line parallel with the straightedge (see Figure 49c). This enables quickly shifting large regions of elements around on the mood board to make space for new content without having to make a selection.

### VI.3.3 Group Sheets

As connections emerge while mood boarding, a user may wish to group together a set of elements that may or may not overlap so that those elements are explicitly connected, allowing the user to move those elements together without losing their relative positioning with each other.

Group sheets are a pen+touch transient interface for defining persistent groups of elements. A group sheet represents a group of elements as a rectangular, translucent region (see Figure 50).

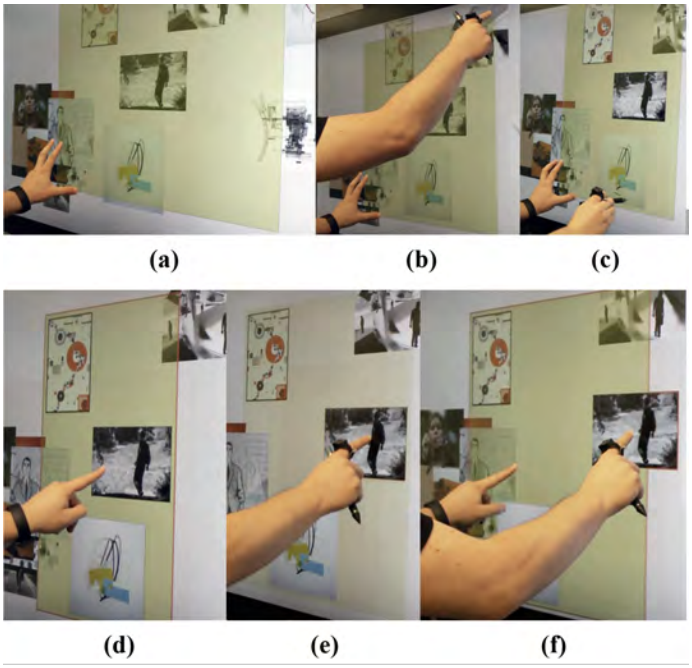


Figure 50: States of group sheet creation (a-c) and editing (d-f): (a) three finger right angle gesture with left hand creates a new sheet; (b) bimanual-symmetric gesture adjusts size and position; (c) trace over elements with pen to add to sheet; (d) move entire sheet with left hand; (e) move an individual element with right hand; (f) edit group contents by phrasing with left hand and dragging an element out with right hand.

### VI.3.4 Phrasing: Sheet Position, Size and Contents

Creating a group sheet begins with a three-finger right-angle gesture with the left hand (see Figure 50a). A sheet with a unique background color (with 43% opacity) appears. Position of the right angle gesture defines the bottom-left corner of the sheet. This left hand gesture phrases Group Sheet transactions. While the left hand remains down, the user can modify the sheet's position and size using both hands through bimanual-symmetric interaction, and add elements to the sheet using the pen in the right hand.

Using a touch with the right hand, the user can specify the opposing top corner of the sheet (see Figure 50b). Moving both hands simultaneously defines the position and size of the sheet, based upon the positions of the left and right hands.

With the right hand, the user marks over with a pen (see Figure 50c) or taps with touch elements to add them to the sheet. A short mark or tap is enough to add an element quickly. However, for selecting thin ink strokes that are difficult to precisely mark with pen or touch, the user can also trace over the ink stroke with the pen. This ensures that the stroke is selected, and not elements behind the stroke that were accidentally traced over. On pen up, whatever elements the user marked are added to the sheet. Lifting up the left hand ends group sheet creation.

Touching the sheet again with the left hand selects the sheet and activates an edit mode. The user can change which elements are in the sheet using the right hand. With the pen, the user can mark over elements to toggle adding or removing them from the sheet. With touch, elements can be removed by directly dragging the element out of the sheet (see Figure 50f), or conversely, holding the element in place while dragging away the sheet with the left hand.

#### VI.3.4.1 Unimanual Interaction: Positioning

We use different hand mappings for spatial positioning of sheets and elements (see Figure 50d-e). With the left hand, the user positions an entire group sheet, and with the right hand, the user positions individual images within a group sheet. These mappings are consistent with our other unimanual interactions beyond group sheets (see Unimanual section).

#### VI.3.4.2 Tool Sheets Revisited: Ink Styling

In Affinity, left hand touches activate an in-place tool sheet. A single touch provided an easy way to activate tool sheets, but in MoodBoard, a single touch is used to move elements around. Positioning elements is a critical operation for mood boards to support juxtaposing ideas. However, we can take advantage of our ability to sense orientation of wearable devices to still use a single touch, but with the back of the finger, to activate the

tool sheet.

An in-place tool sheet is activated with the left hand by pinching together the index finger and thumb and pressing the back of the index finger at the fingernail against the display (see Figure 51). We designed this interaction with a physical mapping, where pinching the index finger and thumb simulates holding the top corner of the tool sheet, as if it were a physical sheet. Dragging the activating touch, moves the tool sheet, maintaining the relative positioning between touch and tool sheet. Lifting up the activating touch deactivates the tool sheet, removing it from the display.

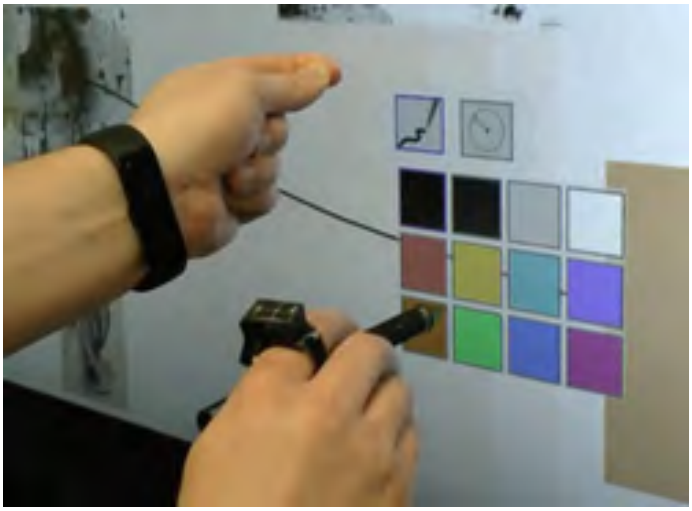

Figure 51: Tool sheet activated with back of finger gesture using left hand. User can select color and adjust brush size.

The mood board's in-place tool sheet enables changing the color and size of ink strokes (see Figure 51). The tool sheet is interacted with using pen or touch with the right hand. The user can draw ink strokes while the tool sheet is active, allowing the user to quickly create multi-colored drawings without having to repeatedly activate the tool sheet.

### VI.3.5 Unimanual: Spatial Positioning

MoodBoard also supports unimanual interactions. Since we can identify the hands, we can map different actions to each hand. We map spatial positioning of individual elements to the right hand, while we map positioning of groups to the left hand. Others have used a similar mapping [Annett et al. 2011].

Arranging the spatial positioning of elements within a mood board is one of the most basic and frequently performed actions. In MoodBoard, individual elements are freely positioned using a single finger touch and drag with the right hand (see Figure 52, left).

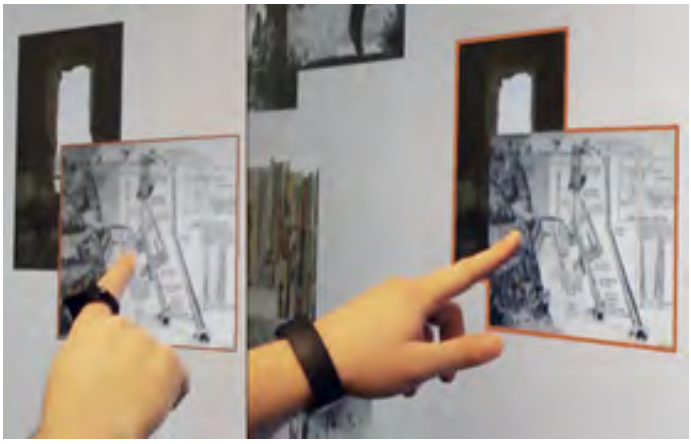

Figure 52: *Left*: Right hand selects single element on top (thin orange outline). *Right*: Left hand selects two overlapping elements (thick orange outline).

Overlapping elements visually form groupings through their spatial contiguousness, similar to a roughly arranged pile of photos or documents. The lack of white space between these elements connects them as a whole. A single touch and drag on an element with the left hand moves the touched element and all elements which it overlaps or are overlapped by it while maintaining their relative positions (see Figure 52, right). We use visual feedback to differentiate individual element selection from overlapping group se-

lection. For single elements, a thin one-pixel line outlines the selected element. For overlapping groups, a thicker four-pixel line contours the contiguous group of elements. The color of the feedback matches the color assigned to a user.

#### VI.3.5.1 Bimanual Interaction through Phrasing Unimanual Together

We address lateral preference by assigning distinct roles to the hands for spatial positioning. The fine-grain positioning of individual elements is assigned to the right hand. While, the coarse group positioning is assigned to the left hand.

These role assignments can be combined to support bimanual-asymmetric interactions without need for any additional gesture or phrasing. For example, in some cases, an element may overlap with others, but the user does not wish to move that element with the others. The two hands may work together. The left selects an overlapping group, and the right hand pulls an element away from the group (with a drag), removes an element from the selection (with a tap) or holds an element in place (with a press and hold) while the left hand drags the group away.

### VI.3.6 Hardware and Signal Processing

MoodBoard consists of the same hardware components from Affinity with the addition of the ring. The ring contains an inertial measurement unit (IMU), wireless transmitter for sending data from the IMU, and a Velcro strap for fastening it to the user's finger (while easily accommodating variable sizes). The ring samples the three-axis accelerometer, gyro, and magnetometer embedded in the IMU at 90 Hz.

We build upon our signal processing techniques from Affinity. Again, we identify which user is touching and with what hand by detecting spikes in the accelerometer signals from the wearable devices. Signals from both wearables are passed through a high-pass filter to remove effects of slow changing forces, such as gravity. The Îś-values for our high pass filter are 0.75 for the band and 0.9 for the ring.

We can identify orientation by looking at the gravitational force on a wearable device. We apply a low-pass filter to the accelerometer signal to isolate slow changing, gravitational forces. The Îś-value for the low-pass filter is 0.1 for both wearables. We define a vector for each orientation that describes that orientation in terms of gravitational force. We subtract the current gravitational vector (result of low pass filter) from each of the orientations' vectors, and observe the smallest difference. The orientation with the smallest difference is defined as the current orientation.

### VI.4 Preliminary User Evaluation

We ran a preliminary informal study with 8 participants (4 female, all right-handed, 26-66 years old). All participants had experience using touch devices, but minimal to no experience using large displays. The goal of the study was to gather user feedback on our interaction techniques. Before investigating the more challenging multi-user context, we first wanted to better understand individual user experience. Thus, study sessions involved only one participant.

Participants were given an introduction to MoodBoard and a step-by-step explanation and demonstration of all the interaction techniques. The participants put on the band and ring devices, and the investigator guided the participants through a repeated process for each technique involving first training, then performing the technique, followed by a short questionnaire. Separating out the techniques in this way allowed us to gain focused feedback from the participants.

After completing this process for all techniques, the mood board canvas was cleared, and participants were asked to author a mood board using the techniques they had just tried (see Figure 53). This allowed us to both observe what techniques stuck out to participants, but also provided ecological validity that our techniques support mood board design. Each session lasted 40-70 minutes.

