This is an excerpt from Frank Elavsky's dissertation on *Tool-making as an Intervention on the Accessibility of Interactive Data Experiences*, which can be accessed in full at this archival link:

http://reports-archive.adm.cs.cmu.edu/anon/hcii/CMU-HCII-26-103.pdf

This document contains the following sections:

- Abstract
- Table of contents
- Chapter 1: Introduction
- Chapter 2: Background & Related Work
- Chapter 3: Overview of Contributions
- Chapter 5: *Data Navigator*: Low-level Tooling for Creating Navigable Data Structures
- Chapter 6: *Skeleton*: Visual Authoring of Non-visual Data Experiences
- Chapter 8: *Softerware*: Enabling Personalization of Interactive Data Representations for Users with Disabilities
- Chapter 9: Discussion & Future Work (Sections 9.1 and 9.3 only)
- References

This document does not contain the following chapters:

- Chapter 4: *Chartability*: Heuristics as a Tool and Resource
- Chapter 7: *Cross-perception*: Rethinking Input Design Towards Blind Analytical Interaction
- Chapter 10: Biographical Sketch

Contents

List of Figures

Abstract

In this dissertation, we contribute practical advancements in tool-making as an intervention on the accessibility of interactive data experiences. The thesis of this dissertation is as follows: *This dissertation argues that the tools practitioners use to build interactive data experiences are themselves sites where accessibility barriers are produced, prevented, or alleviated for both end users and authors. This work contributes five tools—Chartability, Data Navigator, Softerware, Cross-perception, and Skeleton—that collectively center accessibility work on the empowerment of disabled and non-disabled practitioners across the full arc of evaluation, data navigation, analytical interaction, and personalization.*

Rather than framing accessibility research solely around ideal experiences for end users with disabilities, this thesis investigates why accessibility work is so difficult for the practitioners who build interactive data experiences and what tool-making can reveal about those difficulties. We organize this investigation around four domains where practitioners face the most persistent challenges: evaluation, navigation, interaction, and personalization. *Chartability*, a heuristic framework contributed first and maps the full landscape of accessibility barriers and identified these three latter domains (navigation, interaction, and personalization) as the areas where the most severe gaps remain. *Data Navigator* and *Skeleton* then investigate navigation, finding that practitioners struggle because navigation structure has no visible, manipulable representation in their workflows. *Cross-perception* engages interaction, demonstrating that blind data analysis has been constrained by existing tools and that a new interaction design framework can reshape what analytical work is possible. *Softerware* addresses personalization, revealing that access needs genuinely conflict across users and that meaningful personalization requires system-level infrastructure that does not yet exist in practitioner tooling ecosytems.

Combined, these contributions provide empirical insights and practical advancements in the state of the art for tooling that bridges gaps in current accessibility practices in visualization and data science. Our work ultimately enables people with and without disabilities to better evaluate barriers in, analyze with, design for, develop, and personalize interactive data experiences. We demonstrate that tool-making is a productive intervention that both engages accessibility barriers and elucidates why those gaps exist in practitioner work.

Part I

Introduction

Chapter 1
Introduction

This thesis is a body of research situated within the existing research area focused on making interactive data representations more accessible for people with disabilities. Much of the work in this existing area is situated within the context of making interactive data *visualizations* accessible, particularly (but not exclusively) for people who are blind. My work, contributed here in this thesis, is focused on using *tools* as a specific intervention and sub-area of study for making interactive data *representations* accessible for people with disabilities, broadly speaking. ("Representations" here is an intentionally broader term than "visualizations," which are exclusively visual representations of data.)

Before we begin, two things must be understood up front, or else the rest of this thesis could be interpreted with disruptive assumptions: we must interrogate the phrase "making visualizations accessible" and unpack why *tools* are a meaningful area of study.

1.1 Is "accessible visualization" really an oxymoron?

The first assumption that must be disrupted is perhaps the motivating cornerstone of this research, which is that the phrase "making visualizations accessible," while a noble goal, is not the semantically correct phrasing nor precisely what describes my work. This can be misleading. I do use the phrase "accessible visualization" but will admit that this seems to confuse certain people with very particular opinions about things. We will clear this up.

Villains of our field's past have written incendiary and ableist perspectives on why "no forms of data visualization, not just dashboards jam-packed with graphics, can be made fully accessible to someone who is blind," and that "[a blind man] will never be able to analyze data as I do visually, because many aspects of vision cannot be duplicated by his other senses" [41]. However, this position misunderstands what the goal of accessibility is, and arguably even what the goal of visualization itself is.

Making visualizations accessible *isn't* about the visualization, it's about making the outcomes of the visualization accessible.

Visualizations are ubiquitous and paramount for decision-making. However, the *artifact* that is a visualization is not even the goal of the act of visualizing: developing understanding, insight, confidence, and communication among and between human beings are the goals of visualization. Visualization is about making data easier to use for all kinds of things. Yes, our visual system enables us more than any other form of sensory cognition that we have [15, 48, 158]. But we aren't trying to make sight itself accessible. We are trying to make it possible for people to make meaningful decisions, gain valuable information, build conjectures, and effectively communicate with others.

Many, many people who I've spoken to over the course of my career, even before embarking on this thesis journey, misunderstand this simple fact: making a "visualization" accessible *isn't* about the visualization itself but rather making what the visualization is meant to *accomplish*

accessible. It's about equal outcomes, not equal interactions with an artifact.

People with disabilities are no small portion of the world's population. In the United States, 27% of people self-report living with at least one disability that affects their daily lives [120] and all of us will eventually age into disability (if we are lucky to live a long life).

People with disabilities (again, that will be all of us *eventually*) deserve to participate fully in life. They deserve financial independence. They also deserve loving care and interdependence. People with disabilities have a right to make informed decisions, to know about the status of a global pandemic, and to have an understanding of local and national politics [39]. While we use visualizations to navigate all of these domains, the goal is not to make the charts and graphs themselves somehow equally useful to all people. That would be a false measurement of success.

Our goal then, is measured by the success of lives led by people with disabilities [161]. Many other measurements are just metrics along the journey towards that goal. We then ask: Can people with disabilities also use data to live full lives? Can they make *fast* decisions based on data? Meaningful, careful, *slow* decisions? Communicate complex ideas? Crunch and clean data, develop models, find errors, and build hypotheses? Can they have memorable, immersive, beautiful, aesthetic experiences with data too [77]? Making "visualizations" accessible really is a misnomer. We are ultimately trying to make everything about what interactive data experiences *accomplish* for people equitable and accessible.

Again, if the goal of accessible visualization were about visualizations themselves, then the correct course of action would be one framed by the medical model [109]: that there is a normative state of behavior and capability (in this case, it would be "normal" to be able to read a visualization and make a decision) and any deviation from that norm must be corrected. This framing first assumes that the visualization should not be altered or improved. And then this framing puts the burden on the bodies of people with disabilities: that they must be "fixed" and given sight or brought to some equivalent state as someone who is "healthy," normal, and sighted. Plenty of scholars have already discussed why this framing is a problem, not only because it places undue burden on people with disabilities, produces pathologies and hierarchies of disability, but also because it is fundamentally not economically or ethically feasible.

So we then turn to other models of disability, such as the social model. The social model is heavily discussed by disability scholars and is not the end-game or last and total way of thinking about disability [109, 127, 130, 156, 191]. But the core motivation is that society, not medicine, is also a path towards solving problems that people with disabilities face. A few important concepts and concretely actionable things come from the social model that can help motivate the work of this thesis.

First, we look to the historical birth of the social model of disability: in the 504 sit-ins that took place in the United States in 1977. Cities had curbs and curbs are a barrier for people who use wheelchairs. So protests happened because decisions were being made without people with disabilities at the table. In this instance, people acknowledged that political power was an exclusive club and fought to ensure their cry "nothing about us, without us!" materialized.

And this leads us to the first and most-foundational philosophical framing for this thesis: that our *artifacts*, these things we've created from curbs to data visualizations, can become *barriers* for people with disabilities. And it is then the artifact, not the body of the person with a disability, where disability is produced in this model. Rather than a comparison to a normative state as a way to frame disability (the medical model), we instead must observe and evaluate material

outcomes based on human-made problems.

So, the social model is framed around society "solving" inequities: we get involved and make political and legal change tangible. But a second model also emerges from within the social model: one where we can now frame *who is first responsible* for repair: the curb designers and implementers.

And knowing who is first responsible for access leads us into the moral and ethical imperative that motivates this thesis: the builders and makers of visualizations are ultimately the ones who provide exclusive value for only a subset of people: those *without* disabilities. **We must first change how builders and makers do their work.**

So the phrase "accessible visualization" is really about recognizing that visualizations produce barriers for people. That means that it is our ethical imperative, as builders and makers, to fix them. And that act of fixing barriers leads us away from mere visual representations of data into a wide variety of other senses and interaction modalities. There are many paths forward towards fuller and more-equitable lives led by people with disabilities.

1.2 On *tools*, *tool-making*, and *human-tool* interaction

Then the act of making becomes immensely important: we, the builders and makers of our world, need to get things right; there is a risk involved when making things that we will exclude people with disabilities. We need to make sure that we build a better world than the one we have now. We must care for new things we create and tend to the repair and maintenance of what we've already made. And this ethical imperative leads us to the topic of *tools* and *tool-making*.

So the second thing that must be understood before we embark on this thesis is that *tools* are not the same as *solutions* or *applications*. Sometimes tools can be used to *solve* things and are certainly, in ideal circumstances, *applied* in various contexts. But understanding the role of the "tool" in human-tool interaction is paramount for engaging in the work of making anything accessible for people with disabilities.

We use tools to shape our world, break old things, and make new things. But a tool, like the hammer (as an example), does not inherently *solve* something like homelessness. But a hammer can be used to build homes if there are social policies in place and proper resources allocated. This means that for the success of tooling, there is often a larger material, social, legal, and policy reality that supports and necessitates those tools. This thesis will not be focusing on changing the upstream dependencies, but optimistically operating as if they were true (or will be true in time).

However, in some cases, tools can *destroy*. The hammer has a claw and can easily pry apart boards and tear down homes. So tools carry potential to do all kinds of things, both good and bad, and how a tool is used is often open-ended, variable, and heavily dependent on socio-technical realities. Tools participate in personal and political agendas [181] and are sometimes, for this reason, regulated or made proprietary and controlled by powerful entities [51, 177].

So tools are not without any sort of ethics. We cannot just blame tool-users for outcomes when much of a tool depends on these larger systems and structures. Technologies (tools included) encode the assumptions and biases of their *creators* as much as, if not more than, their users. Tools that build things for others to use can be loaded with assumptions about what peo-

ple are *able* to do [182] and also rules and guardrails about what anyone downstream from that tool's design *should* do [51, 176]. These assumptions, biases, and rules *limit*, *enforce*, *magnify*, *exclude*, and *enable* what a tool-user is capable of.

Tools for visualizing data are a perfect case study in this problem: virtually every major data visualization library, application, or software ever made was made entirely with the assumption that data should be transformed into visual representations. This is a reasonable assumption, since virtually all of the tool-makers are sighted and visualization is incredibly helpful to our cognition of and communication with data [45].

So data visualization, as a field, has focused its tool-making efforts on reducing the difficulty involved in visualizing data. Some visualization tools are concise [136], others are lower level but much more expressive [11]. Tool-making in visualization has focused on making it easier to scaffold a wide variety of interactions both with the visualizations as well as with their underlying data models [63].

However, as time has moved on, people began to speak out about color-vision deficiency in data visualization. Some people, primarily those with X/Y chromosomes (largely men) who are of European ancestry, have a deficiency in their ability to perceive certain colors. Then a plethora of research arose that began to look into the barriers that folks who are colorblind face in data visualization. As a result, our practices and tools improved. We began to educate practitioners, develop new color palettes, researched new methods for testing our designs, and built new systems for handling automatic color encoding. Our tools evolved.

But now data visualizations have arguably become ubiquitous in daily life. By comparison, we have far more tools now for making visualizations quickly and easily than we do for representing data in non-visual ways. We also have far more research, relatively speaking, into how sighted end users interact with visualizations.

So this thesis engages gaps that arise in this space: Practitioners face immense challenges when crafting accessible data experiences. We first need to educate practitioners on what accessibility barriers actually are in interactive visualizations. Then, we must help them engage the hardest barriers in this work and create building blocks that help them to construct navigable data experiences, build design frameworks that can inform entirely new kinds of data interaction, and develop software systems for end-user personalization and agency. Our research seeks to advance the state of the art in tools that assist in accessible data interaction while also using tool-making as an intervention that helps us to better understand and characterize *why* and *how* data practitioners face barriers themselves in this work.