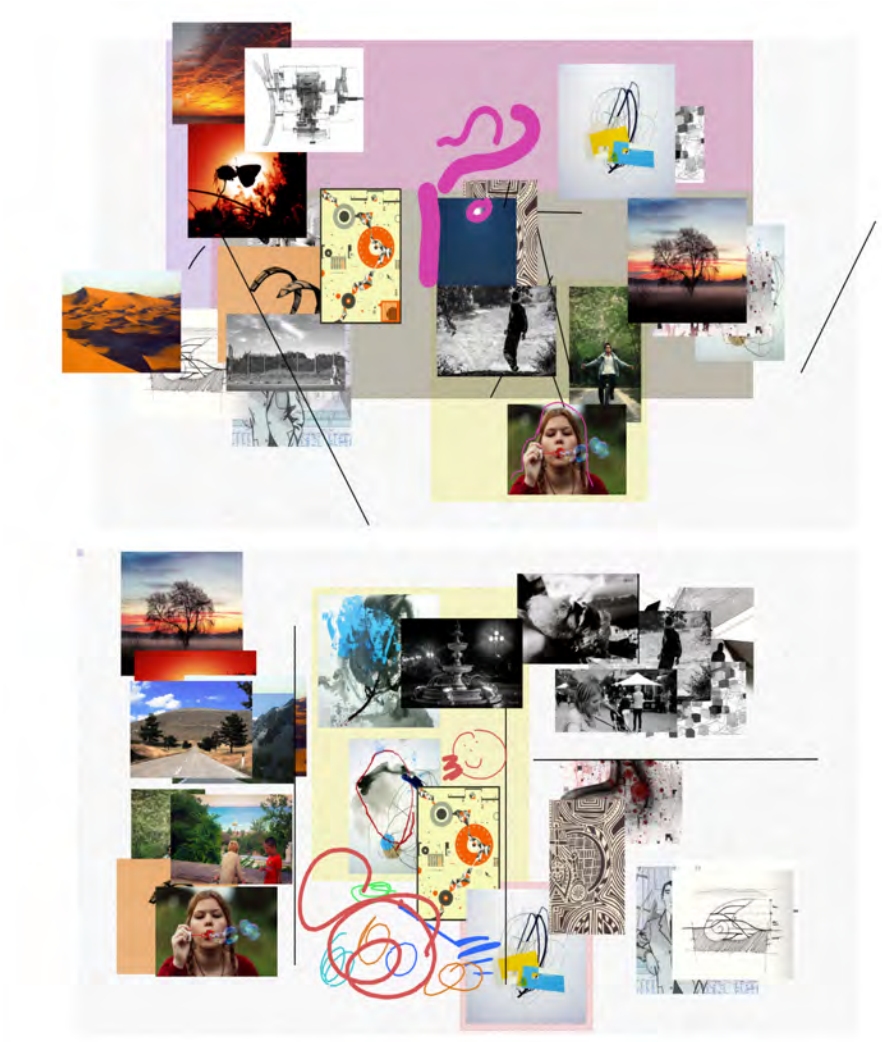

Figure 53: Two mood boards authored by participants.

**ChopZ**: Most participants found ChopZ easy to use. Many found value in the ability to define a scope and freely arrange the stacking order of elements without having to use some out-of-place widget or repeated contextual menu commands. Some participants reported fatigue issues with maintaining the chop gesture. Many of our bimanual techniques phrase interactions through a sustained left hand gesture. An implication is then that phrasing gestures for longer tasks should have reduced muscle tension. However, this

may be particularly challenging on large vertical displays where tension can vary by user depending on user height and where on the display the user interacts.

**Straightedge Tool**: Participants felt the straightedge tool was quick and easy to use, particularly with the pen. They had no problems understanding the differences between using the pen and using touch. Some participants did express a desire for allowing alignment on the line perpendicular to the straightedge. They wanted the ability to use the straightedge as a virtual wall, preventing elements from moving past the line, as a way of aligning them against the straightedge.

**Ink Style Menu**: All participants found the in-place ink style menu easy to perform. Some participants did take time getting used to touching with the fingernail. One participant experimented and found the technique was easier for him to perform if he used the middle knuckle (proximal inter-phalangeal joint) on his index finger [e.g. Harrison et al. 2011].

**Single Finger Positioning**: Participants were able to quickly pick up on the different positioning operations performed with each hand. They found it intuitive and desirable. During the mood board creation phase at the end of a session, a few participants did report accidentally using the wrong hand at first, but felt that with practice, this issue would not occur. This error typically involved using the left hand instead of the right to move a single element. The selection highlight provided quick visual feedback, helping participants recognize that they needed to use the other hand. We suspect that ergonomics—in that one hand is closer to the target element than the other—may influence participants hand choice, particularly with large displays. Prior experiences with touch interfaces where a single touch with either hand moves an element may also be a contributing factor.

**Group Sheets**: The least preferred technique and the only one that participants reported requiring extra steps to perform was group sheets. Participants felt this technique required more time to learn and understand than the others. Adding too many sheets led

to confusion due the visual complexity induced by many overlapping sheets. Only a few participants used group sheets in the final mood board design stage.

**Overall Reactions**: Participants found our techniques easy to learn and use. The simplest technique, single finger positioning, was almost unanimously the most favored. The techniques involving gestures with different orientations of the hand were hardest for participants to perform, but not remember. We suspect the uniqueness of the gestures made them easier to remember.

## VI.5 Discussion

### VI.5.1 Ubiquitous Wearable Devices for Context

The growing popularity of personal wearables provides emerging avenues for engaging Mark Weiser's ubiquitous computing vision of interconnected wireless devices [Weiser 1991]. Because they exist in a society of interconnected devices, wearables have great potential to shape our interactions with other form-factors, including direct-touch interaction with large displays. It is important to note that a solution based on wearables is truly mobile and personal: wearables move about with users, and remain under users' direct control. Wearables—by their very nature owned by and identified with a specific user—can obviate the identity gap and provide context for lateral preference by distinguishing preferred vs. non-preferred hands.

When a user is uniquely identified, they could walk up to a public display and receive personalized experiences [Vogel and Balakrishnan 2004]. For example, in an office, a user could simply approach an electronic whiteboard and touch it. We know the user's information from the wearable, and the system could immediately bring up their personal workspace with the last ideas they were working on.

### VI.5.2 Self-Revelation of Context-Augmented Touch Interaction

A key challenge in gestural interfaces is how users learn what interactions are possible and how to perform associated gestures. Through self-revealing gestures [Wigdor and Wixon 2011], we could afford the possible interactions via feedforward techniques [e.g. Bau and Mackay 2008; Freeman et al. 2009; Ghomi et al. 2013] that provide revelation of the interactions possible, present dynamic guides within the context of interaction, and require physical rehearsal of the gestures [Kurtenbach et al. 1994]. With wearables, we could identify the user and adjust feedforward and feedback mechanisms accordingly based upon prior experience interacting with a system. We could scaffold learning [Guzdial 1994], requiring less revelation and guidance as the user becomes more experienced.

### VI.5.3 Revisiting Asymmetric vs. Symmetric Interaction

The main focus of our investigation was on designing bimanual-asymmetric interactions with direct touch that adhere to Guiard's principles. Yet, our goal is not to favor asymmetric over symmetric, but rather to open up a variety of appropriate and consistent mappings. These include both symmetric and asymmetric mappings, as well as the interleaving of the two, as afforded by the additional context our system provides. Affinity and MoodBoard illustrate two contrasting examples of how to achieve this, without ambiguity for bimanual touch input, and without compromising or restricting the interface's design choices.

### VI.5.4 Flexible Assignment of Roles to the Hands

While our interactions use strict hand assignments to adhere to Guiard's principles, it is not our intention to argue that strict role assignments are always necessary or even desirable. But armed with additional context for each touch, our approach makes this an intentional design choice rather than an inherent (and oft-unrecognized) ambiguity foisted

on the designer by the underlying sensing capabilities (or lack thereof). We seek to empower designers with choices for hand assignments in order to motivate new possibilities and considerations in bimanual interaction design.

## VI.6 Conclusion

We have shown how personal wearables—used to identify hands and users—enable phrasing together direct-touch interactions in a manner consistent with the bimanual-asymmetric assignment of roles to the hands as set forth by Guiard. Through our two prototypes, we develop bimanual techniques that demonstrate the interaction possibilities when ambiguity as to who does what (and with which hand) is removed. Our work contributes insights for designing bimanual interactions with lateral preference that has been lacking in standard direct-touch interaction. While we have illustrated several possibilities for augmenting touch with wearables, we hope and anticipate that other possible uses of this input technology may emerge, for touch and beyond.

# CHAPTER VII

# CONCLUSIONS

This research investigates how forms of bimanual interaction, which take advantage of the affordances of pen and multi-touch input, can effectively support visual designers, in early-stage ideation, who need to transition between different transformation operations. To perform this investigation, we explored a range both of user interface software and technologies and of evaluations. We developed Emmâ, a new pen+touch diagramming environment to support early-stage design ideation. Emmâ allows designers to gather, arrange, and sketch amidst their ideas in a free-form, zoomable space. In a landscape architecture design studio, we conducted a qualitative, hybrid investigation of Emmâ, combining aspects both of laboratory and field studies. To address particular user needs involving layering, we developed LayerFish, a bimanual interaction technique for ordering and manipulating overlapping content in large 2D design spaces, and evaluated user layering performance with a controlled experiment. We augmented touch interaction on large displays with wearables to support identifying the hands and users. From our investigation as a whole, we derived a set of methods that support phrasing together the various modal operations required in visual design. To conclude, we reiterate findings and discuss lessons learned. We further expand on the role of tension in phrasing, describe alternate methods for phrasing beyond the modality of pen+touch, and discuss how embodied interaction design is a form of choreography.

## VII.1 Phrasing with Quasimodes, Transient Interfaces, and Contextual Operations

Phrasing uses tension to tie together a series of inputs as a unit of interaction. The objective of phrasing is to help users—typically novices—abstract a series of inputs, that perform a sequence of operations, as a compound task [Buxton 1986]. This abstraction is hypothesized to help novice users function like experts, engaging in higher level tasks

(e.g., visually composing elements together, including arranging them and adjusting an element's opacity), rather than the required series of input and sub-tasks needed to perform the high level tasks (e.g., select an element, activate an inspector view on the side, find the opacity property, drag the opacity slider, examine result, drag the slider again, repeat). We identify three different techniques that support phrasing bimanual interactions: quasimodes, transient interfaces, and contextual operations.

### VII.1.1 Quasimodes (Space)

Quasimodes activate temporary global states, in which all input acts within the context of a specific mode. We use touches on the edge-constrained Hotpad to activate different quasimodes. These touches provide kinesthetic tension to phrase the quasimodes. The muscular effort of holding a touch in contact with the interactive surface creates this tension. The arm must be slightly extended and sustained in the air while the touch makes contact. Tactile feedback from the finger pressing against the surface adds to this tension. Lifting all the touches (and possibly returning the arm to its resting state as a result) removes the tension, ending the phrase and the associated quasimode.

An edge-constrained area choreographs gestural interaction, such that the left hand must move to a specific *space* on the display. We observed participants engaging effectively with the Hotpad, particularly when they worked near the bottom-left quadrant of the display. By using an edge-constrained area to activate quasimodes, we keep the left hand from occluding elements and interfering with actions of the right hand. The edge-constrained area is quickly accessible in the bottom-left corner, similar to why a menu bar is located at the top of the display. However, unlike a menu bar, the movement of the left hand will not stop as hits the bottom edge of the display, while the movement of a cursor stops at the top of the display.

We suspect further benefit is to be gained when we can design physical constraints that

help users find the edge of the display without having to look. An alternative would be to attach a touch sensitive area to the back of the display that functions as the Hotpad. Then, the physical boundaries of the display would serve as tactile and kinesthetic feedback, allowing the user to access the Hotpad without directing her visual attention away from the task at hand. This approach might also be useful in tablet settings, where the left hand is already holding the device.

Alternative methods for activating quasimodes include physical buttons and in-air gestures. Pressing and holding physical buttons could activate and phrase quasimodes. The keyboard has long been used as a mode switcher in WIMP interfaces; however, in pen+touch post-WIMP interfaces, a physical keyboard is not typically present. While displays, such as the Wacom Cintiq, have buttons located on the sides of the display, many other displays do not, such as the Microsoft Surface products. Still, physical buttons have tactile qualities, which are useful for interaction design [Hoggan et al. 2008; Ishii and Ullmer 1997; Lee et al. 2004; Schiphorst 2009]. Depressing a button requires more effort than touching a surface. Thus, kinesthetic tension seems stronger with physical buttons. Further investigation is needed to examine what impact, if any, that increased tension has on phrasing.

In-air gestures above the surface could be used to activate quasimodes anywhere over the display. This approach would allow users to activate quasimodes anywhere on the screen, rather than near an edge-constrained region, such as the Hotpad. However, it requires additional sensing of the hands above the surface, and may be prone to false activation, if in-air gestures are not sufficiently distinct. Also, the tension of maintaining an in-air gesture may be fatiguing for users. Choreographing in-air gestures followed by a touch—as an air+touch technique [Chen et al. 2014b]—could address these activation and fatigue issues. The touch conveys intention in the gesture that preceded it, and holding a touch on a display should be less fatiguing that holding a posture in-air. Using a touch

reintroduces the occlusion and interference issues that Hotpad sought to avoid. Chen et al. [2014b] supported quick mode switching with in-air gestures before touch and in-air gestures after touch. As presented by Wilkinson et al. [2016], the periods before and after a touch can serve to express intention (before)—activating a quasimode—and follow-up or recovery (after)—adjusting additional parameters or reverting changes made with the touch.

### VII.1.2 Transient Interfaces (Shape)

Transient interfaces are interactive controls that appear when activated, and hide when not [Webb and Kerne 2008]. Transient interfaces are activated as needed by a user to invoke commands, such as changing ink styling or discarding a set of wet ink strokes. As opposed to quasimodes, which restrict all input to a specific mode, transient interfaces only restrict input on the presented controls. Input performed outside the transient interface uses the default mode. For example, when the Ink Palette is activated, pen input on the transient interfaces changes the ink style, while pen input anywhere else produces ink.

We used two forms of activation for transient interfaces: gesture-based and timeouts. We choreograph gestures with unique *shapes* to activate transient interfaces. The shapes function as choreography, irregardless of *space* where the gestures are performed. Users may perform these gestures anywhere on the display, unlike contextual operations, which must be performed in specific *space*. In Emmâ and MoodBoard, an Ink Palette for adjusting ink styles is activated with a left-hand gesture. Lifting the gesture ends the phrase and deactivates the Ink Palette. ChopZ and Group Sheets are all activated and phrased with left-hand gestures. Through requiring the user to hold the gesture, we create kinesthetic tension to phrase the interaction.

Gesture-based activation raises interaction design issues in terms of visibility and learnability. How do users know what gestures to perform? For our evaluations in lab-

oratory settings, we showed participants how to perform the gestures. For other settings, one solution is to employ self-revealing techniques, such as feedforward approaches [e.g. Bau and Mackay 2008], where a touch provides visual feedback about where else the user can touch to perform different gestures. We can tailor feedforward representations based upon what interactive elements a user is touching. For example, if the user is touching an element in the diagramming canvas, then we can present cues for adding another touch to duplicate the element or add a pen press to perform perspective transform. If the user is touching an empty area of the canvas, then we can present cues for making a three-finger right-angle gesture to activate Ink Palette. Feedforward representations will need to use different visual affordances for each input modality, so that the user is aware of which modality to use.

In Emmâ, the Ink Menu is activated with a timeout after the user stops sketching. The Ink Menu is deactivated whenever the user starts sketching again or executes a command in the menu by pressing one of the buttons. The user does not control timeout activation. This can cause problems if a transient interface is activated and positioned exactly where a user is about to interact. Conversely, a timeout means the interface will always appear, which improves its visibility, and in the case of Ink Menu, helps remind participants what ink strokes are currently being grouped.

Previously, we proposed an alternative activation method for Ink Menu that occurs when the user tucks the pen. Tucking the pen allows the user to express their intent to stop sketching. Tucking the pen also provides kinesthetic tension as the pen is flipped and stowed. We suspect this will help phrase interaction with the Ink Menu. Hinckley et al. [2014] demonstrated modal switching among different operations based upon how the hand is holding the pen. We further extend this concept for phrasing transient interfaces.

### VII.1.3 Contextual Operations (Space+Shape)

Contextual operations are performed on a specific frame of reference—a *space* defined by user input—which could be an interactive element (e.g., for perspective transform) or a spatial area in proximity to the input, such as the area directly surrounding a touch used by the Straight Line Tool (see Chapter III, Section III.3.4). The same *shape* gesture may perform different contextual operations, depending on the *space* where the contextual operation is activated. A single left hand touch in Emmâ can be used to activate both the perspective transform command on an element and the straight line command on an empty area of the canvas. Differing from the choreography used for quasimodes with edge-constrained *spaces*, the choreography for contextual operations is more dynamic and dependent on the user's content, as the *spaces* change and move.

In Emmâ, contextual operations create or modify elements. Examples of contextual operations include duplicate, straight line, and perspective transform. In WIMP interfaces, commands activated via a right-click pop-up menu are contextual operations. Both quasimodes and contextual operations can involve interacting with elements. Yet unlike quasimodes, where input anywhere on the surface performs the mode's command, inputs for contextual operations are spatially constrained to the bounds of the reference frames being acted upon. For example, a user places two left-hand touches to duplicate an element. She then touches another element with her right hand. The right-hand touch will not duplicate. It will instead translate this other element. Unlike transient interfaces, contextual operations forgo interactive widgets (e.g. sliders) for adjusting parameters, instead employing more directness and embodiment through gestural interaction.

When performing a perspective transform, the user presses the pen on different areas of the element to adjust the rotational angle. The change in angle is mapped to pen pressure and position, rather than dragging a slider handle. For perspective transforms, the touch

not only phrases the interaction, but also defines the pivot point on which to perform the 3D perspective transform. The harder and longer the pen is pressed, the greater the perspective rotation. Additional pen presses are used to tweak and modify the transformation. Our goal was not to create a precise tool, but one that supports exploration in an embodied approach.

Creativity support tools, such as Emmâ, need to support exploration [Shneiderman et al. 2006]. We suspect that together, the visual manipulations of edge blending, translucence, and perspective transform, can support formation of emergent ideas through conceptual combination [Finke et al. 1992], even though we have not yet observed this by study participants. Conceptual combination is when two or more concepts are cognitively synthesized into a composite idea. These manipulations directly support visual blending of elements. In free-form web curation, elements represent gathered concepts. Thus, the visual blending of elements can serve as external cognition [Scaife and Rogers 1996], in which gathered concepts are visually blended to support conceptual combinations. In design fixation, a designer becomes stuck in a mental rut, unable to consider solutions beyond a limited set of ideas [Jansson and Smith 1991]. Playful exploration through visual blending could create provocative stimuli [Kerne et al. 2014; Shah 1998] to help overcome design fixation.

For the duplicate operation, the two left-hand touches phrase the interaction, but do not define any parameters. Using pen or touch with the right hand, a press and drag copies the element. The position of the new element can be adjusted by continuing the drag. This operation can be performed repeatedly with successive right-hand drags.

A problem with using left-hand touches for activation and phrasing in contextual operations is that the left-hand may occlude or interfere with the right hand's actions. Lifting the left-hand touches after the right hand begins to interact could solve this problem. However, it would remove the phrased tension created by holding the left-hand touches down,

and it would prevent repeated performance of the operations without placing the left-hand touches down again. Phrasing issues would arise as the user prematurely lifted the left hand before the right hand acted. We observed design studio participants performing duplicate and perspective operations multiple times for a single left-hand activation. Thus, it would seem more efficient to maintain the tension of the phrase, without requiring multiple activations.

One possible solution for this interference issue is to allow the left-hand touches to be moved outside the spatial constraints of the contextual operation—the element's boundaries. Visual feedback is needed to convey that a phrase is being maintained despite the touches not residing over the element. For perspective transform, in which a left-hand touch defines a parameter of the operation, we need to separate touch moves to redefine the pivot point, versus those to avoid interference. We recommend using pen tucking to phrase these different actions. When the pen is not tucked, the user is performing perspective transform, and left-hand touch moves define changes in the pivot point. When the pen is tucked, the user is not performing perspective transform, and the left-hand touch can be moved outside the element boundaries.

We hypothesize that this phrasing helps users abstract a repeated sequence of operations as a whole structural unit. A primary difference between expert and novice users is their ability to abstract a series of operations as a higher-level unit, allowing them to focus on the bigger picture rather than how to perform specific actions [Buxton 1986]. In language, people use phrasing to connect words, as smaller units, to express ideas in human-to-human dialogues. Similarly, in gestural interfaces, phrases of human movement provide units of expression to invoke commands and manipulate parameters in human-to-computer dialogues. Linguistic rules help people form phrases to effectively communicate meaning, while still supporting individual expression. Embodied interaction designers need to develop methods that help users construct phrases to communicate meaning to computing

systems, while still supporting expressive differences. This research has demonstrated several such methods, using kinesthetic tension as a mechanism for phrasing.

## VII.2 Tension

We adhered to Guiard's principles in our design of bimanual-asymmetric interactions. As a result, the tension is also asymmetric. The left and right hands experience different levels of tension. In choreography, a phrase is defined by its tension and the associated high points that are created from that tension [Blom and Chaplin 1982]. High points in the tension indicate key moments. This is also true for the interaction design, where visual feedback is needed to help the user learn and recognize those key moments. Beyond kinesthetic tension, visual tension plays an important role in our interaction design. By differing tension, high points, and feedback, we could support designing interactions that do not adhere to Guiard's principles, including bimanual-symmetric interactions.

### VII.2.1 Left Hand

For all our bimanual-asymmetric interactions, the left hand performs the following sequence of actions. The left hand acts first, performing a gesture to begin the phrase. The left hand maintains the gesture, creating sustained tension. Lifting the gesture ends the phrase. Following Guiard's principle of Right-to-Left Spatial Reference, the left hand defines the context of right-hand actions. The left hand can also adjust parameters for an operation, as in the pivot point for perspective transforms in Emmâ or the left selection bounds for ChopZ in MoodBoard. The left hand is still free to move, the phrase's tension is maintained as long as the initial touch contact remains. By making one hand—the left—define the context of interaction by sustained tension, we seek to help the user phrase a series of inputs, representing a compound task as a structural unit.

One design consideration for this approach is what to do if the left hand is lifted while the right hand is acting. This could be an accident or an intentional action. For accidents,

the user should be able to immediately reapply the left-touch to continue the phrase. Using a timeout, we can wait to see the touch is reapplied within a short window. Intentionally lifting the left hand should end the phrase. The question then becomes, what was the user's intent? Is the user's intent to cancel the current operation and return to the previous state? Or, is the user's intent to keep the current changes?

Matulic and Norrie [2013], in their document editing application, canceled quasimodes for document editing when the left hand was lifted before the pen (in the right hand) finished an operation. We consider this approach to canceling helpful and efficient in most cases. On many occasions, we observed participants using undo to immediately revert the effects of just-completed operation. Canceling saves an extra undo step required to revert the incorrect change that was made. However, this approach could result in problems when the phrasing left hand is lifted at nearly the same time as the right-hand action is ending. The input device(s) may sense one order in which actions were ended (e.g., right then left), while the user perceives the actions ending in a different order (e.g. left then right). The user's intent is unclear. She could be in the process of performing several different operations quickly and accidentally lifted her left hand first. In this case, the changes should stay. Or, she could have adjusted the wrong parameter for an operation and immediately lifted both hands in response. In this case, the operation should be canceled, and the changes removed. The appropriate method for determining intent in these edge case may vary by operation or type of operation. Operations that are often performed together in quick succession, should cancel in this case. Other operations should not. Interaction designers would benefit from considering the choreographic phrases of movements for how users might sequence embodied interactions (see Section VII.5).

### VII.2.2 Right Hand

The right hand performs operations with fluctuating levels of tension as the pen or touches are pressed against, moved across, and lifted from the interactive surface. The movements of the right hand modify parameters of operations. Right hand actions may involve repeated contacts with the surface as parameters are manipulated multiple times. In visual design, exploratory processes can support ideation. We hypothesize that this ability to repeatedly manipulate parameters without needing to reactivate commands will aid this exploration. Additionally, choreographic phrasing of these repeated manipulations should help novices function like experts, abstracting a series of inputs as a single compound task [Buxton 1986].