Chapter 2
Background & Related Work

2.1 Practitioners and Tools

2.1.1 Understanding Builders, Makers, Designers, and Developers

Research investigating the practices and experiences of individuals who create with computers employs a range of high-level methods. Ethnographic studies, case studies, and design ethnographies are common approaches, allowing researchers to immerse themselves in communities such as the DIY/maker and assistive technology spaces [76, 79]. These methods capture the nuanced challenges practitioners face when engaging in new and unfamiliar work, including the transition from traditional to digital fabrication, coding, and tool creation [78]. By observing and interviewing practitioners in naturalistic settings, researchers uncover the social, cultural, and technical factors that shape how makers adapt and evolve their work practices.

Participatory design and co-creation are also central to this field [157]. These approaches encourage collaboration between researchers and practitioners or end-users, enabling a deeper understanding of the cognitive and creative processes behind design and development [55]. Such collaborative sessions reveal how designers shift their thinking when encountering novel challenges, embracing iterative processes that blend experimentation with reflection. Similarly, developers often modify their applications, tools, and even programming languages through feedback loops and community-driven innovation, highlighting a dynamic interplay between individual creativity and collective knowledge.

Additionally, design-based and case-study research methods explore how new practices can augment the existing work of practitioners [22, 83]. This involves not merely filling gaps or solving isolated problems but reimagining the possibilities for creative and technical expression. Researchers in this space envision systems that support continuous learning, adaptation, and innovation [53]. The focus is on enabling practitioners to extend their capabilities—providing scaffolds for experimentation, fostering environments where unconventional approaches are encouraged, and integrating new technologies in ways that amplify creativity and intelligence rather than simply addressing deficits [87, 163, 182].

Overall, the research methods used in this area are multidisciplinary, combining qualitative insights with iterative design practices to offer a holistic picture of the challenges and opportunities of builders, makers, designers, developers.

2.1.2 Approaches to Tool-making in Human-Computer Interaction

In human-computer interaction, tool-making research spans both the creation of entirely new capabilities and the enhancement of existing systems. One prominent approach involves piggybacking on current systems—leveraging their established functionalities to introduce improvements that streamline workflow or unlock new interactions [52]. This method often focuses on integrating with widely used platforms to amplify their usability, enabling users to perform

tasks in more intuitive or efficient ways. By building on existing infrastructures, researchers can demonstrate how small, targeted modifications have the potential to transform user experiences.

Another significant approach centers on the notion of appropriation [27, 30, 133, 160]. Here, research examines how users adapt tools for uses beyond their original intent. Studies in this vein explore the creative processes behind such re-purposing, uncovering the latent functionalities and opportunities that emerge when practitioners modify systems to suit their unique needs. This perspective often leads to the development of modular, extensible tools that encourage experimentation and user customization, fostering a more personalized interaction with technology. In some cases, theory has been developed from the study of emergent and generative tool-use [5, 7], broadly informing future tooling projects as well as general theories of creative human interaction with technology.

Beyond these, tool-making in HCI also includes the development of systems designed to empower users by providing entirely new capabilities, sometimes explicitly named "toolkits" and other times generally just referred to for their ability to enable novel interaction and outcomes [94, 123, 134, 147, 147]. These projects may range from novel software environments that facilitate rapid prototyping and live programming to innovative hardware devices that bridge the gap between digital and physical interactions [66, 68, 123]. The emphasis is not solely on problem-solving but on enabling creative exploration, new possibilties, and even hacking the potential of technologies towards new ends [67]. Such projects often present their contributions through demonstrative prototypes and case studies that reveal potential applications, even if they are accompanied by minimal formal evaluations [35].

This body of work reflects a balance between novelty and practicality. While some projects aim to introduce groundbreaking new ways to interact with data and systems, others refine existing practices to improve efficiency and accessibility. Together, these approaches underscore a commitment to enhancing human capabilities, allowing users to not only solve problems more effectively but also to unlock new avenues for creativity and innovation.

2.2 Data, Accessibility, and Data *and* Accessibility

2.2.1 Advancements in Interactive Data Visualization and Data Science

Recent years have witnessed significant advancements in interactive data science and visualization, driven by innovations that enhance both the performance and usability of data tools. Cross-filtering, as a subtype of cross-linked interaction, has emerged as a powerful technique, enabling users to interact with multiple data dimensions simultaneously [6, 62, 98, 172]. By linking various filters, analysts can quickly build hypotheses and isolate patterns, trends, and anomalies in complex datasets, leading to more informed decision-making. Stress has been placed in recent years on developing fast systems that are optimized showing more and more data at once while reducing latency in user interaction as much as possible [63, 98, 185].

Automated data processing and cleaning have revolutionized workflows by reducing the time spent on manual data wrangling [37]. Sophisticated algorithms now automatically detect inconsistencies, fill missing values, and transform raw data into usable formats. These improvements enable researchers and practitioners to focus more on analysis rather than preparation.

Faster tooling has further accelerated data exploration. Enhanced computational frameworks and optimized libraries allow for real-time data manipulation, making interactive visualization more responsive [63]. Coupled with easier-to-use grammars and scripting languages, these tools lower the barrier to entry, empowering users with limited visualization, geometry, trigonometry, data, and graphics coding experience to generate complex, interactive, visual representations of data [136]. New visualization types and techniques—ranging from dynamic dashboards, faceting, to immersive 3D visualizations—offer novel ways to explore and interpret data [185].

Despite significant breakthroughs, current advancements have largely neglected the needs of people with disabilities. Innovations in data science and visualization have focused on sighted user populations, prioritizing visual clarity and interaction speed using direct pointer techniques (such as with touch or a mouse) [110]. This focus often overlooks accessibility requirements for individuals who are blind, have low vision, experience cognitive or vestibular challenges, or possess motor disabilities that limit traditional pointer use [179].

2.2.2 Accessibility and Assistive Technology in Research versus Practice

2.2.2.1 Research: Focus on Blindness and Computer Output

Research and standards are both somewhat limited by a strong bias towards visual disabilities. In *Chartability*, 36 of the 50 criteria related to accessible visualization considerations involve visual disabilities [34, 39]. Marriott et al. also found that visual disability considerations are the primary focus of data visualization literature [110], leaving the barriers that many other demographics face unstudied. Accessibility research broadly has traditionally concentrated on the experiences of individuals who are blind, investigating how they perceive and interpret computational output [105]. Studies in this area explore alternative modalities for conveying data, such as auditory representations (through synthesized speech), tactile interfaces, and sonification techniques. Researchers focus on identifying effective methods for transforming visual data into formats that blind users can easily comprehend. This body of work not only examines the perceptual challenges but also delves into cognitive processing differences, aiming to optimize the accessibility of complex information and interactive systems for users with visual impairments.

While research has made strides in converting visual outputs into auditory or tactile forms for blind users, interactive input methods remain underdeveloped. Most efforts have concentrated on optimizing screen reader navigation and information retrieval, leaving text entry and command execution cumbersome. Screen readers, as they currently exist, offer limited support for efficient input, making it challenging for users to perform complex interactions. Although tactile interfaces hold promise for providing more intuitive input methods, they are still in the experimental stage and have not been fully integrated into mainstream accessible computing solutions, perpetuating a critical gap in effective user interaction.

2.2.2.2 Practice: Focus on Standards and Specialization

In contrast, practical accessibility efforts are often centered on the implementation and adherence to established standards and guidelines, such as WCAG [166]. There has been a growing interest in developing guidelines for practitioners [31, 34] and even applying guidelines as a

method of validation alongside human studies evaluations and co-design [39, 101, 102, 194]. Existing accessibility standards bodies like the Web Content Accessibility Guidelines do stress the importance of accurate, functional semantics in order for screen reader users to know how to interact with elements [169]. For interactive visualizations this means that button-like or link-like behavior should expressly be made using elements that are semantically buttons and links.

Accessibility professionals, who typically possess specialized expertise, act as intermediaries between the design and development of digital products and the strict requirements of accessibility standards. Their role involves translating abstract guidelines into concrete design solutions, ensuring that websites, applications, and services meet regulatory benchmarks. By focusing on a standards-based approach, practitioners help organizations navigate the complexities of legal and technical requirements, thus ensuring that accessible design principles are integrated into mainstream technology development. This dual focus on rigorous standards and specialized expertise ensures that accessibility is both technically sound and legally compliant across diverse digital environments.

However, accessibility standards are inherently reactive, often lagging behind rapid technological advancements by five, ten, or even twenty years (or more). This delay occurs because developing, vetting, and formalizing standards requires consensus among diverse stakeholders and extensive testing to ensure compatibility and compliance. In contrast, cutting-edge interfaces and computational capabilities evolve swiftly, driven by dynamic market forces and user innovations. Consequently, accessibility guidelines tend to reflect outdated technologies, creating a persistent gap between modern interactive systems and current best practices in accessibility.

2.2.3 Data and Accessibility

In parallel to Mack et al.'s "What do we mean by Accessibility Research?" [105] nearly all topics of study at the intersection of accessibility and data are focused on visualization and vision-related disabilities [179]. Largely, access issues other than vision that affect data visualization (such as cognitive/neurological, vestibular, and motor concerns) are almost entirely unserved in this research space. Kim et al. found that 56 papers have been published between 1999 and 2020 that focus on vision-related accessibility (not including color vision deficiency), with only 3 being published at a visualization venue (and only recently since 2018) [92]. Marriott et al. found that there is no research at all that engages motor accessibility [110]. We have found 2 papers that engage cognitive/neurological disability in visualization and 1 student poster from IEEE Vis, which are all recent (specifically intellectual developmental disabilities [188] and seizure risk [154, 155]). We found no papers that engage vestibular accessibility, such as motion and animation-related accessibility. We also found that there is no research specific to low vision disabilities (not blindness or color vision deficiency) unless conflated with screen reader usage in data visualization. Blind and low vision people are often researched together, but in practice may use different assistive technologies (such as magnifiers and contrast enhancers) and have different interaction practices (such as a combination of sight, magnification, and screen reader use) [159].

Since the 1990s, the most prominent and active accessibility topic in data has been color vision deficiency in data visualization [20, 96, 112, 119, 121]. Research projects that explore tactile sensory substitutions to charts have been a topic in computational sciences dating back to

the 1983 [139], with tactile sensory substitutions being used for maps and charts as far back as the 1830s [54]. Sonification used both in comparison to and alongside visualization and tactile methods for accessibility dates as far back as 1985 [14, 23, 44, 108, 113, 192]. Some more recent work has explored robust screen reader data interaction techniques [49, 152], screen reader user experiences with digital, 2-D spatial representations, including data visualizations [137, 144], dug deeper into the semantic layers of effective chart descriptions [101], and investigated how to better understand the role of sensory substitution [21]. Jung et al. offer guidance that expands beyond commonly cited literature that chart descriptions are preferably between 2 and 8 sentences long, written in plain language, and with consideration for the order of information and navigation [86].

A wide array of emerging research projects investigate screen reader users needs, barriers, and preferences, and offer guidelines, models, and considerations for creating accessible data visualizations [21, 39, 101, 144]. Jung et al. offer guidance to consider the order of information in textual descriptions and during navigation [86]. Kim et al. collected screen reader users' questions when interacting with data visualizations, which could open the door for more natural language data interaction [91].

Data visualization accessibility has come far in recent years. But little work has been done to explore what disability scholars call "access friction" - a tension that arises when access must be negotiated [57, 77]. This friction is often a result of static barriers in shared spaces: one artifact or approach designed to include some people may end up excluding others.

Yet despite these resources, making data visualizations more accessible remains a difficult task for practitioners [85, 146]. Some accessibility guidelines even conflict, for example on the topic of patterns and textures used in charts. One side stresses that patterns are harmful to cognitive and visual accessibility [138] while another stresses that redundant encoding strategies are necessary [34].

These difficulties point to a deeper problem: the tools practitioners use to build data experiences were not designed with accessibility in mind, and the resulting gaps are not evenly distributed. Even in an ideal state where guidelines agree, the hardest remaining challenges cluster in three domains: navigation, interaction, and personalization. In these, practitioners lack the structural, technical, and infrastructural means to reason about accessible design and act on those considerations.

Chapter 3
Overview of Contributions

In this dissertation, we contribute practical advancements in tool-making as an intervention on the accessibility of interactive data experiences. The thesis of this dissertation is as follows: *This dissertation argues that the tools practitioners use to build interactive data experiences are themselves sites where accessibility barriers are produced, prevented, or alleviated for both end users and authors. This work contributes five tools—Chartability, Data Navigator, Softerware, Cross-perception, and Skeleton—that collectively center accessibility work on the empowerment of disabled and non-disabled practitioners across the full arc of evaluation, data navigation, analytical interaction, and personalization.*

Rather than framing accessibility research solely around ideal experiences for end users with disabilities, this thesis investigates why accessibility work is so difficult for the practitioners who build interactive data experiences and what tool-making can reveal about those difficulties. We organize this investigation around four domains where practitioners face the most persistent challenges: evaluation, navigation, interaction, and personalization. *Chartability*, a heuristic framework contributed first and maps the full landscape of accessibility barriers and identified these three latter domains (navigation, interaction, and personalization) as the areas where the most severe gaps remain. *Data Navigator* and *Skeleton* then investigate navigation, finding that practitioners struggle because navigation structure has no visible, manipulable representation in their workflows. *Cross-perception* engages interaction, demonstrating that blind data analysis has been constrained by existing tools and that a new interaction design framework can reshape what analytical work is possible. *Softerware* addresses personalization, revealing that access needs genuinely conflict across users and that meaningful personalization requires system-level infrastructure that does not yet exist in practitioner tooling ecosytems.

We engage each domain below with the questions: "Why does this work matter?", "Why is it hard?", and "What has tool-making within this domain showed us?"

The 4 domains of work that I engage in this thesis start first with **evaluation**. In work contexts where someone is designing and developing interactive data experiences, the practitioner must have the knowledge, tools, and resources available to systematically identify how their interfaces produce barriers for people with disabilities. A significant portion of professional accessibility work (arguably most, if not all) is founded on auditing and evaluating barriers to access. This work is pre-dominantly done through a standards-based approach [166], although in more robust evaluation work, people with disabilities are actively involved in the process [124].

Evaluation is difficult work because much of it is contextually defined by the author themselves, and most tasks at this intersection require careful, non-automated processes and methods [26, 124]. To make matters more difficult, no comprehensive guidelines, tests, and tools exist in any singular location. Practitioners often must gather these resources themselves, which tend to be situated towards accessibility in general or are high-level and provide minimal usefulness in practice. Additionally, practitioners themselves often have little knowledge about the veracity or quality of any given bit of information they gather [85, 146], and often do the work themselves to synthesize this disparate space of information into a usable format they can apply

to their own evaluation work.

To engage this, our first main chapter focuses on *Chartability*, a heuristic framework that enables designers, developers, and auditors to systematically evaluate data visualizations and interfaces for a wide range of accessibility barriers, considering people with visual, motor, vestibular, neurological, and cognitive disabilities. In this project, we did the hard work for other practitioners and contributed our collection of synthesized accessibility resources in a single workbook. We had practitioners try out our resource in real environments, in-situ, in order to learn more about the challenges and barriers they themselves faced in evaluation work. With *Chartability*, practitioners, especially those with limited accessibility expertise, gained more confidence and clarity in assessing and improving their work. Additionally, *Chartability* has since become widely applied as a framework that isn't just used for evaluation but also as design guidance in many contexts, internationally, including policy organizations, governmental groups, and more than 100 companies and businesses.

Chartability then opened up a significant landscape of new projects and research directions. From a combination of my existing expertise as a visualization designer and engineer, in addition to continued application of *Chartability* in the wild, we began to identify the trickiest and most-difficult domains of work for practitioners. Chartability has 50 total heuristics, or tests, each organized under one of 7 principles. But 3 larger domains began to emerge as the areas where the most severe and dramatic accessibility barriers remained unaddressed: on data *navigation*, analytical *interaction*, and interface *personalization*.

So the next section of this thesis engages the first of these three: **navigation**. Navigation is a fundamental type of interaction that is leveraged by modern software-based assistive technologies. Screen readers, the primary tool used by people who are blind to interact with computers, navigate content. Additionally, many other assistive technologies, such as a sip and puff device (like the "POSSUM" from as far back as '63 [107]) also navigate. Navigational technologies are leveraged by people with a significant array of disabilities, yet tend to be entirely ignored by existing data visualization tools, which are pre-dominantly built to support direct input (using a computer mouse).

Empirical work has already demonstrated that structural navigation is actually good [194], even when regular alternative text (image descriptions) exist. This is both because people who are blind can gain both a high level understanding (from the description) as well as lower-level sense of the data's structure and arrangement, in addition to the fact that discrete, structural navigation exposes interactivity that may exist on any visualization elements (such as they can be hovered or clicked with a mouse in order to perform some action). So, if good empirical work exists: *why haven't practitioners put this research into action? What makes this work hard to do?*

We first built *Data Navigator* to provide the building blocks we needed in order to address the technical and conceptual gaps that were required to make any visualization or visualization tool provide a navigable, interactive structure. *Data Navigator* is a low-level toolkit which can be used to construct accessible navigation structures such as lists, trees, and diagrams from an underlying graph structure. We leveraged graph theory for an applied HCI problem: nodes and edges represent any relationships within the data as a structure, which then supports rich expressiveness of data navigation experiences. Users can navigate discrete marks in a visualization, clusters, groupings, and more. In addition to its structural scaffolding, *Data Navigator* also supports a wide array of input modalities leveraged by people with disabilities (screen readers,

keyboards, speech, and gestures).

Data Navigator provided a substrate, but this contribution alone wasn't enough to engage the question *why is navigation so hard, in practice?*. So the chapter following *Data Navigator* introduces *Skeleton*, a data navigation authoring tool built on top of *Data Navigator*. Our novel approach in *Skeleton* involves visualizing and making manipulable the nodes, edges, and textual data that comprise non-visual end user experiences. *Skeleton* visualizes the building blocks that comprise *Data Navigator*. Additionally, *Skeleton* provides expressive, rapid scaffolding capabilities that leverage data visualization rendering engines. This scaffolding engine helps practitioners quickly create common configurations for non-visual data navigation structures that retain visual congruence to the underlying structure.

But most importantly, *Skeleton* serves as a framework that shapes designerly consideration. Our conjecture was that because sighted practitioners cannot *see* navigation building blocks, they will not treat those elements as iterable design materials. We conjectured: Navigation is hard in practice because sighted designers face barriers to iteration and understanding. We conducted an empirical study with sighted practitioners and found that making non-visual elements visual helped practitioners shift from treating accessibility as a compliance task to treating it as a design problem, re-iterating on the visual aspects of their design, and engaging in the complex and nuanced components that comprise data navigation experiences.

Now, **interaction** becomes the next area we wanted to engage. Existing accessible data interaction for people who are blind, including our previous work on navigation (which is a form of interaction), predominantly seeks to expose information. This is what we call *access-oriented interaction*. In terms of low-level analytical tasks, most are then made feasible through navigation, sonification, or summarization-based and question-answering approaches: retrieving values, filtering, computing derived values, sorting, determining ranges, clustering, and finding outliers. What remains are analytical tasks that, despite being "low level" (understood as *unable to be reduced into more fundamental tasks*), are cognitively highly complex: finding correlations and characterizing distributions [3]. These tasks require complex hypothesization and exploration, rather than a system that simply encourages surfacing what is known or what is already present in the data: it requires combining, remixing, restructuring, and dividing data.

But blind *analytical interaction* isn't just important to engage because it is understudied, it is important to engage because many of interactive information visualization's most impactful tools for data science enable it [44, 63, 117, 172]. In visualization, *cross-filtering* is one example of an interaction that enables a user to filter one visual space while seeing a coordinated change in another visual space simultaneously and near-instantaneously. The speed of input interaction and perception of output also matters: even a small bit of latency changes the quality of a user's data exploration activities [98]. We conjectured that a screen reader, the most-used tool leveraged by blind people when interacting with computers, may be insufficient for engaging this task.

To engage this, this thesis introduces *Cross-perception*, an approach for building analytical interactions that support perception in one space of input interaction with simultaneous, non-competing perception of output in another space of data representation. We first formalized a design framework for producing *cross-perception* experiences and then built a novel prototype device, the *cross-feelter*, that enables blind *cross-perception* of a cross-filtering data exploration interface. In an empirical study with blind users (with and without existing data expertise), we found *cross-perception* speeds up analytical exploration by 90% and helps blind users consider

vastly more questions of their dataset (+188% computational queries, +54% spoken aloud) compared to a screen reader-driven interaction.

Beyond performance, we found that our input modality itself shaped the character of analytical engagement: participants didn't just work faster, they considered more dimensions of their data and asked qualitatively different questions. The *cross-feelter* also reduced anxiety and substantially increased enjoyment, particularly for participants without prior data expertise, suggesting that the barriers blind practitioners face in data work are not only functional but affective. Additionally, we had our blind practitioners imagine new interaction possibilities that *cross-perception* could enable including and beyond our *cross-feelter* device.

Our final domain of work is the most difficult for visualization practitioners to engage: **personalization**. While *navigation* demanded better software tooling and visual support and *interaction* required new hardware, *personalization* completely re-orients how software authoring takes place. Personalization matters because of *access friction*, which is a design challenge where one design or interface configuration that might be accessible for one person or group of people turns out to create barriers for someone else [57, 77]. In existing work on personalization and accessibility, studies have demonstrated that end-user control is great to have and can alleviate friction [84, 187], but little work has been done to explore what personalization looks like for an existing data visualization library and how practitioners should build and maintain their existing systems to support it.

Our final chapter introduces *Softerware* to address the tension between standardized accessible design and the diverse needs of real users with disabilities. In the wild, *access friction* exists in every design that reaches a public audience; it is inevitable. This tension ultimately means that some users have a worse experience, and may even face exclusion, with any particular design configuration. So for this work, we conducted our research in-situ with visualization software engineers and designers and worked to build a scalable, flexible software system dubbed "softerware" that enables end users to manipulate the appearance and functionality of the charts and graphs they encounter according to their own preferences. We conducted empirical research to inform our collaborators, as well as other visualization system authors, with guidelines and considerations for building *softerware* systems. In our study, no two participants chose the same preference configuration and participants with the same diagnosed condition sometimes needed opposite design treatments. Practitioners, meanwhile, immediately raised ethical concerns about whether personalization would let designers off the hook for poor defaults. These findings revealed that the real barriers to personalization are not at the level of any individual chart but at the level of system infrastructure: without persistence, cross-system interoperability, and shared standards, the effort required of end users exceeds the value they receive.

Combined, these contributions provide empirical insights and practical advancements in the state of the art for tooling that bridges gaps in current accessibility practices in visualization and data science. Our work ultimately enables people with and without disabilities to better evaluate barriers in, analyze with, design for, develop, and personalize interactive data experiences. We demonstrate that tool-making is a productive intervention that both engages accessibility barriers and elucidates why those gaps exist in practitioner work.

Part III

Navigation: Making Data Structures Traversable

Chapter 5
Data Navigator: Low-level Tooling for Creating Navigable Data Structures

---

This chapter was adapted from my published paper:

F. Elavsky, L. Nadolskis, and D. Moritz, 'Data Navigator: An Accessibility-Centered Data Navigation Toolkit', *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–11, 2023.

5.1 Abstract

Making data visualizations accessible for people with disabilities remains a significant challenge in current practitioner efforts. Existing visualizations often lack an underlying navigable structure, fail to engage necessary input modalities, and rely heavily on visual-only rendering practices. These limitations exclude people with disabilities, especially users of assistive technologies. To address these challenges, we present Data Navigator: a system built on a dynamic graph structure, enabling developers to construct navigable lists, trees, graphs, and flows as well as spatial, diagrammatic, and geographic relations. Data Navigator supports a wide range of input modalities: screen reader, keyboard, speech, gesture detection, and even fabricated assistive devices. We present 3 case examples with Data Navigator, demonstrating we can provide accessible navigation structures on top of raster images, integrate with existing toolkits at scale, and rapidly develop novel prototypes. Data Navigator is a step towards making accessible data visualizations easier to design and implement.

5.2 Overview

While there is a growing interest in making data visualizations more accessible for people with disabilities, current toolkit and practitioner efforts have not risen to the challenge at scale. Major data visualization tools and ecosystems predominantly produce inaccessible artifacts for many users with disabilities. We believe this is largely a gap caused by a lack of underlying structure in most visualizations, failure to engage the input modalities used by people with disabilities, and over-reliance on visual-only rendering practices.

Users who are blind or low vision commonly use screen readers and users with motor and dexterity disabilities often do not use "pointer" (precise mouse and touch) based input technology when interacting with digital interfaces. Many users with motor and dexterity disabilities use discrete navigation controls, either sequentially using keyboard-like input, or directly using voice or text commands.