### VII.2.3 High Points and Feedback

We consider the point when operations are performed as the primary high point in a interaction phrase. This high point may be quick and sudden, such as when deleting an element, or sustained, such as when adjusting the opacity of an element. Other high points exist, such as performing a gesture with the left hand to initiate a phrase.

Associated with these high points should be feedback to convey changes in state that occur when these high points are reached [Webb et al. 2006]. We use visual feedback on the Hotpad to indicate activation of different quasimodes as the number of left-hand touches changes. A further improvement of this feedback would be to provide in-place feedback, such as a cursor change, so that the user does not need to look at the Hotpad. Transient interfaces provide visual feedback through their appearance upon activation. Contextual operations can provide feedback by overlaying visual information on the context element or modifying that element in some way. In Emmâ, we present a slightly offset, translucent copy of an element when the user activates the duplicate operation by placing two fingers on an element. A touch drag on this translucent copy, allows the user

to position the duplicate. The copy remains relatively positioned under the user's touch drag. Lifting up the touch, drops the copy in place and removes its translucence.

### VII.2.4 Visual Tension

We employ kinesthetic, rather than visual, tension as the primary form for phrasing. Prior work has demonstrated the benefits of kinesthetic tension [Sellen et al. 1992]. We also create tension through visual sensory experiences. For example, ChopZ employs visual tension (as well as kinesthetic), where elements in a selected area are rotated in 3D space as if looking at that area from the side of the display. The visual distortion of the space creates tension that remains while ChopZ is activated, and is removed when chop gesture is lifted. Since ChopZ has visual tension, phrasing still remains even if we were to remove the kinesthetic tension. We suspect that with only visual tension, the user will experience performing ChopZ not as a single compound task, but as several separate tasks, such as activation, selection, layering, and deactivation.

### VII.2.5 Breaking from Guiard

We made the design decision to adhere to Guiard's principles in our bimanual interaction design. We favor Guiard's principles to take advantage of human familiarity with bimanual activity that takes place everyday in the physical world. However, we understand that interaction designers may not always wish to follow Guiard's principles, to sometimes design bimanual interactions where the right hand initiates action and defines the frame of reference for the left. Here, we discuss a bit about what those interactions might look like and how they might be combined with those that are Guiard-abiding.

One of the main design considerations when breaking from Guiard is whether each hand initiates different operations or the same operations. For example, in MoodBoard, a two-finger gesture with the left hand activates the Straightedge Tool. That same gesture with the right hand, could also activate the Straightedge Tool or an entirely different

operation. A user is unlikely to hold the pen in her non-preferred hand, so having the Straightedge Tool activate for both hands seems unhelpful. Instead, the right hand could activate a visual clipboard with a collection of images for adding to the mood board. Then, the left hand could drag an image—a coarse action that does not require the pen—from the visual clipboard into the mood board space.

If we choose to assign different operations to each hand, then another design consideration is what operations get assigned to which hand. The choices for assigning operations to hand should consider hand dominance. Operations that require greater precision should follow Guiard's principles, allowing the right hand to precisely adjust parameters—possibly with the pen. More coarse operations can use the left hand for adjusting parameters.

Alternatively, an interaction designer could use the semantics of operations to assign hands. For example, using the left hand for sketching operations and the right hand for gathering and arranging operations. This matches with our example above for MoodBoard, where the Straightedge Tool, a primarily sketching operation, is mapped to the left hand, and the visual clipboard is mapped to the right hand. We suspect that semantic mappings may help the user better learn and remember which hand to use when performing an operation.

### VII.2.6 Symmetric Interactions

We primarily investigate bimanual techniques with asymmetric assignment of roles to the hands. Our asymmetric assignments use tension of the left hand to phrase interactions. When employing symmetric assignments, an interaction designer cannot phrase interactions in the same way. Since both hands serve the same role, the user can initiate interaction with either hand or both simultaneously, such as when the landscape architecture students zoomed the diagramming canvas in Emmâ with two hands. The kinesthetic

tension is still present. A phrase is then defined and sustained by the combined tensions of both hands pressing touches against the interactive surface.

Interaction designers also have the choice of intermixing asymmetric and symmetric bimanual interactions. For example, Group Sheets in MoodBoard begins with a left-hand gesture to phrase the interaction, presenting a visible sheet. Next, the user can add a right-hand touch to adjust the top-right corner of the sheet. The left hand can also be moved to adjust the sheet's bottom-left corner. The two hands operate together (symmetrically) to define the bounds of the sheet. By lifting the right-hand touch, the user can return to an asymmetric mapping, allowing the right hand to tap elements to add to the sheet. Lifting the left hand at any point in this sequence, ends the phrase and the Group Sheets operation.

### VII.3 Alternative Embodied Modalities to Support Phrasing

Our investigation explored phrasing for pen+touch and multi-touch interaction modalities. These are not the only modalities that support phrasing using the methods we identified. Other embodied modalities for post-WIMP interfaces include in-air hand gestures, tangibles, whole-body, wearables, and game controller. We suspect that the methods we identified will apply to these other modalities, but there also likely exist other methods specific to them. Here, we look at in-air hand gestures and tangibles as alternative interaction modalities for phrasing.

#### VII.3.1 In-Air Hand Gestures

The in-air modality involves gesturing in the air—often near an interactive surface—to invoke commands and manipulate parameters of those commands. We observed that participants would hover their hands above the surface to support efficient modal switching. This suggests that interaction designers should consider in-air gestures and the positions of the hands for phrasing. An advantage of the in-air modality for interactive surfaces is that it takes the hands off the surface, preventing interference and occlusions issues.

Guiard's principles still apply here. The left hand should generally still define the frame of reference for right-hand actions. Phrasing should begin with left hand gestures and end with a release of tension from the left hand.

This leads to one essential design question: how can we choreograph the left hand sustaining and releasing tension after performing a gesture? We previously mentioned how air+touch could be used to support phrasing in contextual operations, while avoiding potential false-positives in recognition. With air+touch, a left hand gesture would be followed by a touch, which would sustain tension and end the phrase when lifted. However, air+touch will not work when the interactive surface is not touchable—perhaps because it is a public display behind glass or it lacks touch sensing capabilities.

For alternatives to air+touch, we refer back to Laban's analysis of effort in human movement, in particular his discussion of weight and time [Laban and Lawrence 1947]. Time of a movement is either quick or sustained. The gesture to activate a command is quick, while the effort to maintain the phrase is sustained. The release of tension at the end is also quick. The weight of a movement conveys intention. It can be either strong or light. Strong movements tend to be directed downward, as the weight of the movement against gravity pulls it that way. Conversely, light movements tend to be directed upward.

We can apply Laban's concepts when choreographing in-air phrases. The left hand first gestures in a quick movement. This gesture is followed with a downward pull towards the ground or the user's body—a strong effort—perhaps also involving closing the hand. This downward pull is sustained at its lowest point, maintaining the phrase. The right hand then gestures to manipulate parameters. After the right hand finishes, the left hand makes a quick, light movement upward—opening a closed hand if made—releasing the tension and ending the phrase. Games for the Microsoft Xbox with a Kinect have use closed hands to phrase interactions with menus. We further extend this idea, proposing choreographic principles in the design of in-air interactions.

### VII.3.2 Tangibles

Tangible user interfaces (TUIs) map physical representations to virtual controls [Shaer and Hornecker 2010]. They give the user something to grasp while manipulating parameters, providing a physical affordance that is richer than virtual ones [Fitzmaurice et al. 1995]. Grasping a tangible provides kinesthetic tension, which could be used to phrase interaction. Interaction designers can use different shapes and kinds of tangibles to support different types of operations [Mazalek et al. 2002]. We suspect that the tactile qualities and weight of different shaped tangibles could help reinforce users' mental models of phrased interactions. As physical artifacts, tangibles can be stacked [Baudisch et al. 2010], connected together [Grote et al. 2015; Merrill et al. 2012], or spatially arranged [Jordà et al. 2007] on an interactive surface. We can design interactive sequences through combining different tangibles—each phrasing an operation—to perform complex compound tasks.

Tangibles can be used in conjunction with other forms of input, such as pen or multitouch, and our phrasing methods. For quasimodes, a modal state can be temporarily activated when a tangible is placed on the surface. The mode is deactivated when the tangible is removed. If the hand that places the tangible maintains its grasp after placing, then kinesthetic tension can be sustained to support phrasing. However, a user may remove their grasp from the tangible without lifting it off the surface. We suspect that this lost of tension could cause modal switching errors, as the user forgets that the tangible is placed. This issue is similar to problems with using keyboard shortcuts for modal switching [Sellen et al. 1992]. Tangibles can support transient interfaces. Recently, Microsoft presented the Surface Dial[1], a tangible device that can specify modal states and adjust operational parameters, such as ink color. They demonstrated a user drawing with a pen in the right hand, while simultaneously changing the color by rotating the dial with the left hand. For

[1] https://www.microsoft.com/en-us/surface/accessories/surface-dial

contextual operations, placing a tangible on an element could activate the operation, while input with pen, touch, or another tangible with the other hand could manipulate parameters of that operation.

### VII.4 Beyond Video Observations: Capturing Human Movement Data

Embodied interaction design involves considering the role of the body as it engages with computing systems [Dourish 2001]. For pen+touch environments, the hands are the primary parts of the body that directly interact with a surface, either by making contact with fingers or using a pen. We observed interesting phenomena with regards to the positions and movements of the hands above the surface. This indicates that evaluations of embodied interaction design should include observing positions and movements of relevant parts of the body on and around surfaces. We used video cameras (from multiple angles) to capture this data in our embodied process recordings. However, other methods for sensing human movement, such as depth cameras and wearables, have potential value for evaluation, as they capture other aspects not possible with video.

#### VII.4.1 3D Sensing: Depth Cameras

Sensors containing depth cameras, such as the Microsoft Kinect, can capture 3D recordings of body movements. These 3D recordings provide quantitative data that can be analyzed programmatically without need for a human to watch and code many hours of video footage. For example, Mentis and Johansson [2013] used Kinect sensor recordings to computationally perform Laban Movement Analysis on participants' whole-body movements. They compared the results of the computational analysis with what participants and a LMA expert perceived. While 3D sensing has been used extensively to design new interaction techniques around surfaces, we have not seen its use for evaluating techniques, in particular those involving pen+touch or multi-touch modalities.

By capturing 3D positional data of the hands and syncing that with logged interaction

data, we can quantitatively analyze and compare hand positions and movements for different interaction phrases or sequences. This analysis could identify differences in the quality and quantity of movement among participants. We can create heat map representations of the data to show where the hands are mostly located, similar to Benko et al. [2008], but not limited only to contact points on the interactive surface. For our field study, these heat maps could have shown the hover state of G2, where she kept her left hand hovering near the Hotpad, fluidly modal switching between, sketching, ink styling, and undo-redo. These heat maps could also point out interaction issues, such as repeated long movements by one hand to access a poorly-positioned control.

### VII.4.2 Wearables: Inertial Measurement Unit (IMU)

We used wearables with IMUs to identify which hand and user is touching a large display (see Chapter VI). However, wearables could also provide human movement sensing data for observation in evaluating interaction techniques and systems. Silang Maranan et al. [2014] used a wearable with IMU and machine learning to perform LMA. IMUs provide acceleration and orientation data, which if located on the hand or wrist, could provide measures of the quantity and quality of movement. If we combine IMU data with touch input [Rofouei et al. 2012], we can sync logged interaction data with inertial sensing. We could then perform analysis of movement based upon free-form web curation processes, similar to our investigation in design studio education (see Chapter IV), but without requiring human researchers to watch and transcribe hours of video.

## VII.5 Choreography as Interaction Design

We use the term, "phrasing," to describe how a series of inputs can be tied together to form a whole interaction. Choreographers create phrases as unit structures for developing dance performances, where a series of movements are tied together to form a whole expression. We argue that embodied interaction design involves choreographic processes.

The designer specifies how the user should move their body to interact with a system. While choreographic principles have been applied in the design of whole-body interactions [Alaoui et al. 2012; Cuykendall et al. 2016; Loke et al. 2005, 2007], little to no investigation has been performed for choreography of interactions around surfaces involving the hands.

Interactive environments offer a multiplicity of interaction possibilities. It is important for interaction designers, like choreographers, to consider how actions will be combined into phrases and how phrases will be combined into larger sequences. We used the Hotpad as a chorded gesture region, where transitioning among quasimodes simply requires adding or removing touches. Repeated selection and translucence operations can be combined without lifting the left hand off the Hotpad. We used a transient interface for the Ink Palette so that it could be active while the user sketches, allowing quick access when needing to use different ink styles. We used a contextual operation for Duplicate, so that users could create and position multiple copies in a single phrase.

Laban and Lawrence [1947] described qualities of human movement—weight, space, and time—in relation to physical effort. These qualities have been used in HCI research in various ways, including as interaction modalities [Alaoui et al. 2012], for evaluation [Silang Maranan et al. 2014], and for design of movement-based interactions [Webb et al. 2006]. A notational system derived from Laban's work—Labanotation—could be used as a tool for designing movement-based interaction [Loke et al. 2005]. We suspect that Laban's ideas and associated Labanotation could support designing pen+touch interactions. A common goal in interaction design is to build techniques that are easy and quick to use. Ease of use is directly related to aspects of physical effort. In examining our own interaction design with regards to Laban, we used sustained, heavy kinesthetic tension to phrase interactions. Tension is released through quick, light movements, upward off the interactive display.

### VII.6 Phrasing in a Post-WIMP World

We have presented methods for phrasing bimanual interactions to support visual design. We view phrasing as a valuable approach for interaction design in general, but specifically embodied interaction design for post-WIMP interfaces. We are no longer reliant on the single pointer paradigm. We can define interactions through gestures involving multiple points of contact on an interactive surface. However, HCI continues to encounter mode error and confusion challenges, when gestures activate, but do not phrase, operational states. By using kinesthetic tension through sustaining gestures with the left hand, we provide a methodology for choreographic phrases to deal with modal switching issues in post-WIMP interfaces. We imagine the emergence of new forms of phrasing in the future, as choreographic approaches are applied in conjunction with improving methods of human movement sensing.

## REFERENCES


Adobe Illustrator (2016). http://www.adobe.com/products/illustrator.

Agarawala, A. and Balakrishnan, R. (2006). Keepin' it real: Pushing the desktop metaphor with physics, piles and the pen. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 1283–1292. ACM. http://doi.acm.org/10.1145/1124772.1124965.

Akers, D., Jeffries, R., Simpson, M., and Winograd, T. (2012). Backtracking events as indicators of usability problems in creation-oriented applications. *ACM Transactions on Computer-Human Interaction*, 19(2):16:1–16:40.

Alaoui, S. F., Caramiaux, B., Serrano, M., and Bevilacqua, F. (2012). Movement qualities as interaction modality. In *Proceedings of the Designing Interactive Systems Conference*, DIS '12, pages 761–769. ACM. http://doi.acm.org/10.1145/2317956.2318071.

Amabile, T. M. (1985). Motivation and creativity: Effects of motivational orientation on creative writers. *Journal of Personality and Social Psychology*, 48(2):393.

Annett, M., Grossman, T., Wigdor, D., and Fitzmaurice, G. (2011). Medusa: A proximity-aware multi-touch tabletop. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 337–346. ACM. http://doi.acm.org/10.1145/2047196.2047240.

Antle, A. N., Bevans, A., Tanenbaum, J., Seaborn, K., and Wang, S. (2011). Futura: Design for collaborative learning and game play on a multi-touch digital tabletop. In *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '11, pages 93–100. ACM. http://doi.acm.org/10.1145/1935701.1935721.

Apitz, G. and Guimbretière, F. (2004). Crossy: A crossing-based drawing application. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, pages 3–12. ACM. http://doi.acm.org/10.1145/1029632.1029635.

Apple Watch (2016). http://www.apple.com/watch.

Ardito, C., Buono, P., Costabile, M. F., and Desolda, G. (2015). Interaction with large displays: A survey. *ACM Computing Surveys*, 47(3):46:1–46:38.

Aristides, J. (2006). *The Classical Drawing Atelier: A Contemporary Guide to Traditional Studio Practice*. Watson-Guptill Publications, New York, US, USA.

Arnheim, R. (1969). *Visual Thinking*. University of California Press, Berkeley, CA, USA.

Augsten, T., Kaefer, K., Meusel, R., Fetzer, C., Kanitz, D., Stoff, T., Becker, T., Holz, C., and Baudisch, P. (2010). Multitoe: High-precision interaction with back-projected floors based on high-resolution multi-touch input. In *Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 209–218. ACM.
http://doi.acm.org/10.1145/1866029.1866064.

Azad, A., Ruiz, J., Vogel, D., Hancock, M., and Lank, E. (2012). Territoriality and behaviour on and around large vertical publicly-shared displays. In *Proceedings of the Designing Interactive Systems Conference*, DIS '12, pages 468–477. ACM.
http://doi.acm.org/10.1145/2317956.2318025.

Bailly, G., Lecolinet, E., and Guiard, Y. (2010). Finger-count & radial-stroke shortcuts: 2 techniques for augmenting linear menus on multi-touch surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 591–594. ACM.
http://doi.acm.org/10.1145/1753326.1753414.

Bailly, G., Lecolinet, E., and Nigay, L. (2008). Flower menus: A new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '08, pages 15–22. ACM.
http://doi.acm.org/10.1145/1385569.1385575.

Balakrishnan, R. and Hinckley, K. (2000). Symmetric bimanual interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, pages 33–40. ACM.
http://doi.acm.org/10.1145/332040.332404.

Bau, O. and Mackay, W. E. (2008). Octopocus: A dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 37–46. ACM.
http://doi.acm.org/10.1145/1449715.1449724.

Baudisch, P., Becker, T., and Rudeck, F. (2010). Lumino: Tangible blocks for tabletop computers based on glass fiber bundles. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 1165–1174. ACM.
http://doi.acm.org/10.1145/1753326.1753500.

Beaudouin-Lafon, M. (2000). Instrumental interaction: An interaction model for designing post-wimp user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, pages 446–453. ACM.
http://doi.acm.org/10.1145/332040.332473.

Beaudouin-Lafon, M. and Lassen, H. M. (2000). The architecture and implementation of cpn2000, a post-wimp graphical application. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, pages 181–190. ACM.
http://doi.acm.org/10.1145/354401.354761.

Bederson, B. B. (2000). Fisheye menus. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, pages 217–225. ACM.
http://doi.acm.org/10.1145/354401.354782.

Benko, H., Wilson, A. D., and Balakrishnan, R. (2008). Sphere: Multi-touch interactions on a spherical display. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 77–86. ACM.
http://doi.acm.org/10.1145/1449715.1449729.

Beyer, H. and Holtzblatt, K. (1997). *Contextual Design: Defining Customer-Centered Systems*. Morgan Kaufmann Publishers, San Francisco, CA, USA.

Bi, X., Grossman, T., Matejka, J., and Fitzmaurice, G. (2011). Magic desk: Bringing multi-touch surfaces into desktop work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2511–2520. ACM.
http://doi.acm.org/10.1145/1978942.1979309.

Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. (1993). Toolglass and magic lenses: The see-through interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 73–80. ACM.
http://doi.acm.org/10.1145/166117.166126.

Blažica, B., Vladušič, D., and Mladenić, D. (2013). Mti: A method for user identification for multitouch displays. *International Journal of Human-Computer Studies*, 71(6):691–702.

Blom, L. A. and Chaplin, L. T. (1982). *The Intimate Act of Choreography*. University of Pittsburgh Press, Pittsburgh, PA, USA.

Blumer, H. (1954). What is wrong with social theory? *American Sociological Review*, 19(1):3–10.

Bødker, S. (1991). *Through the Interface: A Human Activity Approach to User Interface Design*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.

Bragdon, A., Zeleznik, R., Williamson, B., Miller, T., and LaViola, Jr., J. J. (2009). Gesturebar: Improving the approachability of gesture-based interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages

2269–2278. ACM.
http://doi.acm.org/10.1145/1518701.1519050.

Brandl, P., Forlines, C., Wigdor, D., Haller, M., and Shen, C. (2008). Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '08, pages 154–161. ACM.
http://doi.acm.org/10.1145/1385569.1385595.

Brown, B., Reeves, S., and Sherwood, S. (2011). Into the wild: Challenges and opportunities for field trial methods. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 1657–1666. ACM.
http://doi.acm.org/10.1145/1978942.1979185.

Buxton, B. (2010). *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann, Burlington, MA, USA.

Buxton, W. (1986). Chunking and phrasing and the design of human-computer dialogues. In *Proceedings of the IFIP World Computer Congress*, pages 475–480.

Buxton, W., Fitzmaurice, G., Balakrishnan, R., and Kurtenbach, G. (2000). Large displays in automotive design. *IEEE Computer Graphics and Applications*, 20(4):68–75.

Cage, J. (1961). Indeterminacy. In *Silence: Lectures and Writings*. Wesleyan University Press, Middletown, CT, USA.

Callahan, J., Hopkins, D., Weiser, M., and Shneiderman, B. (1988). An empirical comparison of pie vs. linear menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '88, pages 95–100. ACM.
http://doi.acm.org/10.1145/57167.57182.

Cao, X., Wilson, A. D., Balakrishnan, R., Hinckley, K., and Hudson, S. E. (2008). Shapetouch: Leveraging contact shape on interactive surfaces. In *2008 3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems*, pages 129–136.

Carroll, E. A., Latulipe, C., Fung, R., and Terry, M. (2009). Creativity factor evaluation: Towards a standardized survey metric for creativity support. In *Proceedings of the Seventh ACM Conference on Creativity and Cognition*, C&C '09, pages 127–136. ACM.
http://doi.acm.org/10.1145/1640233.1640255.

Charmaz, K. (2014). *Constructing Grounded Theory*. Introducing Qualitative Methods series. SAGE Publications, Los Angeles, CA, USA.

Chatty, S. (1994). Extending a graphical toolkit for two-handed interaction. In *Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology*, UIST '94, pages 195–204. ACM.
http://doi.acm.org/10.1145/192426.192500.

Chen, X. A., Grossman, T., Wigdor, D. J., and Fitzmaurice, G. (2014a). Duet: Exploring joint interactions on a smart phone and a smart watch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 159–168. ACM.
http://doi.acm.org/10.1145/2556288.2556955.

Chen, X. A., Schwarz, J., Harrison, C., Mankoff, J., and Hudson, S. E. (2014b). Air+touch: Interweaving touch & in-air gestures. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 519–525. ACM.
http://doi.acm.org/10.1145/2642918.2647392.

Cockburn, A. and Gin, A. (2006). Faster cascading menu selections with enlarged activation areas. In *Proceedings of Graphics Interface 2006*, GI '06, pages 65–71. Canadian Information Processing Society.
http://dl.acm.org/citation.cfm?id=1143079.1143091.

Cockburn, A., Gutwin, C., and Greenberg, S. (2007). A predictive model of menu performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 627–636. ACM.
http://doi.acm.org/10.1145/1240624.1240723.

Cockburn, A., Karlson, A., and Bederson, B. B. (2009). A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys*, 41(1):2:1–2:31.

Csikszentmihalyi, M. (2009). *Flow: The Psychology of Optimal Experience*. Harper Perennial Modern Classics. Harper Collins, New York, NY, USA.

Cuykendall, S., Soutar-Rau, E., Schiphorst, T., and DiPaola, S. (2016). If words could dance: Moving from body to data through kinesthetic evaluation. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*, DIS '16, pages 234–238. ACM.
http://doi.acm.org/10.1145/2901790.2901822.

Daly, S. R., Adams, R. S., and Bodner, G. M. (2012). What does it mean to design? a qualitative investigation of design professionals' experiences. *Journal of Engineering Education*, 101(2):187–219.

Davidson, P. L. and Han, J. Y. (2008). Extending 2d object arrangement with pressure-sensitive layering cues. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 87–90. ACM.
http://doi.acm.org/10.1145/1449715.1449730.