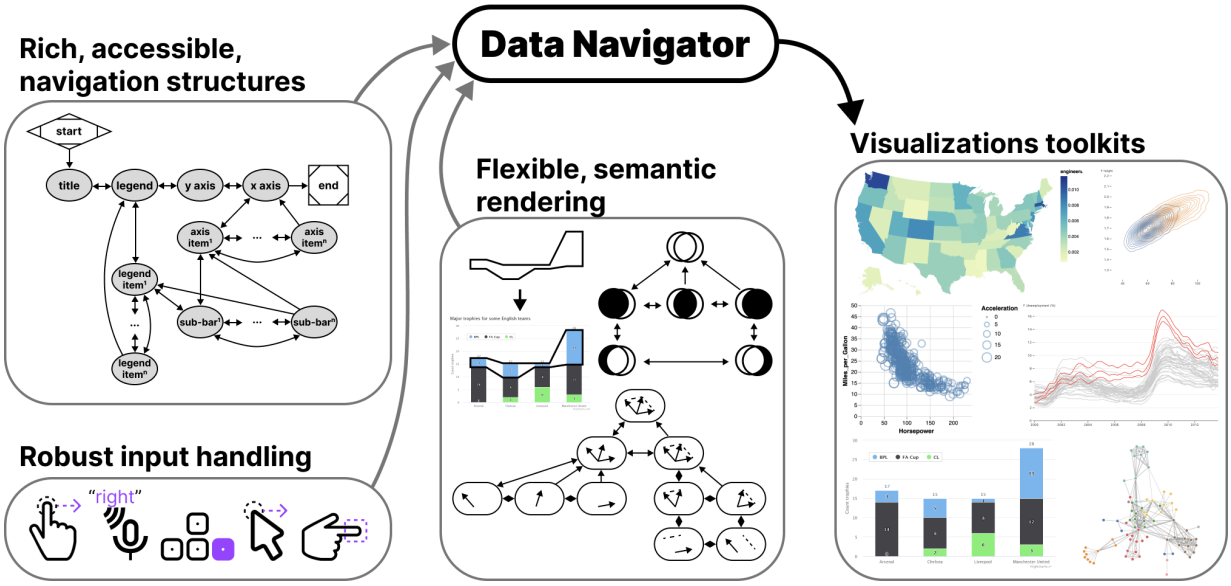

Figure 5.1: Data Navigator provides data visualization libraries and toolkits with accessible data navigation structures, robust input handling, and flexible semantic rendering capabilities.

Most interactive visualizations simply focus on pointer-based input: they can be clicked or tapped, hovered, and selected in order to perform analytical tasks. This excludes non-pointer input technologies. These devices require consideration for the navigation structure and underlying semantics of a visual interface.

However, building navigable spatial and relational interfaces is a difficult task with current resources.

Raster images, arguably the most common format for creating and disseminating data visualizations, currently cannot be made into navigable structures. These are only described using alt text, which limits their usefulness to screen reader users.

Unfortunately, more accessible rendering formats like SVG with ARIA (accessible rich internet applications) properties are more resource intensive than raster approaches, like WebGL-powered HTML canvas or pre-rendered PNG files. SVG puts a burden on low-bandwidth users and a ceiling on how many data points can be rendered in memory.

In addition, ARIA itself has 2 major limitations. First, when added to interface elements, ARIA only provides *screen reader* access, which means that developers must build a solution from scratch for other navigation input modalities. Second, ARIA's linear navigation structure can be time-consuming for screen reader users if a visualization has many elements. This may impede how essential insights and relationships are understood [49, 86, 144, 152, 162, 194].

Some emerging approaches have sought to address this serial limitation of data navigation and provide richer experiences for screen reader users [49, 152, 162, 194]. However, these approaches rely on a tree-based navigation structure which is often not an appropriate choice for visualizations of relational, spatial, diagrammatic, or geographic data. Many visualization structures are currently unaddressed.

Zong et al. stress that in order to realize richer, more accessible data visualizations, the responsibility must be shared by "toolkit makers," the practitioners who design, build, and maintain

visualization authoring technologies [194]. Our contribution is towards that aim, to make more accessible data experiences easier to design and implement within existing visualization work.

We present Data Navigator. Data Navigator is a toolkit built on a graph data structure, within which a broad array of common data structures can be expressed (including list, tree, graph, relational, spatial, diagrammatic, and geographic structures). Data Navigator also exposes an interface that supports interactions via screen reader, keyboard, gesture-based touch, motion gesture, voice, as well as fabricated and DIY input modalities. Data Navigator provides expressive structure and semantic rendering capabilities as well as the ability for developers to use their own, preferred method of rendering.

Data Navigator builds upon human-studies motivated work on accessible navigation [162, 194] towards a more generalizable resource for visualization practitioners. We contribute a high-level system design for our node-edge graph-based solution as well as an implementation of this system on the web, using JavaScript, HTML, and CSS. Through our case examples we also demonstrate that our generalized approach is suitable for replication of existing best practices from other systems, integration into existing visualization toolkit ecosystems, and development of novel prototypes for accessible navigation. We illustrate how Data Navigator's use of generic edges, dynamic navigation rules, and loose coupling between navigation and visual encodings provides practitioners robust, expressive, control over their system designs.

5.3 Related Work

Our contribution is an attempt to bridge the gap between research and practice more effectively across broad ecosystems in order to enable deeper and more expressive accessible data navigation interfaces. Below we outline the prior research and standards that inform our project, a breakdown of existing visualization toolkit approaches to data navigation, and then accessible input device considerations.

5.3.1 Accessibility research and standards in visualization

Research and standards are both somewhat limited by a strong bias towards visual disabilities. In *Chartability*, 36 of the 50 criteria related to accessible visualization considerations involve visual disabilities [34, 39]. Marriott et al. also found that visual disability considerations are the primary focus of data visualization literature [110], leaving the barriers that many other demographics face unstudied.

However, despite the heavy focus on visual disabilities, the work that does exist in the visualization community is deeply valuable and serves as an important starting point for our technical contribution.

5.3.1.1 Accessible navigation design considerations

Zong et al.'s research, which was conducted as in-depth co-design work and validated in usability studies involving blind participants, presented a design space for accessible, rich screen reader navigation of data visualizations. They organized their design space into *structure, navigation,*

and *description* considerations and demonstrated example *structural*, *spatial*, and *direct* tree-based approaches [194].

Chart Reader also engaged these design space considerations in their co-design work on accessible data navigation structures [162]. We consider these design dimensions as the best starting point for our work, bridging the gap between research and toolkits.

There are additional research projects that have focused on accessible data navigation and interaction [49, 143, 145, 152]. These contributions explore a range of different interaction structures, including lists, trees, and tables of information as well as direct access methods such as voice interface commands and simple, pre-determined questions.

5.3.1.2 Accessible visualization: understanding users

A wide array of emerging research projects investigate screen reader users needs, barriers, and preferences, and offer guidelines, models, and considerations for creating accessible data visualizations [21, 39, 101, 144]. Jung et al. offer guidance to consider the order of information in textual descriptions and during navigation [86]. Kim et al. collected screen reader users' questions when interacting with data visualizations, which could open the door for more natural language data interaction [91].

5.3.1.3 Accessibility standards and guidelines

In the space of research, there has been a growing interest in developing guidelines for practitioners [31, 34] and even applying guidelines as a method of validation alongside human studies evaluations and co-design [39, 101, 102, 194]. Unfortunately, most accessibility standards and guidelines do not explicitly engage how to structure data navigation.

Despite this, existing accessibility standards bodies like the Web Content Accessibility Guidelines do stress the importance of accurate, functional semantics in order for screen reader users to know how to interact with elements [169]. For interactive visualizations this means that button-like or link-like behavior should expressly be made using elements that are semantically buttons and links. Our system should be capable of expressing meaningful semantics to users of assistive technologies.

5.3.2 Visualization toolkits and technical work

Unfortunately while many data visualization toolkits offer some degree of accessible navigation and interaction capabilities to developers, very few toolkits currently out there offer control over the important aspects of accessible data navigation design. Replicating existing research and strategies, remediating toolkit ecosystems, and building novel prototypes are all difficult or impossible to do due to the current lack of toolkit capabilities.

Existing data visualization toolkits have 3 major limitations that we wanted to address in the design of Data Navigator:

1. **Built on visual materials**: toolkits produce either raster or SVG-based visualizations, neither of which are focused towards designing navigable, semantic structures. As a consequence, many visualizations are simply entirely inaccessible.

2. **Lacking relational expressiveness**: When data navigation *is* provided, the navigation is based on either a tree or list structure (see Figure 5.2). The consequence of this limitation is that many other non-list and non-tree data relationships become difficult or impossible to represent without overly tedious navigation or inefficient architecture.
3. **Designed only for screen reader interaction**: When *accessible* data navigation is provided, it is generally only made possible through SVG with ARIA (Accessible Rich Internet Application) attributes. ARIA is primarily only leveraged by screen readers [170]. If a data element can be clicked and performs some form of function, only direct pointer (mouse and touch) and screen reader users are included. The consequence of this is that a wide array of other input devices, many used as assistive technologies by people with motor and dexterity disabilities, are excluded.

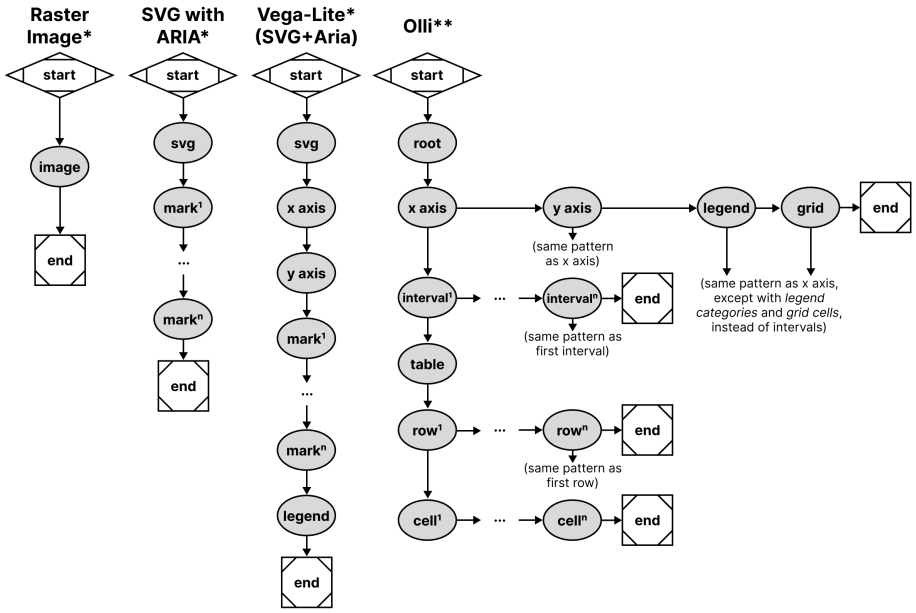

Figure 5.2: Existing accessibility trees and lists, shown using node-edge graph conventions. (*) Denotes only *screen reader* access. (**) Denotes *screen reader*, *keyboard-only*, and *pointer* access as well.

5.3.2.1 Rich, tree-based approaches

De-coupling rendered, visual structures from meaningful and effective navigation experiences can provide richer experiences for screen reader users [194]. Prior research and industry work, with the exception of the *Visa Chart Components* library [165], has relied heavily on a 1 to 1 relationship between structure (the encoded marks) and navigation. This emerging work is significant, because it paves the way for considering the design dimensions of accessible data interaction and navigation without dependence on a visually encoded space.

Olli's approach has been to build ready-to-go adaptors that automatically build multiple tree structures for a few ecosystems (*Vega*, *Vega-Lite*, and *Observable Plot*) and is entirely uncoupled

from a data visualization's graphics. Their approach renders navigable tree structures *underneath* a visualization.

Other than *Olli*, *Highcharts* [72], *Visa Chart Components*, and *Progressive Accessibility Solutions'* visualization toolkits [49, 152] also primarily provide tree and list navigation structures across all of their chart types. These toolkits render their structures *upon* the visualization's graphic space. These tools also provide some degree of support for other assistive technologies and input modalities, although are limited exclusively to SVG rendering.

Unfortunately, these toolkits lack capabilities for dealing with graph, relational, spatial, diagrammatic, and geographic data structures.

5.3.2.2 Serial, list-based approaches

Toolkits like *Vega-Lite* [136] and *Observable Plot* only provide basic screen reader support through ARIA attributes when visualizations are rendered using SVG. These libraries do not currently provide additional access to other assistive technologies and input modalities.

Microsoft's *PowerBI* largely uses a serial structure, although it has tree-like elements as well. *PowerBI* generally provides the same access to keyboard users as it does to screen readers, although not completely.

5.3.2.3 No navigation provided

Other visualization tools, like *ggplot2* or *Datawrapper*, *Tableau*, as well as both *Vega-Lite* and *Highcharts* (when rendering to canvas), produce raster images and have no navigable structure available. Raster, or pixel-based graphics have been an accessibility burden since the early days of graphical user interface development [12]. Practitioners who use these toolkits can only provide alternative text.

5.3.3 Considering assistive technologies and input devices

Modern data visualizations may contain functional capabilities such as the ability to hover, click, select, drag, or perform some analytical tasks over the elements of the visualization space [136]. Virtually all of these analytical capabilities are designed for use with a mouse.

Input device consideration can roughly be organized as either *pointer-based* (such as a mouse or direct touch) or *non-pointer based* (which may employ speech recognition or sequential, discrete navigation such as with a keyboard). Assistive pointer-based devices, such as a head-mounted touch stylus, can typically perform any actions that a mouse can and are therefore served by current interactive visualizations. However, assistive non-pointer devices, such as a tongue, foot, or breath-operated switch, are not.

By only providing pointer-based interactivity, modern interactive visualizations exclude users who leverage non-pointer based input, who are most commonly people with motor and dexterity disabilities. And unfortunately, there is a complete lack of engagement with these populations in the data visualization research community [110].