Deitch, J., Geiss, S., and Gruen, J. (2014). *Keith Haring*. Rizzoli International Publications, New York, NY, USA.

Deleuze, G. (1988). *Foucault*. University of Minnesota Press, Minneapolis, MN, USA.

Deleuze, G. and Guattari, F. (1987). *A Thousand Plateaus: Capitalism and Schizophrenia*. University of Minnesota Press, Minneapolis, MN, USA.

Dietz, P. and Leigh, D. (2001). Diamondtouch: A multi-user touch technology. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*, UIST '01, pages 219–226. ACM. http://doi.acm.org/10.1145/502348.502389.

Dondis, D. (1974). *A Primer of Visual Literacy*. MIT Press, Cambridge, MA, USA.

Dourish, P. (2001). *Where the Action Is: The Foundations of Embodied Interaction*. MIT Press, Cambridge, MA, USA.

Eichmann, P., Song, H., and Zgraggen, E. (2016). cted: Advancing selection mechanisms in web browsers. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '16, pages 3049–3055. ACM. http://doi.acm.org/10.1145/2851581.2892553.

Elmqvist, N., Vande Moere, A., Jetter, H.-C., Cernea, D., Reiterer, H., and Jankun-Kelly, T. J. (2011). Fluid interaction for information visualization. *Information Visualization*, 10(4):327–340.

Ewerling, P., Kulik, A., and Froehlich, B. (2012). Finger and hand detection for multi-touch interfaces based on maximally stable extremal regions. In *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces*, ITS '12, pages 173–182. ACM. http://doi.acm.org/10.1145/2396636.2396663.

Fass, A., Forlizzi, J., and Pausch, R. (2002). Messydesk and messyboard: Two designs inspired by the goal of improving human memory. In *Proceedings of the 4th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, DIS '02, pages 303–311. ACM. http://doi.acm.org/10.1145/778712.778754.

Fdili Alaoui, S., Carlson, K., Cuykendall, S., Bradley, K., Studd, K., and Schiphorst, T. (2015). How do experts observe movement? In *Proceedings of the 2Nd International Workshop on Movement and Computing*, MOCO '15, pages 84–91. ACM. http://doi.acm.org/10.1145/2790994.2791000.

Finke, R., Ward, T., and Smith, S. (1992). *Creative Cognition: Theory, Research, and Applications*. MIT Press, Cambridge, MA, USA.

Fitbit (2016). http://www.fitbit.com.

Fitzmaurice, G., Khan, A., Pieké, R., Buxton, B., and Kurtenbach, G. (2003). Tracking menus. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*, UIST '03, pages 71–79. ACM. http://doi.acm.org/10.1145/964696.964704.

Fitzmaurice, G., Matejka, J., Khan, A., Glueck, M., and Kurtenbach, G. (2008). Piecursor: Merging pointing and command selection for rapid in-place tool switching. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1361–1370. ACM. http://doi.acm.org/10.1145/1357054.1357268.

Fitzmaurice, G. W., Ishii, H., and Buxton, W. A. S. (1995). Bricks: Laying the foundations for graspable user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 442–449. ACM Press/Addison-Wesley Publishing Co. http://dx.doi.org/10.1145/223904.223964.

Forlines, C., Wigdor, D., Shen, C., and Balakrishnan, R. (2007). Direct-touch vs. mouse input for tabletop displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 647–656. ACM. http://doi.acm.org/10.1145/1240624.1240726.

Freeman, D., Benko, H., Morris, M. R., and Wigdor, D. (2009). Shadowguides: Visualizations for in-situ learning of multi-touch and whole-hand gestures. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 165–172. ACM. http://doi.acm.org/10.1145/1731903.1731935.

Frisch, M., Heydekorn, J., and Dachselt, R. (2009). Investigating multi-touch and pen gestures for diagram editing on interactive surfaces. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 149–156. ACM. http://doi.acm.org/10.1145/1731903.1731933.

Frisch, M., Langner, R., and Dachselt, R. (2011). Neat: A set of flexible tools and gestures for layout tasks on interactive displays. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '11, pages 1–10. ACM. http://doi.acm.org/10.1145/2076354.2076356.

Furnas, G. W. (1986). Generalized fisheye views. *SIGCHI Bull.*, 17(4):16–23.

Garner, S. and McDonagh-Philp, D. (2001). Problem interpretation and resolution via visual stimuli: the use of 'mood boards' in design education. *Journal of Art & Design Education*, 20(1):57–64.

Ghomi, E., Huot, S., Bau, O., Beaudouin-Lafon, M., and Mackay, W. E. (2013). Arpège: Learning multitouch chord gestures vocabularies. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces*, ITS '13, pages 209–218. ACM.
http://doi.acm.org/10.1145/2512349.2512795.

Gibson, J. J. (1979). *The Ecological Approach to Visual Perception*. Houghton Mifflin, Boston, MA, USA.

Glenberg, A. (1999). Why Mental Models Must Be Embodied. *Advances in Psychology*, 128:77–90.

Glenberg, A. M., Brown, M., and Levin, J. R. (2007). Enhancing comprehension in small reading groups using a manipulation strategy. *Contemporary Educational Psychology*, 32(3):389 – 399.

Goldschmidt, G. (1991). The Dialectics of Sketching. *Creativity Research Journal*, 4(2):123–143.

Greenberg, S., Marquardt, N., Ballendat, T., Diaz-Marino, R., and Wang, M. (2011). Proxemic interactions: The new ubicomp? *Interactions*, 18(1):42–50.

Grossman, T. and Balakrishnan, R. (2005). The bubble cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 281–290. ACM.
http://doi.acm.org/10.1145/1054972.1055012.

Grossman, T., Baudisch, P., and Hinckley, K. (2009). Handle flags: Efficient and flexible selections for inking applications. In *Proceedings of Graphics Interface 2009*, GI '09, pages 167–174. Canadian Information Processing Society.
http://dl.acm.org/citation.cfm?id=1555880.1555918.

Grossman, T., Dragicevic, P., and Balakrishnan, R. (2007). Strategies for accelerating online learning of hotkeys. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 1591–1600. ACM.
http://doi.acm.org/10.1145/1240624.1240865.

Grossman, T., Hinckley, K., Baudisch, P., Agrawala, M., and Balakrishnan, R. (2006). Hover widgets: Using the tracking state to extend the capabilities of pen-operated devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 861–870. ACM.
http://doi.acm.org/10.1145/1124772.1124898.

Grote, C., Segreto, E., Okerlund, J., Kincaid, R., and Shaer, O. (2015). Eugenie: Multitouch and tangible interaction for bio-design. In *Proceedings of the Ninth International*

*Conference on Tangible, Embedded, and Embodied Interaction*, TEI '15, pages 217–224. ACM.
http://doi.acm.org/10.1145/2677199.2680605.

Guiard, Y. (1987). Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of Motor Behavior*, 19(4):486–517.

Guimbretière, F., Stone, M., and Winograd, T. (2001). Fluid interaction with high-resolution wall-size displays. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*, UIST '01, pages 21–30. ACM.
http://doi.acm.org/10.1145/502348.502353.

Guimbretiére, F. and Winograd, T. (2000). Flowmenu: Combining command, text, and data entry. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, pages 213–216. ACM.
http://doi.acm.org/10.1145/354401.354778.

Gutwin, C. (2002). Improving focus targeting in interactive fisheye views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, pages 267–274. ACM.
http://doi.acm.org/10.1145/503376.503424.

Guzdial, M. (1994). Software-realized scaffolding to facilitate programming for science learning. *Interactive Learning Environments*, 4(1):001–044.

Hamilton, W., Kerne, A., and Robbins, T. (2012). High-performance pen + touch modality interactions: A real-time strategy game esports context. In *Proc. ACM UIST*, pages 309–318. ACM.
http://doi.acm.org/10.1145/2380116.2380156.

Hammond, T. and Davis, R. (2005). Ladder, a sketching language for user interface developers. *Comput. Graph.*, 29(4):518–532.

Harrison, C., Schwarz, J., and Hudson, S. E. (2011). Tapsense: Enhancing finger interaction on touch surfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 627–636. ACM.
http://doi.acm.org/10.1145/2047196.2047279.

Harrison, C., Xiao, R., Schwarz, J., and Hudson, S. E. (2014). Touchtools: Leveraging familiarity and skill with physical tools to augment touch interaction. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*, CHI '14, pages 2913–2916. ACM.
http://doi.acm.org/10.1145/2556288.2557012.

Heath, C., Hindmarsh, J., and Luff, P. (2010). *Video in Qualitative Research*. Sage Publications, Los Angeles, CA, USA.

Hegel, G. W. (1807). *Phänomenologie des Geistes*. Bamberg und Würzburg.

Hegel, G. W. and Baillie, J. (2003). *The Phenomenology of Spirit (The Phenomenology of Mind)*. Dover Publications, Mineola, NY, USA.

Heidegger, M. (1962). *Being and Time*. SUNY Press, Albany, NY, USA.

Herrlich, M., Walther-Franks, B., Schröder-Kroll, R., Holthusen, J., and Malaka, R. (2011). Proxy-based selection for occluded and dynamic objects. In *Proceedings of International Conference on Smart Graphics*, pages 142–145. Springer-Verlag. http://dl.acm.org/citation.cfm?id=2032567.2032588.

Hesselmann, T., Flöring, S., and Schmitt, M. (2009). Stacked half-pie menus: Navigating nested menus on interactive tabletops. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 173–180. ACM. http://doi.acm.org/10.1145/1731903.1731936.

Hilliges, O., Izadi, S., Wilson, A. D., Hodges, S., Garcia-Mendoza, A., and Butz, A. (2009). Interactions in the air: Adding further depth to interactive tabletops. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, pages 139–148. ACM. http://doi.acm.org/10.1145/1622176.1622203.

Hinckley, K., Baudisch, P., Ramos, G., and Guimbretiere, F. (2005). Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 451–460. ACM. http://doi.acm.org/10.1145/1054972.1055035.

Hinckley, K., Chen, X. A., and Benko, H. (2013). Motion and context sensing techniques for pen computing. In *Proceedings of Graphics Interface 2013*, pages 71–78. Canadian Information Processing Society. http://dl.acm.org/citation.cfm?id=2532129.2532143.

Hinckley, K., Guimbretiere, F., Agrawala, M., Apitz, G., and Chen, N. (2006). Phrasing techniques for multi-stroke selection gestures. In *Proceedings of Graphics Interface 2006*, GI '06, pages 147–154. Canadian Information Processing Society. http://dl.acm.org/citation.cfm?id=1143079.1143104.

Hinckley, K., Pahud, M., Benko, H., Irani, P., Guimbretière, F., Gavriliu, M., Chen, X. A., Matulic, F., Buxton, W., and Wilson, A. (2014). Sensing techniques for tablet+stylus interaction. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 605–614. ACM. http://doi.acm.org/10.1145/2642918.2647379.

Hinckley, K., Yatani, K., Pahud, M., Coddington, N., Rodenhouse, J., Wilson, A., Benko, H., and Buxton, B. (2010). Pen + touch = new tools. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 27–36. ACM.
http://doi.acm.org/10.1145/1866029.1866036.

Hinckley, K., Zhao, S., Sarin, R., Baudisch, P., Cutrell, E., Shilman, M., and Tan, D. (2007). Inkseine: In situ search for active note taking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 251–260. ACM.
http://doi.acm.org/10.1145/1240624.1240666.

Hinrichs, U. and Carpendale, S. (2011). Gestures in the wild: Studying multi-touch gesture sequences on interactive tabletop exhibits. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 3023–3032. ACM.
http://doi.acm.org/10.1145/1978942.1979391.

Hoggan, E., Brewster, S. A., and Johnston, J. (2008). Investigating the effectiveness of tactile feedback for mobile touchscreens. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1573–1582. ACM.
http://doi.acm.org/10.1145/1357054.1357300.

Holland, S. and Oppenheim, D. (1999). Direct combination. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, pages 262–269. ACM.
http://doi.acm.org/10.1145/302979.303057.