By comparison, the broader accessibility and HCI research communities have rich engagement with interaction and assistive technologies for users with motor and dexterity disabilities.

Most research either focuses broadly on physical peripheral devices or sensors [150], wearables [135], or DIY making and fabrication [78].

The DIY making space involves a broad spectrum of complex input devices and materials, such as fabricating with wood and sensors for children with disabilities [97], 3D printed materials for rehabilitation professionals [47], and even using produce-based input (such as bananas and cucumbers) for aging populations [129].

Broadly, both research and practical developments related to accessible, non-pointer input are much further ahead than data visualization research and practice. Our goal for Data Navigator is to provide a technical resource towards engaging this under-addressed space.

5.4 Data Navigator: System Design

We categorized our system design goals into design considerations for *Structure*, *Input*, and *Rendering*:

1. **Generic structure and navigation specification**: Human studies work has validated that lists, tables, trees, and even pseudo-treelike and direct structure types are all valuable to users in different contexts and with different considerations. Our system must be able to work with all of these as well as less frequently-used structures (spatial, relational, geographic, graph, and diagrammatic).
2. **Robust input handling**: Blind and low vision users may use combinations of different assistive technologies, such as magnifiers, voice interfaces, and screen readers. Users with motor impairments may rely on voice, gesture, eye-tracking, keyboard-interface peripherals (like sip-and-puffs or switches), or fabricated devices. Both the developer and user should therefore be able to leverage and customize a broad range of input types, including the above as well as fabricated, adaptive, and future input modalities.
3. **Flexible rendering and semantics**: Visuals may or may not be necessary to render to demonstrate Data Navigator's structure. In addition, much of the latest research has shown that different screen reader users may prefer different orders of information and at different levels of verbosity. In addition, the context of tasks the user is performing as well as the nature of the data itself may influence the design of semantic descriptions and visual indications for elements. Data Navigator must provide a high degree of flexibility and control.

To help bridge the gaps between research and standards knowledge about best practices and building an effective toolkit for practitioners, we intend for Data Navigator to provide both *exploratory support* and *vocabulary correspondence* [103].

In particular, our ideal users are developers who specify data visualizations using code. To that aim, we intend to provide *exploratory support* through generic, dynamic, and flexible system design decisions. Our system is expressive and customizable, which encourages exploration of different options.

And we also want the API to include properties that have conceptual and *vocabulary correspondence* to our design considerations. Each design consideration (*Structure*, *Input*, and *Rendering*) are separately composable, modular subsystems of Data Navigator that can be used independently or in tandem with one another.

In this paper we present an implementation of our system using JavaScript, HTML, and CSS on the web. The demonstration of our system is best suited to the web due to the nature of existing, accessible building blocks (HTML), which resolve many of the semantic complexities and logic involved in enabling screen readers to programmatically navigate and announce meaningful information to users. In addition, many existing visualization toolkits target the web as an output platform and we believe that this is the best starting point for adoption and use of Data Navigator. However, this system design could be implemented as a toolkit in other environments with proper consideration for input device handling and screen reader semantics.

5.4.1 Structure

5.4.1.1 Beyond trees: towards an accessibility *graph*

The first major contribution in the design of Data Navigator is to use node-edge data as the substrate for our navigation system.

The most important argument in favor of using a graph-based approach is that a graph can construct virtually any other data structure type (see Figure 5.2), including list, table, tree, spatial, geographic, and diagrammatic structures. Graphs are generic, which enables them to represent structures both in current and future interface practices [46].

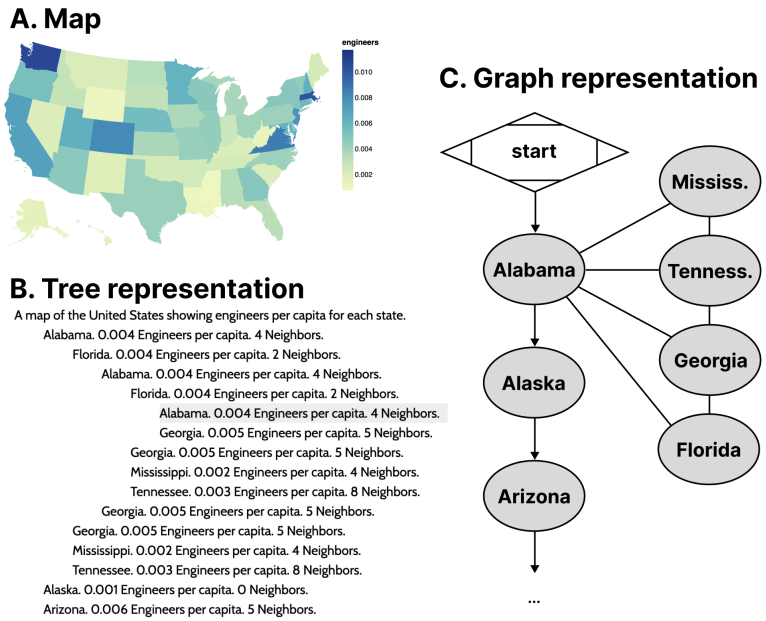

Figure 5.3: **A.** Map of engineers per capita of US states. **B.** Tree representation of the map data where states are listed alphabetically and also include links to neighboring states. The structure repeats itself if users navigate in a loop. **C.** Graph representation with the same navigation potential without redundant rendering.

To demonstrate our point, the most recent emerging work with advancements in accessible data navigation used node-edge diagrams to demonstrate their tree-like structures [162, 194]

similar to Figure 5.2, Figure 5.9, and Figure 5.10. This is because trees are a form of node-edge graph, but with a root, siblings, parents, and children as sub-types of nodes that generally have rules for how they relate to one another.

Node-edge graph structures prioritize direct relationships. Examples of common direct relationships in visualization are boundaries on maps (see Figure 5.3), flows and cycles, data with multiple high level tree structures pointing to the same child datasets (such as *Olli* in Figure 5.2), or even just in diagrammatic, graph-based visualizations.

A graph structure allows for direct access between information elements that are not just part of the input data or 1:1 rendered elements, but may also have perceptual or human-attributed meaning. Examples of this might include semantic or task-based relationships, such as navigating to annotations or callouts, between visual-analytic features like trends, comparisons, or outliers. Spatial layouts such as intersections of sets or parallel vectors (see Section 5.5.3), or even relationships to information outside of a visualization and back into it (like in Figure 5.7) are enabled by a graph structure.

5.4.1.2 Graph structures are more computationally efficient

Data visualizations often portray information that becomes difficult to handle when using trees and lists. The distance users must travel between relational elements is significant in lists while redundancy when navigating relational elements in trees can be problematic.

As an example of this, often a data table or list of locations are used in conjunction to a map, such as listing all 50 states alphabetically along with relevant information. The list itself is expensive to navigate and may not provide any relationship information about which states border others, let alone ways to easily and directly access those states.

Part of the visual design justification of using a map instead of a table is for sighted individuals to understand how geospatial information may interact with a given variable. The spatial relationships matter. But when supplementing the list of states with sub-lists for each state's bordering states (see Figure 5.3), it produces redundancy in the rendered result. The rendered data contains circular connections between nodes but must render every reference, producing a computational resource creep and cluttered user experience that can be difficult to exit.

5.4.1.3 Specific edge instances and generic edges

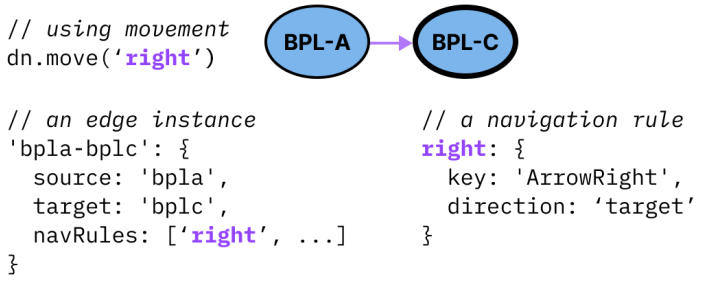

Figure 5.4: An example of how a single edge instance references a navigation rule and can even have multiple navigation rules. A navigation rule can be referenced by multiple edges.

In Data Navigator, nodes are *objects* that always contain a set of edges, where each edge contains a minimum of 4 pieces of information: a unique identifier, a source, a target, and navigation rules. These properties are only accessed when a navigation event occurs on a node with an edge that contains a reference to a rule for that navigation event. Navigation rules may be unique to an edge instance or shared among other edge instances.

The source and target properties of edges are either ids that reference node instances (see Figure 5.4) or *functions* (see Figure 5.5). Because some edges in a graph may be directed or not, non-directed graphs can use source and target properties to arbitrarily refer to either node attached to an edge.

Generic functions for source or target properties can link nodes to other nodes based on changing content, structure, or behavior that may be difficult or impossible to determine before a user navigates the structure.

Function calling also allows some edges to be *purely* generic. An example of a reasonable use case of a purely generic edge is in Figure 5.5, where the source is a function which returns the present node and the target is whichever node the user was on previously. This single edge may then be part of every node's set of edges, enabling users to have a simple *undo* navigation control without creating an *undo* edge unique to every source node.

Using this pattern, it is possible to have fully navigable structures using only generic edges.

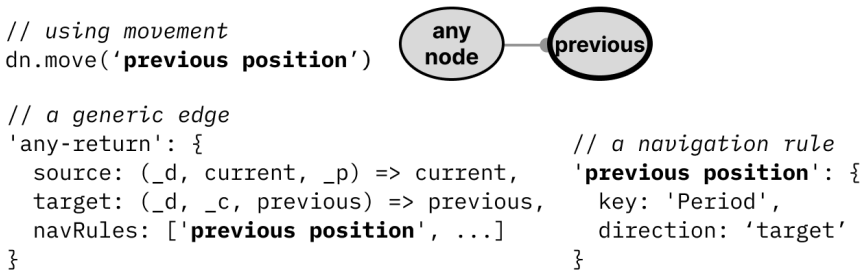

Figure 5.5: A generic edge, such as "any-return" can be applied to any node. Function calls handle dynamically assigning the edge's source and target nodes on-demand.

5.4.2 Input

5.4.2.1 Abstracted navigation facilitates agnostic input

Navigation rules in Data Navigator (see Figure 5.4 and Figure 5.5) are created alongside the node-edge structure. Edges reference rules for navigation. However, these rules are generic and agnostic to the specifics of input modalities and can be invoked as methods by virtually any detected user input event (see Figure 5.6).

Navigation rules are objects with a unique name, ideally as a noun or verb in natural language that refers to a direction or location, a movement direction (a binary used to determine moving towards the source or target of an edge), and optionally any known user inputs that activate that navigation, such as a keyboard keypress event name.

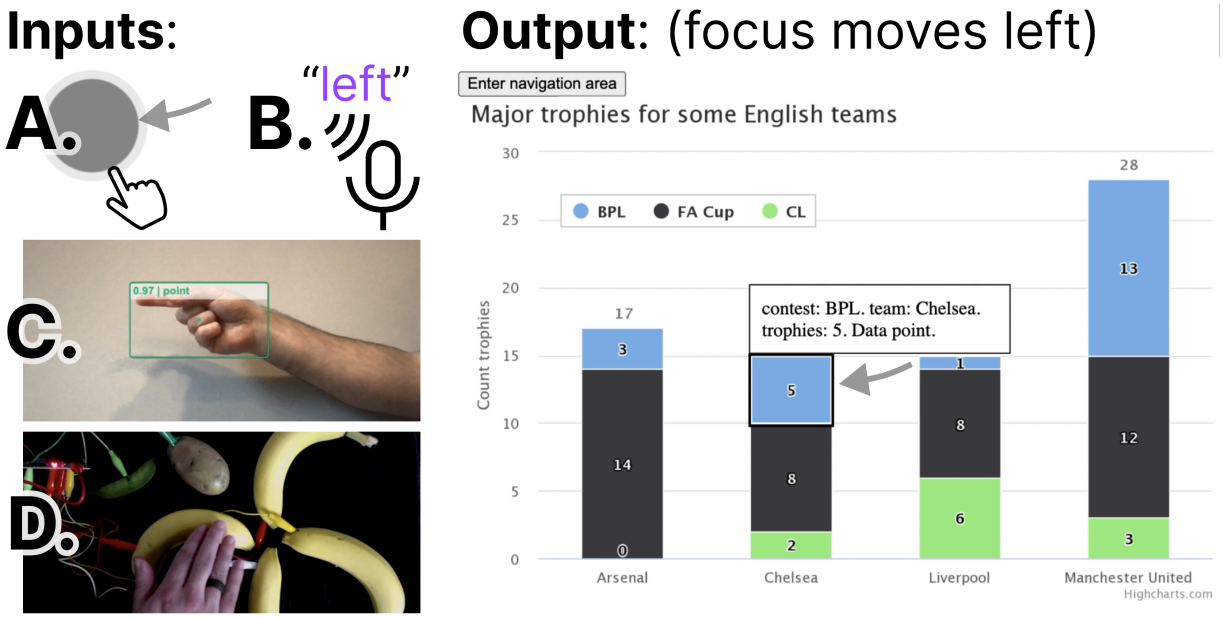

Figure 5.6: An example navigation rule to move "left" can be called as a method by an event from any input modality. Some examples include common modalities such as touch swiping (**A**) or speaking "left" (**B**). This also includes advanced or future modalities such as gesture recognition (**C**) or touch-activated, fabricated interfaces (**D**).

It is important for a system to abstract navigation events so that inputs can be uncoupled from the logic of Data Navigator. This allows higher level software or hardware logic to handle input validation while Data Navigator is just responsible for acting on validated input.

Later in our first case example (Section 5.5.1), we demonstrate an application that handles screen reader, keyboard, mouse and touch (pointer) swiping, hand gestures, typed text, and speech recognition input. Abstract navigation namespaces can be called by any of these input methods.

Additionally, since navigation rules are flexible, end users can also supply their own key-bind remapping preferences or input validation rules if developers provide them with an interface.

Because calling a navigation method is abstract, users can even supply events from their own input modalities as long they have access to either a text input interface or access to Data Navigator's navigation methods. Our demonstration material (in Section 5.5.1) also includes handling for DIY fabricated interfaces, which are important in accessibility maker spaces. We chose a produce-based interface [129], since it was an easy and low cost proof of concept.

We believe that enabling agnostic input provides a rich space for future research projects. In addition, browser addons and assistive technologies could both leverage this flexible interface for end users.

5.4.2.2 Discrete, sequential input opens new avenues

The *keyboard interface* is considered foundational for many assistive devices, which leverage this technology for discrete, sequential, non-pointer navigation and interaction [171]. Desktop

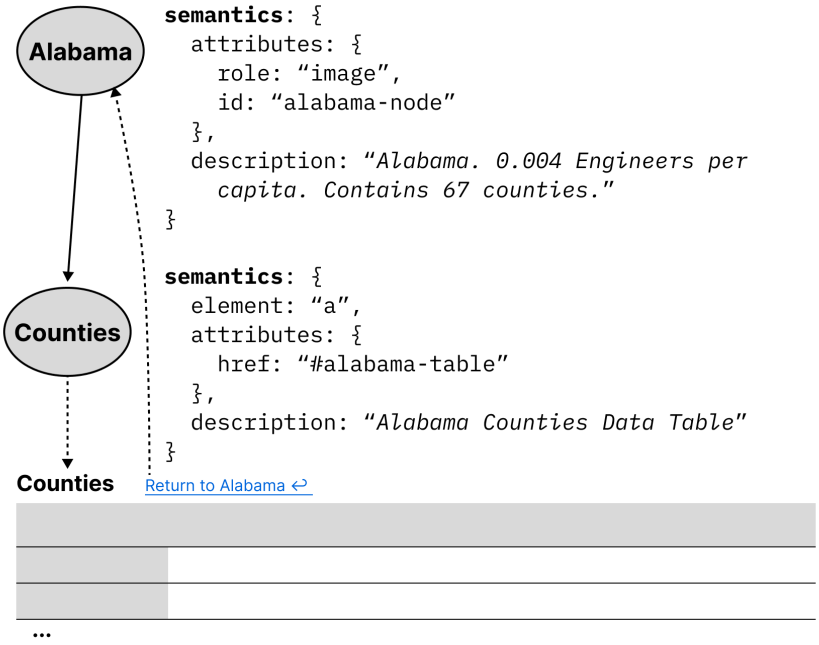

Figure 5.7: An example of how navigation within Data Navigator could use semantic nodes as hyperlinks to provide access to other areas in an application. Alabama has a child node “Counties” which is a semantic HTML link element pointing to a table of counties, outside of Data Navigator’s graph structure. A link is provided to return.

screen readers are the most common example of an assistive technology device that leverages the keyboard interface, however single or limited button switches, sip-and-puff devices, on-screen keyboards, and many refreshable braille displays do as well. Support for the keyboard interface by default in turn provides all discrete, sequential input devices with access as well.

However by basing Data Navigator’s foundational infrastructure on a keyboard-like modality, this also provides designers and developers new avenues to imagine how existing direct, pointer-based, or continuous inputs can map to discrete, sequential navigation experiences.

For example, with mobile screen readers this already happens: screen reader users swipe and tap on their screen to sequentially navigate, but the exact pixel locations of their swiping and tapping generally does not matter. Their current focus position is discrete and determined by the screen reader software.

Data Navigator therefore allows for many new possibilities. One possibility is that sighted mouse and touch users may now also swipe their way through dense plots or use small interfaces (such as on mobile devices) that may otherwise be too hard to precisely tap. Data Navigator optionally removes the accessibility barriers sometimes posed by precision-based input in visualizations.

Data Navigator does not have to be in conflict with precision-based input, either. A discrete, sequential navigation infrastructure can be used in tandem with precision-based pointer events as well as instant access when coupled with voice commands and search features.

5.4.3 Rendering

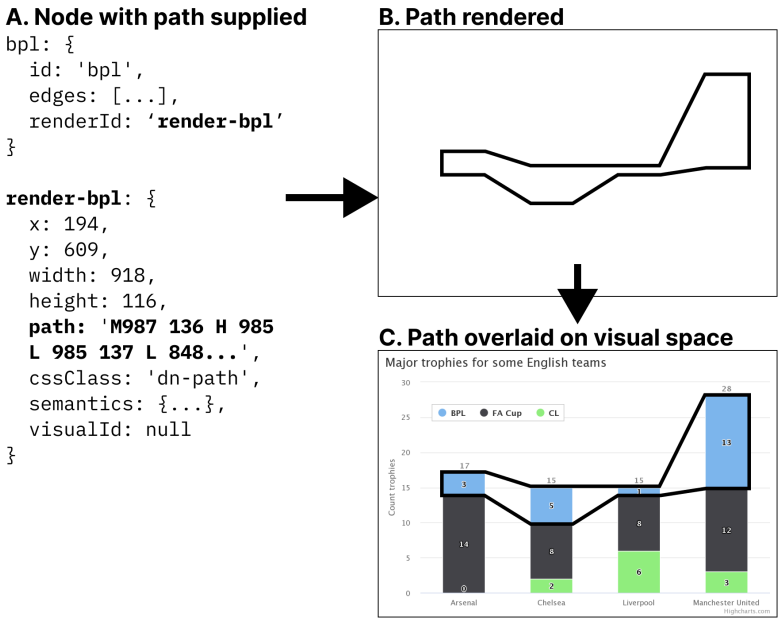

Figure 5.8: **A.** The data specified for a node with a reference to separate data that is used to render that node. **B.** The node will render as a path at the specified Cartesian coordinates. **C.** This rendered node may then be placed over a visual.

5.4.3.1 Flexible node semantics provide freedom

Nodes in Data Navigator are semantically flexible. This is because the marks in a data visualization may represent many things, that are either dependent on the data or the user interface materials.

Since our toolkit implementation is in JavaScript and HTML, our map example from Figure 5.3 might use image semantics for states, alongside a description of the data relevant to that node. However since semantics are flexible in this way, Data Navigator could also be used to integrate into a larger ecosystem, with nodes rendered as hyperlinks to tables or other elements such as in Figure 5.7.

The concept of using node-edge graphs can even extend to have "nodes" that are entirely different parts of a document or tool, as well as integrated into the explicit structure provided by Data Navigator. In some accessibility toolkits, nodes are geometries without functional semantics [136] or list items nested within lists [10]. But in Data Navigator, nodes can semantically be buttons, links, or any HTML element. Interactive data visualizations sometimes demand more flexible node semantics than geometries or lists.

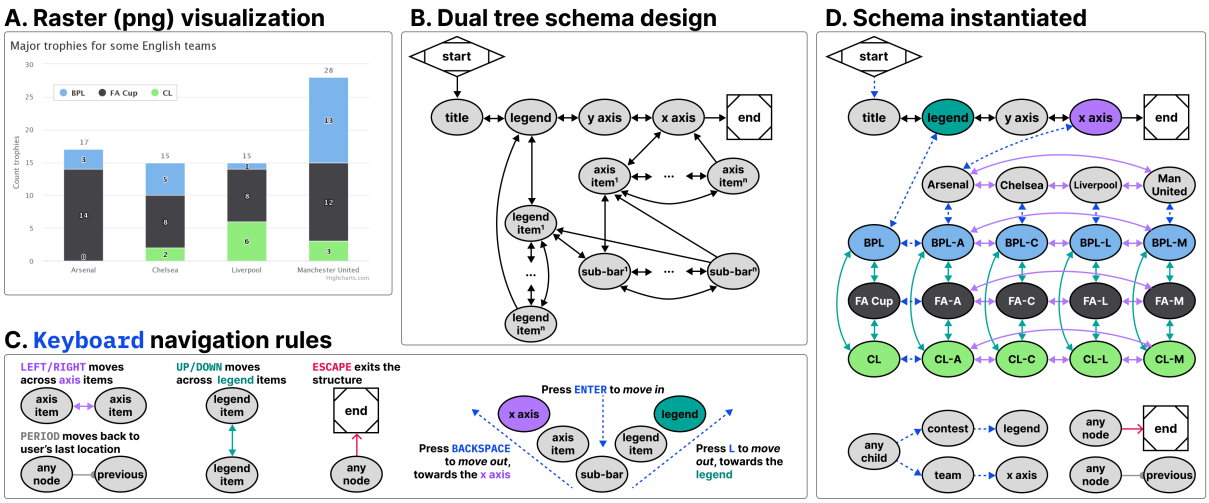

Figure 5.9: **A.** A raster (png) visualization of a stacked bar chart showing how 4 English teams performed across 3 major trophy contests. **B.** An example navigation schema that allows children nodes to have 2 parents (two tree structures intersecting), one for contests and one for teams. **C.** An example of Data Navigator's navigation logic abstraction, which allows edge types to have programmatic sources, targets, and rules, such as a single rule that gives all nodes a edge to exit the visualization. **D.** An instantiation of the schema, showing all corresponding rendered nodes and their edge types according to the schema design and navigation rules.

5.4.3.2 Loose-coupling to visuals enables expressiveness

One of the most significant technical limitations of existing data visualization toolkits with regards to accessibility is that they rely on visual substrate, or visual materials, in order to produce data visualizations. In the case of static, raster images such as png files or WebGL and canvas elements on the web, there are no interface properties at all exposed to screen readers for programmatic exploration and interaction.

If raster images are used, they generally cannot be changed after rendering. However, according to web accessibility standards, elements must have a visual indicator provided when focused [168].

Since Data Navigator navigates using focus, an indicator must be rendered alongside the node semantics. But *what* is focused visually and *where* it is depends on different design needs.

In *Visa Chart Components*, chart elements can be *selected*, so the focus indication is visible over the existing elements in the chart space. The design choice to have interactive visual elements located within a chart or graph is also common in other toolkits that provide accessible focus indication, such as *Highcharts*, *PowerBI*, and *SAS Graphics Accelerator*.

However, some visualization toolkits create accessible structures entirely uncoupled from visual space [10], so focus indication is provided beneath or beside the chart, not over it.

Due to the different ways that accessibility might be provided, Data Navigator enables developers to have complete control over the rendering of which focus elements they want, in what styling, and where. This can accommodate both un-coupled and visually-coupled approaches to focusing and more.

Data Navigator's focus is *uncoupled* by default and may even be used independent of any existing graphics at all. Rendering information may be passed to Data Navigator for it to render (like in Figure 5.8) or developers can provide their own rendered elements and simply use Data Navigator to move between them.

Because of Data Navigator's approach to rendering focusable elements, designers and developers can provide fully customized annotations, graphics, text, or marks that may not be not part of the original visual space or elements. One example of this might be adding an outlined path to a collective cross-stack group of bars in a stacked bar chart (see Figure 5.8).

Loose-coupling in this way provides robust flexibility to designers and developers to handle navigation paths and stories through a data visualization, even in bespoke or hand-crafted ways.

5.4.3.3 On-demand node rendering is efficient

Practitioners care about performance and so do users. Practitioner toolkits often focus on lazy-loading techniques where accessibility elements are rendered on-demand rather than all in-memory up front [10, 32, 194].

Data Navigator's nodes are rendered *on-demand* by default. Data Navigator only renders the node that is about to be focused by the user and after it is focused, the previously focused node is deleted from memory. This technique has advantages in cases where datasets are large or users have lower computational bandwidth available. However, there are cases where practitioners may want to render all of Data Navigator's structure in memory, such as server-side rendering or equivalent. Pre-rendering may be optionally enabled.

5.5 Case Examples with Data Navigator

We built example prototypes using our JavaScript implementation of Data Navigator, available open source at our GitHub repository.