Holz, C. and Baudisch, P. (2011). Understanding touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2501–2510. ACM.
http://doi.acm.org/10.1145/1978942.1979308.

Holz, C. and Baudisch, P. (2013). Fiberio: A touchscreen that senses fingerprints. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 41–50. ACM.
http://doi.acm.org/10.1145/2501988.2502021.

Hornbæk, K. and Hertzum, M. (2007). Untangling the usability of fisheye menus. *ACM Trans. Comput.-Hum. Interact.*, 14(2).

Hornecker, E. (2008). "i don't understand it either, but it is cool"—visitor interactions with a multi-touch table in a museum. In *3rd IEEE International Workshop on Horizontal interactive human computer systems (TABLETOP)*, pages 113–120. IEEE.

Hornecker, E., Marshall, P., Dalton, N. S., and Rogers, Y. (2008). Collaboration and interference: Awareness with mice or touch input. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, pages 167–176.

ACM.
http://doi.acm.org/10.1145/1460563.1460589.

Humphrey, D. (1959). *The Art of Making Dances*. Grove Press, New York, NY, USA.

Hutchins, E. L., Hollan, J. D., and Norman, D. A. (1985). Direct manipulation interfaces. *Human–Computer Interaction*, 1(4):311–338.

Igarashi, T. and Hinckley, K. (2000). Speed-dependent automatic zooming for browsing large documents. In *Proc. ACM UIST*, UIST '00, pages 139–148. ACM.
http://doi.acm.org/10.1145/354401.354435.

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*, CHI '97, pages 234–241. ACM.
http://doi.acm.org/10.1145/258549.258715.

Jacucci, G., Morrison, A., Richard, G. T., Kleimola, J., Peltonen, P., Parisi, L., and Laitinen, T. (2010). Worlds of information: Designing for engagement at a public multitouch display. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 2267–2276. ACM.
http://doi.acm.org/10.1145/1753326.1753669.

Jansson, D. G. and Smith, S. M. (1991). Design fixation. *Design Studies*, 12(1):3–11.

Jensen, K. (2013). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*, volume 1. Springer Science & Business Media.

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*, TEI '07, pages 139–146. ACM.
http://doi.acm.org/10.1145/1226969.1226998.

Ju, W., Lee, B. A., and Klemmer, S. R. (2008). Range: Exploring implicit interaction through electronic whiteboard design. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, pages 17–26. ACM.
http://doi.acm.org/10.1145/1460563.1460569.

Kabbash, P., Buxton, W., and Sellen, A. (1994). Two-handed input in a compound task. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 417–423. ACM.
http://doi.acm.org/10.1145/191666.191808.

Kendon, A. (2004). *Gesture: Visible Action as Utterance*. Gesture: Visible Action as Utterance. Cambridge University Press, Cambridge, England, UK.

Kerne, A., Webb, A. M., Smith, S. M., Linder, R., Lupfer, N., Qu, Y., Moeller, J., and Damaraju, S. (2014). Using metrics of curation to evaluate information-based ideation. *ACM Transactions on Computer-Human Interaction*, 21(3):14:1–14:48.

Kharrufa, A., Nicholson, J., Dunphy, P., Hodges, S., Briggs, P., and Olivier, P. (2015). Using imus to identify supervisors on touch devices. In *Human-Computer Interaction*, pages 565–583. Springer.

Kienzle, W. and Hinckley, K. (2014). Lightring: Always-available 2d input on any surface. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 157–160. ACM. http://doi.acm.org/10.1145/2642918.2647376.

Konecki, K. T. (2011). Visual grounded theory: A methodological outline and examples from empirical work. *Revija za sociologiju*, 41(2):131–160.

Kristensson, P. O. and Zhai, S. (2007). Command strokes with and without preview: Using pen gestures on keyboard for command selection. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 1137–1146. ACM. http://doi.acm.org/10.1145/1240624.1240797.

Kurtenbach, G. and Buxton, W. (1993). The limits of expert performance using hierarchic marking menus. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, pages 482–487. ACM. http://doi.acm.org/10.1145/169059.169426.

Kurtenbach, G., Fitzmaurice, G., Baudel, T., and Buxton, B. (1997). The design of a gui paradigm based on tablets, two-hands, and transparency. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, CHI '97, pages 35–42. ACM. http://doi.acm.org/10.1145/258549.258574.

Kurtenbach, G., Fitzmaurice, G. W., Owen, R. N., and Baudel, T. (1999). The hotbox: Efficient access to a large number of menu-items. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, pages 231–237. ACM. http://doi.acm.org/10.1145/302979.303047.

Kurtenbach, G., Moran, T. P., and Buxton, W. (1994). Contextual animation of gestural commands. In *Computer Graphics Forum*, volume 13, pages 305–314. Wiley Online Library.

Kwinter, S. (1998). The genealogy of models: The hammer and the song. *Diagram Works, ANY*, 23:57–62.

Laban, R. and Lawrence, F. C. (1947). *Effort*. Macdonald and Evans, London, England, UK.

Laban, R. and Ullmann, L. (1963). *Modern Educational Dance*. MacDonald and Evans, London, England, UK, second edition.

Lafreniere, B., Chilana, P. K., Fourney, A., and Terry, M. A. (2015). These aren't the commands you're looking for: Addressing false feedforward in feature-rich software. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, UIST '15, pages 619–628. ACM. http://doi.acm.org/10.1145/2807442.2807482.

Laseau, P. (2001). *Graphic Thinking for Architects & Designers*. John Wiley & Sons, New York, NY, USA.

Latulipe, C., Mann, S., Kaplan, C. S., and Clarke, C. L. A. (2006). symspline: Symmetric two-handed spline manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 349–358. ACM. http://doi.acm.org/10.1145/1124772.1124825.

Lee, J. C., Dietz, P. H., Leigh, D., Yerazunis, W. S., and Hudson, S. E. (2004). Haptic pen: A tactile feedback stylus for touch screens. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, pages 291–294. ACM. http://doi.acm.org/10.1145/1029632.1029682.

Leganchuk, A., Zhai, S., and Buxton, W. (1998). Manual and cognitive benefits of two-handed input: An experimental study. *ACM Transactions on Computer-Human Interaction*, 5(4):326–359.

Leithinger, D. and Haller, M. (2007). Improving menu interaction for cluttered tabletop setups with user-drawn path menus. In *IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 121–128.

Lepinski, G. J., Grossman, T., and Fitzmaurice, G. (2010). The design and evaluation of multitouch marking menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 2233–2242. ACM. http://doi.acm.org/10.1145/1753326.1753663.

Levelt, W. J. (1993). *Speaking: From Intention to Articulation*. ACL-MIT Press series in natural-language processing. MIT Press, Cambridge, MA, USA.

Li, F. C. Y., Guy, R. T., Yatani, K., and Truong, K. N. (2011). The 1line keyboard: A qwerty layout in a single line. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 461–470. ACM. http://doi.acm.org/10.1145/2047196.2047257.

Loke, L., Larssen, A. T., and Robertson, T. (2005). Labanotation for design of movement-based interaction. In *Proceedings of the Second Australasian Conference on Interactive Entertainment*, IE '05, pages 113–120. Creativity & Cognition Studios Press. http://dl.acm.org/citation.cfm?id=1109180.1109197.

Loke, L., Larssen, A. T., Robertson, T., and Edwards, J. (2007). Understanding movement for interaction design: frameworks and approaches. *Personal and Ubiquitous Computing*, 11(8):691–701.

Lucero, A. (2012). Framing, aligning, paradoxing, abstracting, and directing: How design mood boards work. In *Proceedings of the Designing Interactive Systems Conference*, DIS '12, pages 438–447. ACM. http://doi.acm.org/10.1145/2317956.2318021.

Lucero, A., Aliakseyeu, D., and Martens, J.-B. (2008). Funky wall: Presenting mood boards using gesture, speech and visuals. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '08, pages 425–428. ACM. http://doi.acm.org/10.1145/1385569.1385650.

Lupfer, N., Kerne, A., Webb, A. M., and Linder, R. (2016). Patterns of free-form curation: Visual thinking with web content. In *Proceedings of the 2016 ACM on Multimedia Conference*, MM '16, pages 12–21. ACM. http://doi.acm.org/10.1145/2964284.2964303.

Malacria, S., Bailly, G., Harrison, J., Cockburn, A., and Gutwin, C. (2013). Promoting hotkey use through rehearsal with exposehk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 573–582. ACM. http://doi.acm.org/10.1145/2470654.2470735.

Maletic, V. (1987). *Body - Space - Expression: The Development of Rudolf Laban's Movement and Dance Concepts*. De Gruyter, Berlin, Germany.

Mare, S., Markham, A. M., Cornelius, C., Peterson, R., and Kotz, D. (2014). Zebra: zero-effort bilateral recurring authentication. In *2014 IEEE Symposium on Security and Privacy*, pages 705–720. IEEE.

Marquardt, N., Kiemer, J., and Greenberg, S. (2010). What caused that touch?: Expressive interaction with a surface through fiduciary-tagged gloves. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 139–142. ACM. http://doi.acm.org/10.1145/1936652.1936680.

Matulic, F. and Norrie, M. (2012). Empirical evaluation of uni- and bimodal pen and touch interaction properties on digital tabletops. In *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces*, ITS '12, pages 143–152. ACM. http://doi.acm.org/10.1145/2396636.2396659.

Matulic, F. and Norrie, M. C. (2013). Pen and touch gestural environment for document editing on interactive tabletops. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces*, ITS '13, pages 41–50. ACM. http://doi.acm.org/10.1145/2512349.2512802.

Mazalek, A., Davenport, G., and Ishii, H. (2002). Tangible viewpoints: A physical approach to multimedia stories. In *Proceedings of the Tenth ACM International Conference on Multimedia*, MULTIMEDIA '02, pages 153–160. ACM. http://doi.acm.org/10.1145/641007.641037.

Mentis, H. M. and Johansson, C. (2013). Seeing movement qualities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 3375–3384. ACM. http://doi.acm.org/10.1145/2470654.2466462.

Merleau-Ponty, M. (1965). *Phenomenology of Perception*. International library of philosophy and scientific method. Routledge & Kegan Paul, London, England, UK.

Merrill, D., Sun, E., and Kalanithi, J. (2012). Sifteo cubes. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '12, pages 1015–1018. ACM. http://doi.acm.org/10.1145/2212776.2212374.

Microsoft Band (2015). http://www.microsoft.com/microsoft-band.

Moeller, J. and Kerne, A. (2012). Zerotouch: An optical multi-touch and free-air interaction architecture. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 2165–2174. ACM. http://doi.acm.org/10.1145/2207676.2208368.

Morris, M. R., Danielescu, A., Drucker, S., Fisher, D., Lee, B., schraefel, m. c., and Wobbrock, J. O. (2014). Reducing legacy bias in gesture elicitation studies. *Interactions*, 21(3):40–45.

Morris, M. R., Huang, A., Paepcke, A., and Winograd, T. (2006). Cooperative gestures: Multi-user gestural interactions for co-located groupware. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 1201–1210. ACM. http://doi.acm.org/10.1145/1124772.1124952.

Morris, M. R., Wobbrock, J. O., and Wilson, A. D. (2010). Understanding users' preferences for surface gestures. In *Proceedings of Graphics Interface 2010*, GI '10, pages 261–268. Canadian Information Processing Society. http://dl.acm.org/citation.cfm?id=1839214.1839260.

Moscovich, T. and Hughes, J. F. (2004). Navigating documents with the virtual scroll ring. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, pages 57–60. ACM. http://doi.acm.org/10.1145/1029632.1029642.

Müller-Brockmann, J. (1996). *Grid Systems in Graphic Design: A Visual Communication Manual for Graphic Designers, Typographers and Three Dimensional Designers*. Visual communication books. Niggli, Salenstein, Switzerland.

Mynatt, E. D., Igarashi, T., Edwards, W. K., and LaMarca, A. (1999). Flatland: New dimensions in office whiteboards. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, pages 346–353. ACM. http://doi.acm.org/10.1145/302979.303108.