Our first two prototype case examples represent some of the most powerful parts of Data Navigator as a system while reproducing known and effective data navigation patterns from existing industry and research projects. We provide a final case example as a co-design session that demonstrates how Data Navigator may be used to rapidly build new designs.

5.5.1 Augmenting a Static, Raster Visualization

The first case example (shown in Figure 5.9) builds on an online JavaScript visualization library, *Highcharts*. *Highcharts* already provides relatively robust data navigation handling out of the box for screen reader, keyboard, and even voice recognition interface technologies, such as *Dragon Naturally Speaking*. However, these capabilities are only provided when the chart is rendered using SVG. Developers have several other rendering options available, including WebGL, which is significantly more efficient [73]. We wanted to demonstrate that Data Navigator can provide a navigable data structure even if the underlying visualization is a raster image.

For our case example, we exported a png file using the built in menu of a sample stacked bar chart retrieved from their online demos [71]. We selected a stacked bar chart because it allows

us to demonstrate how two tree structures may interact and share the same children nodes.

We recorded the data and hand-created all of the geometries and their spatial coordinates using *Figma*, by tracing lines over the raster image's geometries (see samples of the data and traced geometries in Figure 5.8). While this method was efficient for building an initial prototype, Section 5.5.2 engages deterministic methods for extracting and producing the nodes, edges, and descriptions required by Data Navigator automatically and at scale.

The visualization we selected represents 4 English football teams, *Arsenal, Chelsea, Liverpool,* and *Manchester United* and how many trophies they won across 3 contests, *BPL, FA Cup,* and *CL*.

We chose a schema design that arranged the *contests* to be navigable across one dimension of movement (*up* and *down*) while the *teams* are navigable across a perpendicular dimension of movement (*left* and *right*). This 2-axis style of navigation is used by *Highcharts* (when rendering as SVG) and *Visa Chart Components*. We also chose these directions because it is coincidental that their visual affordance is closely coupled with the navigation design (the x axis is ordered *left* to *right* and since the bars are stacked, *up* and *down* can move within the stack). These directions can also be applied to the axis categories and legend categories as well, moving *left* and *right* across the entire *team's* stacks or up and down across the entire *contest's* groupings.

Using a keyboard, a user might enter this schema and navigate to the legend, where they could press *Enter* to then focus the legend's first child, pictured in Figure 5.8. Pressing up or down navigates in a circular fashion among the *contest* groupings. Pressing *Enter* again then focuses the first child element of that *contest*, all of which are in the *Arsenal* group, since it is the first group along the x axis. A user can then navigate *up, down, left,* and *right* among children. Pressing *L Key* moves the user back up towards the contest while pressing *Backspace* moves the user up towards the x axis. The x axis and *team* groupings represent the second tree which intersects the first (the *contests*).

Our first case example includes handling for additional input modalities beyond screen readers and keyboards, including a hand gesture recognition model, swipe-based touch navigation, and text input (which can be controlled using voice recognition software).

5.5.1.1 Discussion

Our first case example demonstrates several of the most important capabilities of Data Navigator, namely that practitioners can add accessible navigation to previously inaccessible, static, raster image formats and that a wide variety of input modalities are supported easily.

Widely-used toolkits like *Vega-Lite*, *Highcharts*, and *D3* [11] allow practitioners to choose SVG and canvas-based rendering methods. Data Navigator's affordances help overcome the lack of semantic structure in canvas-based rendering, allowing developers to take advantage of its processing and memory efficiency.

Notably in addition to these capabilities, the visual focus highlighting added was entirely bespoke (as in Figure 5.8) and the navigation paths through the visual were based on our design intentions, not an extracted view or underlying architecture such as render order. This demonstrates that our system provides a significant degree of freedom and control for designers and developers.

As a final discussion point, the resulting visualization contains no automatically detectable

accessibility conformance failures according to the W3C's Web Accessibility Initiative's accessibility evaluation tool, *WAVE* [173]. It is important for any technology developed to also meet minimum requirements for accessibility [34, 101, 102, 194], even when following best-practices and research.

5.5.2 Building Data Navigation for a Toolkit Ecosystem

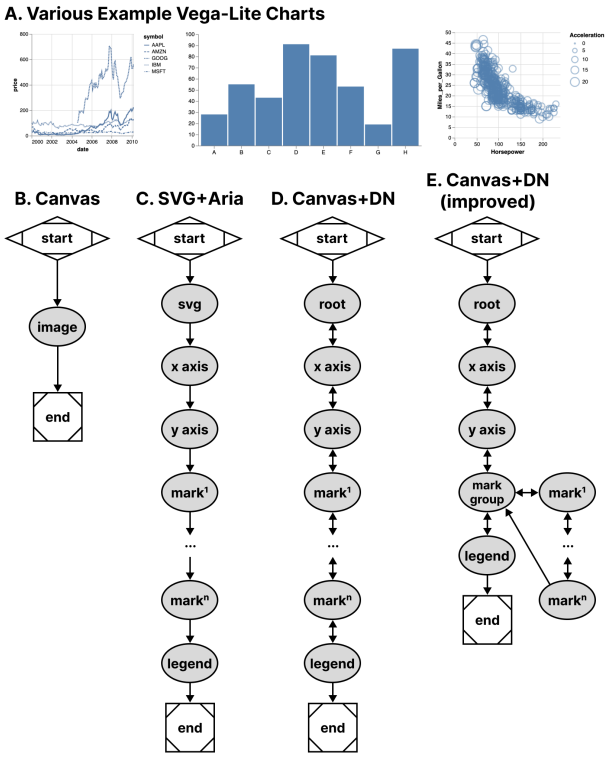

Figure 5.10: **A.** Various charts from *Vega-Lite* share the same general structures with each other when rendered using canvas (**B**) or SVG (**C**). **D.** With Data Navigator, we replicated the existing SVG navigation pattern (**C**) but used a canvas-based rendering for the visualization. **E.** We also improved the navigation scheme to nest marks within a mark group to allow users to skip them, if needed.

Our second case example, shown in Figure 5.10, builds on *Vega-Lite*. As shown in Figure 5.2, *Vega-Lite* offers basic screen reader navigation but provides no navigation at all when rendered using canvas.

While it might be a tedious design choice to allow every mark in a visualization to be serially accessible to screen reader users, we nevertheless set out to build a generic ingestion function that would take a *Vega-Lite View* object and deterministically recreate their existing SVG navigation structure in Data Navigator. This way users would have the same experience between SVG rendered charts and all current and future rendering options that *Vega-Lite* offers to developers.

Notably, *Vega-Lite* does not explicitly manipulate the navigation order at all when rendering with SVG. ARIA is simply provided to allow screen reader users to access each mark in the

visualization in the order the mark appears in the DOM (which is the order it was rendered). The legend appears after the marks in our schema for this reason because *Vega-Lite* renders the legend after marks. This choice of ordering is for visual reasons: z-axis placement is currently based on render order in SVG and *Vega-Lite* wants their legend visually on top of the rendered marks.

In addition to mimicking their existing SVG navigation strategy, we also created a way to nest all of the marks within a group so that users can skip past them and drill in on-demand, which is a valuable pattern when dealing with situations where providing a mark-level fidelity of information may not be relevant to a user's needs by default [148, 194].

In order to deterministically supply Data Navigator with accurate information about any given *Vega-Lite* visualization, we built 3 functions: one that takes a *Vega-Lite View* as input and extracts meaningful nodes, one that produces edges based on those nodes, and one to describe our nodes in a meaningful way for screen reader users. These generic functions technically work on all existing *Vega-Lite* charts, however some are more useful out of the box than others due to the type of marks involved.

5.5.2.1 Discussion

This case example demonstrates that ecosystem-level remediation and customization is not only possible for toolkit builders but Data Navigator offers robust potential. Data Navigator's structure, input, and rendering capabilities are all flexible and can be adjusted to suit the needs of a specific toolkit's design and intended use.

Many visualization libraries may not even provide screen reader accessible SVG using ARIA-based approaches but do have a consistent underlying architectural pattern. Some libraries have a consistent method for converting data into visual formats, readable text labels, and interaction logic. Strong contenders would be visualization libraries popular in online, web-based data science notebooks like *ggplot2* in R or *matplotlib* for Python, which typically only render rasterized pngs or semanticless SVG.

Toolkits with consistent underlying architecture would allow toolkit developers, not just developers who *use* toolkits, to remediate and customize their navigation accessibility using a generic approach.

Enabling accessibility at the toolkit level allows all downstream use of that tool to have better defaults, options, and resources available for building more accessible outcomes for end users.

Many libraries and toolkits provide users with a level of functional defaults and abstract conciseness so that users don't have to worry about low-level geometric considerations [136].

Data Navigator allows toolkits developers to also provide their users with abstractions and defaults for accessibility that make sense for their ecosystem.

Despite our schema recreating a screen reader experience based on SVG (and improving it), Data Navigator's additional features also apply: users are able to leverage a much wider array of input modalities.

Vega-Lite provides many ways to make marks clickable and even perform complex actions using mouse-based input. While Data Navigator does not engage accessible brush and drag-based inputs, it does provide keyboard-only access by default, which can be used to make events

previously only accessible to mouse clicking available to many other technologies. This is an improvement over *Vega-Lite*'s SVG + ARIA rendering option.

When measuring performance across test datasets containing 406 and 20,300 data points in a scatter plot, Data Navigator increases initialization time by ∼0.45 to ∼1.5ms respectively. Our extraction functions specific to *Vega-Lite* increase initialization between ∼4.8 and ∼8.5ms respectively. Given that our benchmark testing for *Vega-Lite's* SVG rendering initialized in ∼1,800ms for 20,300 data points and canvas in ∼700ms, we do not anticipate that Data Navigator will have a negative impact on performance in most visualization contexts.

5.5.3 Co-designing Novel Data Navigation Prototypes

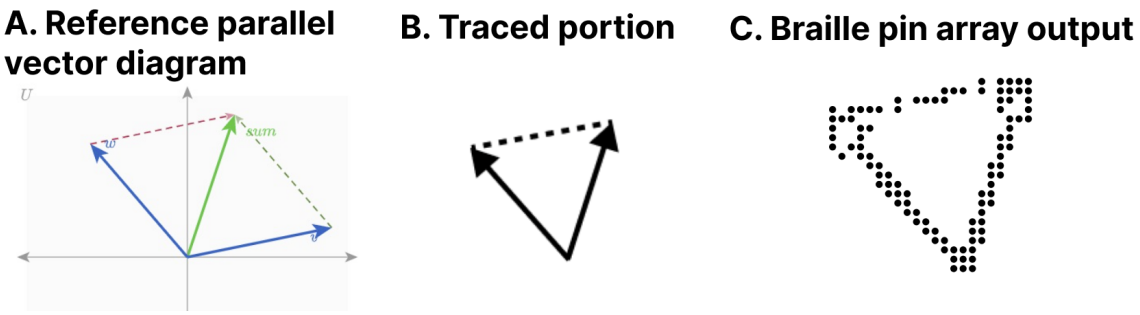

Figure 5.11: Our material preparation process involved taking a reference (**A**), tracing it (**B**), and rendering it on a tactile display (**C**).

Recent projects in accessible data navigation have involved extensive co-design work with people with disabilities, ranging on the magnitude of months with as many as 10 co-designers at a time [101, 102, 162, 194].

However many visualization experiences may be authored in smaller scales, with fewer designers, and less time such as the development of a prototype or demonstration of an emerging idea. In practical or industry contexts, co-design sessions (and design sessions in general) may be much shorter. The goal of these co-design sessions is simply to create an artifact with the artifact's intended users.

Since our paper is contribution towards practical outcomes, we simulated a light co-design session with the aim of producing low-fidelity prototypes of novel data interaction patterns.

5.5.3.1 Co-design Session Methods and Setup

Authors Frank Elavsky (sighted) and Lucas Nadolskis (blind) set out with the goal of developing screen-reader friendly prototypes that can explore geometric and mathematical models produced by the math diagramming tool *Penrose* [190].