Nattiez, J.-J. (1990). *Music and Discourse: Toward a Semiology of Music*. Princeton University Press, Princeton, NJ, USA.

Norman, D. A. (1988). *The Design of Everyday Things*. MIT Press, Cambridge, MA, USA.

Odell, D. L., Davis, R. C., Smith, A., and Wright, P. K. (2004). Toolglasses, marking menus, and hotkeys: A comparison of one and two-handed command selection techniques. In *Proceedings of Graphics Interface 2004*, GI '04, pages 17–24. Canadian Human-Computer Communications Society. http://dl.acm.org/citation.cfm?id=1006058.1006061.

Ogata, M., Sugiura, Y., Osawa, H., and Imai, M. (2012). iring: Intelligent ring using infrared reflection. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 131–136. ACM. http://doi.acm.org/10.1145/2380116.2380135.

Olson, J. S. and Kellogg, W. A. (2014). *Ways of Knowing in HCI*. Springer, New York, NY, USA.

O'Neill, P. (2012). *The Culture of Curating and the Curating of Culture (s)*. MIT Press, Cambridge, MA, USA.

Pedersen, E. R., McCall, K., Moran, T. P., and Halasz, F. G. (1993). Tivoli: An electronic whiteboard for informal workgroup meetings. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, pages 391–398. ACM. http://doi.acm.org/10.1145/169059.169309.

Peponis, J., Lycourioti, I., and Mari, I. (2002). Spatial models, design reasons and the construction of spatial meaning. *Philosophica – Diagrams and the anthropology of space*, 70:59–90.

Perlin, K. and Fox, D. (1993). Pad: An alternative approach to the computer interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 57–64. ACM. http://doi.acm.org/10.1145/166117.166125.

Poller, M. F. and Garter, S. K. (1984). The effects of modes on text editing by experienced editor users. *Journal of the Human Factors and Ergonomics Society*, 26(4):449–462.

Pook, S., Lecolinet, E., Vaysseix, G., and Barillot, E. (2000). Control menus: Excecution and control in a single interactor. In *CHI '00 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '00, pages 263–264. ACM. http://doi.acm.org/10.1145/633292.633446.

Quek, F. (2006). Embodiment and multimodality. In *Proceedings of the 8th International Conference on Multimodal Interfaces*, ICMI '06, pages 388–390. ACM. http://doi.acm.org/10.1145/1180995.1181067.

Ramakers, R., Vanacken, D., Luyten, K., Coninx, K., and Schöning, J. (2012). Carpus: A non-intrusive user identification technique for interactive surfaces. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 35–44. ACM. http://doi.acm.org/10.1145/2380116.2380123.

Ramos, G., Robertson, G., Czerwinski, M., Tan, D., Baudisch, P., Hinckley, K., and Agrawala, M. (2006). Tumble! Splat! Helping users access and manipulate occluded content in 2D drawings. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '06, pages 428–435. ACM. http://doi.acm.org/10.1145/1133265.1133351.

Robbins, E. and Cullinan, E. (1994). *Why Architects Draw*. MIT press, Cambridge, MA, USA.

Rofouei, M., Wilson, A., Brush, A., and Tansley, S. (2012). Your phone or mine? Fusing body, touch and device sensing for multi-user device-display interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1915–1918. ACM. http://doi.acm.org/10.1145/2207676.2208332.

Scaife, M. and Rogers, Y. (1996). External cognition: How do graphical representations work? *International Journal of Human-Computer Studies*, 45(2):185–213.

Scarr, J., Cockburn, A., Gutwin, C., and Bunt, A. (2012). Improving command selection with commandmaps. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 257–266. ACM. http://doi.acm.org/10.1145/2207676.2207713.

Schiphorst, T. (2009). Soft(n): Toward a somaesthetics of touch. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '09, pages 2427–2438. ACM.
http://doi.acm.org/10.1145/1520340.1520345.

Schmidt, D., Block, F., and Gellersen, H. (2009). A comparison of direct and indirect multi-touch input for large surfaces. In *IFIP Conference on Human-Computer Interaction*, pages 582–594. Springer.

Schön, D. (1983). *The Reflective Practitioner: How Professionals Think in Action*. Basic Books.

Schön, D. A. and Wiggins, G. (1992). Kinds of seeing and their functions in designing. *Design Studies*, 13(2):135 – 156.

Sears, A. and Shneiderman, B. (1994). Split menus: Effectively using selection frequency to organize menus. *ACM Transactions on Computer-Human Interaction*, 1(1):27–51.

Sellen, A. J., Kurtenbach, G. P., and Buxton, W. A. S. (1992). The prevention of mode errors through sensory feedback. *Human-Computer Interaction*, 7(2):141–164.

Shaer, O. and Hornecker, E. (2010). Tangible user interfaces: Past, present, and future directions. *Foundations and Trends in Human-Computer Interaction*, 3(1–2):1–137.

Shah, J. J. (1998). Experimental investigation of progressive idea generation techniques in engineering design. In *Proceedings of ASME DETC Design Theory and Methodology Conference*, pages 13–16.

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *Computer*, 16(8):57–69.

Shneiderman, B., Fischer, G., Czerwinski, M., Resnick, M., Myers, B., Candy, L., Edmonds, E., Eisenberg, M., Giaccardi, E., Hewett, T., et al. (2006). Creativity support tools: Report from a us national science foundation sponsored workshop. *International Journal of Human-Computer Interaction*, 20(2):61–77.

Siek, K. A., Hayes, G. R., Newman, M. W., and Tang, J. C. (2014). Field deployments: Knowing from using in context. In *Ways of Knowing in HCI*, pages 119–142. Springer, New York, NY, USA.

Silang Maranan, D., Fdili Alaoui, S., Schiphorst, T., Pasquier, P., Subyen, P., and Bartram, L. (2014). Designing for movement: Evaluating computational models using lma effort qualities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 991–1000. ACM.
http://doi.acm.org/10.1145/2556288.2557251.

Strange, A. (2014). Wear this bluetooth ring to contorl all your devices with your finger. http://mashable.com/2014/04/29/nod-bluetooth-ring.

Strauss, A. and Corbin, J. (1998). *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*. Sage Publications, Los Angeles, CA, USA.

Suchman, L. A. (1987). *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge University Press, Cambridge, England, UK.

Sutherland, I. E. (1964). Sketch pad a man-machine graphical communication system. In *Proceedings of the SHARE Design Automation Workshop*, DAC '64, pages 6.329–6.346. ACM. http://doi.acm.org/10.1145/800265.810742.

Suwa, M. and Tversky, B. (1997). What do architects and students perceive in their design sketches? a protocol analysis. *Design Studies*, 18(4):385 – 403.

Suwa, M., Tversky, B., Gero, J., and Purcell, T. (2001). Seeing into sketches: Regrouping parts encourages new interpretations. In *Visual and Spatial Reasoning in Design*, pages 207–219.

Tesler, L. (1981). The smalltalk environment. *Byte*, 6(8):90–147.

Tsandilas, T. and schraefel, m. c. (2007). Bubbling menus: A selective mechanism for accessing hierarchical drop-down menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 1195–1204. ACM. http://doi.acm.org/10.1145/1240624.1240806.

Tversky, B., Heiser, J., Lee, P. U., and Daniel, M. P. (2009). Explanations in gesture, diagram, and word. In Coventry, K. R., Tenbrink, T., and Bateman, J., editors, *Spatial Language and Dialogue*, pages 119–131. Oxford University Press, Oxford, England, UK.

van Dam, A. (1997). Post-wimp user interfaces. *Communications of the ACM*, 40(2):63–67.

Verstijnen, I. M., van Leeuwen, C., Goldschmidt, G., Hamel, R., and Hennessey, J. (1998). Sketching and creative discovery. *Design Studies*, 19(4):519 – 546.

Vogel, D. and Balakrishnan, R. (2004). Interactive public ambient displays: Transitioning from implicit to explicit, public to personal, interaction with multiple users. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, pages 137–146. ACM. http://doi.acm.org/10.1145/1029632.1029656.

Vogel, D. and Casiez, G. (2011). Conté: Multimodal input inspired by an artist's crayon. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 357–366. ACM. http://doi.acm.org/10.1145/2047196.2047242.

Webb, A. and Kerne, A. (2008). The in-context slider: A fluid interface component for visualization and adjustment of values while authoring. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '08, pages 91–99. ACM. http://doi.acm.org/10.1145/1385569.1385586.

Webb, A., Kerne, A., Koh, E., Joshi, P., Park, Y., and Graeber, R. (2006). Choreographic buttons: Promoting social interaction through human movement and clear affordances. In *Proceedings of the 14th ACM International Conference on Multimedia*, MM '06, pages 451–460. ACM. http://doi.acm.org/10.1145/1180639.1180731.

Webb, A. M., Kerne, A., Brown, Z., Kim, J.-H., and Kellogg, E. (2016a). Layerfish: Bimanual layering with a fisheye in-place. In *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces*, ISS '16, pages 189–198. ACM. http://doi.acm.org/10.1145/2992154.2992171.

Webb, A. M., Kerne, A., Linder, R., Lupfer, N., Qu, Y., Keith, K., Carrasco, M., and Chen, Y. (2016b). A free-form medium for curating the digital. In *Curating the Digital*, pages 73–87. Springer.

Webb, A. M., Linder, R., Kerne, A., Lupfer, N., Qu, Y., Poffenberger, B., and Revia, C. (2013). Promoting reflection and interpretation in education: Curating rich bookmarks as information composition. In *Proceedings of the 9th ACM Conference on Creativity & Cognition*, C&C '13, pages 53–62. ACM. http://doi.acm.org/10.1145/2466627.2466636.

Webb, A. M., Pahud, M., Hinckley, K., and Buxton, B. (2016c). Wearables as context for guiard-abiding bimanual touch. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 287–300. ACM. http://doi.acm.org/10.1145/2984511.2984564.

Weiser, M. (1991). The computer for the 21st century. *Scientific American*, 265(3):94–104.

Wigdor, D. and Wixon, D. (2011). *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*, chapter Design Guidelines: Self-Revealing Multi-touch Gestures, pages 145–156. Morgan Kaufmann, Burlington, MA, USA.

Wilkinson, G., Kharrufa, A., Hook, J., Pursglove, B., Wood, G., Haeuser, H., Hammerla, N. Y., Hodges, S., and Olivier, P. (2016). Expressy: Using a wrist-worn inertial measurement unit to add expressiveness to touch-based interactions. In *Proceedings of the*

*2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 2832–2844. ACM.
http://doi.acm.org/10.1145/2858036.2858223.

Wilson, A. D., Izadi, S., Hilliges, O., Garcia-Mendoza, A., and Kirk, D. (2008). Bringing physics to the surface. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 67–76. ACM.
http://doi.acm.org/10.1145/1449715.1449728.

Wobbrock, J. O., Morris, M. R., and Wilson, A. D. (2009). User-defined gestures for surface computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 1083–1092. ACM.
http://doi.acm.org/10.1145/1518701.1518866.

Wu, M. and Balakrishnan, R. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*, UIST '03, pages 193–202. ACM.
http://doi.acm.org/10.1145/964696.964718.

Yang, X.-D., Grossman, T., Wigdor, D., and Fitzmaurice, G. (2012). Magic finger: Always-available input through finger instrumentation. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 147–156. ACM.
http://doi.acm.org/10.1145/2380116.2380137.

Zhang, H., Yang, X.-D., Ens, B., Liang, H.-N., Boulanger, P., and Irani, P. (2012). See me, see you: A lightweight method for discriminating user touches on tabletop displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 2327–2336. ACM.
http://doi.acm.org/10.1145/2207676.2208392.

Zhao, S. and Balakrishnan, R. (2004). Simple vs. compound mark hierarchical marking menus. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, pages 33–42. ACM.
http://doi.acm.org/10.1145/1029632.1029639.

Zimmerman, J., Forlizzi, J., and Evenson, S. (2007). Research through design as a method for interaction design research in hci. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 493–502. ACM.
http://doi.acm.org/10.1145/1240624.1240704.