Nadolskis is a neuroscience engineer who is a native screen reader user and uses both mathematical concepts as well as data-related tasks in his research. Elavsky proposed a series of possible math-based visualization types produced by *Penrose* to build prototypes for, and Nadolskis selected *set* and *vector* diagrams as the two worth exploring first. The justification for this

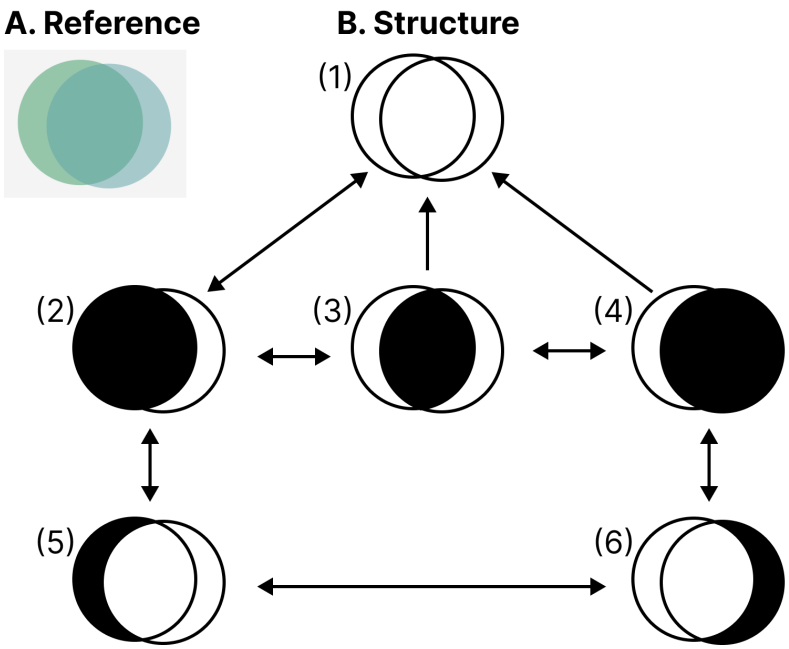

Figure 5.12: **A.** A reference image from *Penrose* of a set diagram containing two sets intersecting. **B.** A diagram of our proposed structure, with three levels of information.

selection is that understanding these two concepts is important for work in data science, programming, and more advanced math concepts.

In particular we grounded the context of our contribution in a hypothetical classroom setting, where a screen reader user who is a student will have access to the equations in both raw text and *MathJax*. We want to provide an experience that does not replace the existing resources screen reader users have to learn in classrooms but rather supplement.

At our disposal for our co-design session was a *Dot Pad* [28], which is a refreshable tactile braille display. Our *Dot Pad* enabled Elavsky to produce something visual and then translate it into the display for Nadolskis. Similar to de Greef et al. [25], we used a tactile interface as an intermediary to help us get a shared sense of the meaningful spatial features of our figures.

Elavsky started with a reference diagram and then traced a wide variety of every possible node that might be worth navigating to in the diagram (see Figure 5.11).

We selected which nodes were most important in each diagram, how to navigate between them, and how we wanted to render their visuals and semantics.

The selection of our problem space, scope of solutions, context of contribution, general discussion, and preparation of materials took approximately 12 hours of work over 2 weeks. The exploration of our prototype design space for our 2 prototypes took 1 hour. Building the prototypes took 2 hours.

5.5.3.2 Creating a Navigable Set Diagram

Our first prototype was a set diagram (see Figure 5.12). For our structure, we decided that it has 3 important semantic levels: the high level, the inclusion level, and the exclusion level. The

inclusion level is first and the siblings are all sets or subsets that include other sets. The exclusion level is beneath and contains sets or subsets that are exclusive to the sets they belong to, which are accessed by drilling down from a set.

Our schema design starts with a user encountering the root level (1) and may optionally drill in to the first child of the next level (2) using the *Enter* key. The user may navigate siblings at this level using *right* and *left* directions, but this level is not circular (like in Figure 5.9) to maintain the spatial relationships. The user may drill in on either set again to view the non-intersecting portion of that set. Any node can drill up, towards the root, using *Escape* or *Backspace*.

5.5.3.3 Creating a Navigable Parallel Vectors Diagram

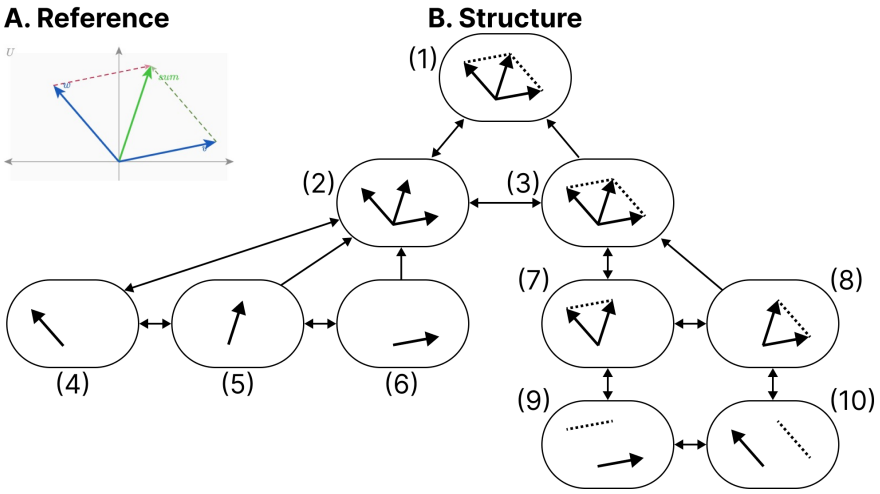

Figure 5.13: **A.** A reference image from *Penrose* of a parallel vectors diagram. **B.** A diagram of our proposed structure, with two main sub-categories of information: understanding the vectors and their parallels.

Our second prototype was a parallel vectors diagram (see Figure 5.13). For the structure of this diagram we created a first level group that contains each vector and vector sum. The sibling to this grouping is another group which organizes sub-equations related to calculating each parallel vector. The sub equations each contain children that pair the sub equation with the vector it is parallel to.

Similar to Figure 5.12, this figure maintains spatial relationships along the x dimension, does not have circular navigation, and allows drilling in and out.

5.5.3.4 Discussion

After our co-design sessions, our visual materials and navigation structures were used in the creation of functional prototypes. We additionally hand-crafted the descriptions and semantics for each node.

Accessibility work often takes a long time, from co-design to building to validation. But we believe that a well-articulated and useful design space, with tools that provide expressiveness

and control over the dimensions of that design space, can improve how this work is done. The above case example demonstrates how builders who are thinking about data navigation design can rapidly scaffold prototypes for use in Data Navigator.

In particular, Data Navigator's design as a system gave our co-design sessions *vocabulary correspondence*. Data Navigator's language helped us focus on the *nodes, edges,* and *navigation rules* for our *structure* while we also explicitly discussed the *rendering* details of *coordinates, shapes, styling,* and *semantics* for each node. The vocabulary of our design space directly corresponded with code details required to create a functional prototype.

We note that this co-design work is not intended to contribute a *validated* set of designs. Rather, our contribution with this case example is to demonstrate that within the larger ecosystem of a research venture, Data Navigator is an improvement over designing and building navigable structures from scratch.

5.6 Limitations and Future Work

Data Navigator is a technical contribution, a system designed for appropriation [29] and adaptation [183] in different applied contexts. It is, as Louridas writes, a *technical material*: a technology that enables new and useful capabilities [100]. While beyond the scope of the current paper, a critical next step for future work is to conduct separate studies with both practitioners and end users to evaluate Data Navigator's affordances.

Unlike toolkits that provide an end-to-end development pipeline for accessible visualization, Data Navigator serves as a low-level building block or material (like concrete). As such, one potential limitation of the framework is that it can be used to build both curbs (which are inaccessible) as well as ramps and *curb-cuts* (which may be more broadly accessible).

Even when building more accessible curb-cuts, we stress the importance of actively involving people with disabilities in the design and validation of new ideas, in line with prior work [101, 102, 127, 194]. For example, while our first two case examples replicate co-designed and validated existing work, our third case example's co-designed prototypes would need to be validated with relevant stakeholders before wider implementation. Our system does not *guarantee* any sort of accessibility on its own.

The diverse array of modalities supported by Data Navigator opens an immediate line of future work in engaging people with a correspondingly diverse set of disabilities. While recent explorations into accessible data visualization have been inspiring, this trend has primarily focused on the experiences of people with visual disabilities [34, 92, 110]. More research should be conducted with other populations, particularly people who leverage assistive technologies beyond screen readers, to understand how interactive data visualizations can be better designed to serve them.

Finally, there are significant opportunities to improve the efficiency of our approach, including developing deterministic and non-deterministic methods to generate node-edge data and navigation rules from a visualization. Ma'ayan et al. stress in particular that reducing tedious complexity can contribute to the success of a well-designed toolkit [103]. Future work should identify areas where graphical interface tools or higher-level specifications can improve the experience of working with Data Navigator.

5.7 Conclusion

Practitioners at large continue to produce inaccessible interactive data visualizations, excluding people with disabilities. We believe that the burden of remediation first starts with the developers who build and maintain the toolkits that practitioners use.

However, the challenges faced by toolkit builders are significant. Most toolkits lack an underlying, navigable structure, support for broad input modalities used by people with disabilities, and meaningful, semantic rendering.

To engage these limitations we present Data Navigator, a technical contribution that builds on existing work towards a more generalizable accessibility-centered toolkit for creating data navigation interfaces. Data Navigator is designed for use by practitioners who both build and use existing toolkits and want a tool to make their data visualizations and interfaces more accessible.

We contribute a high-level system design for our node-edge graph-based approach that can be used to build data structures that are navigable by a wide array of assistive technologies and input modalities. Data Navigator is generic and can scaffold list, tree, graph, relational, spatial, diagrammatic, and geographic types of data structures common to data visualization.

Our system is designed to encourage both remediation of existing inaccessible systems and visualization formats as well as help scaffold the design of novel, future projects. We look forward to further research that explores the possibilities enabled by Data Navigator.

Chapter 6
Skeleton: Visual Authoring of Non-visual Data Experiences

This chapter was adapted from my paper, currently under review with IEEE VIS:

F. Elavsky, C. Nnadozie, L. Nadolskis, P. Carrington, and D. Moritz, '*Skeleton*: Visual Authoring of Non-visual Data Experiences', *IEEE Transactions on Visualization and Computer Graphics*, 2026.

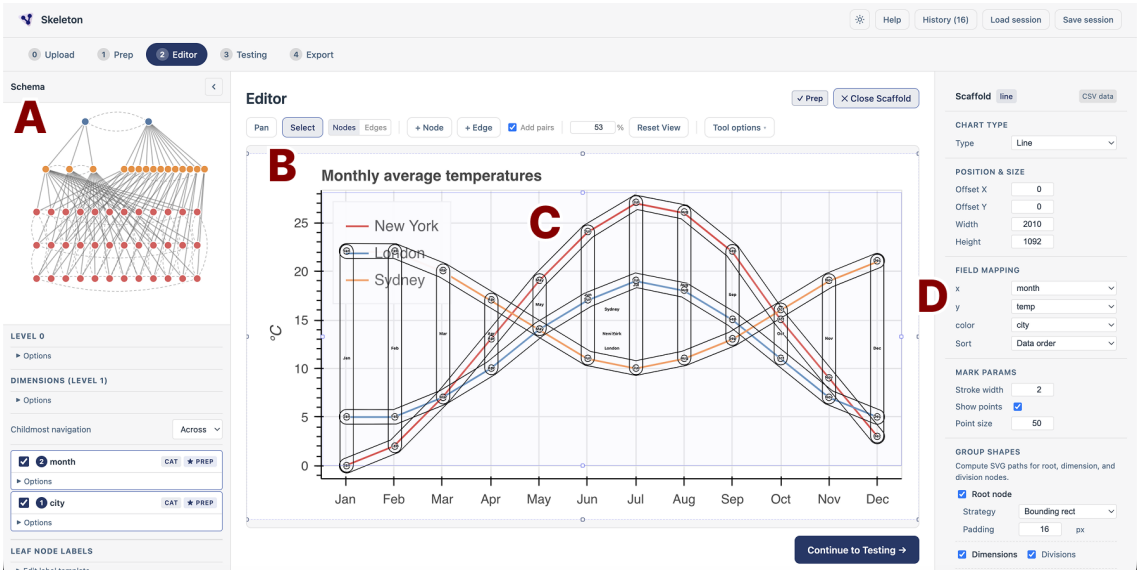

Figure 6.1: Low-fidelity design draft of *Skeleton*'s main user interface components and interactions. A. *Skeleton*, our graphical user interface for creating and debugging screen reader navigation experiences of data visualizations. B. Users can add nodes wherever they want over the chart, manually or automatically with algorithmic assistance. C. Users can then "draw" edges between nodes, which signify navigation paths through the visualization.

6.1 Abstract

When sighted practitioners author accessible data visualizations, they build navigation structures (the nodes, edges, and input bindings that govern how assistive technologies traverse an interface) entirely in code, with no visual representation. This invisibility makes navigation structures difficult to inspect, debug, and iterate on. To sighted practitioners, every other aspect of a visualization is iterated on because it is visible; navigation structure ships as a first draft, if at all,

because it is not. Without a representation to react to, practitioners cannot develop judgment about what makes navigation good or bad, and the quality ceiling of non-visual experiences is set by the absence of a feedback loop. We address this problem through longitudinal co-design with practitioners across cartography, design systems, and open-source visualization, and make three contributions. First, we introduce technical advancements for making the properties of accessible navigation structure visible and directly manipulable during authoring, grounded in two foundational pieces of infrastructure produced by our co-design work: an *Inspector* that renders navigation graphs as interactive node-link diagrams, and a *Dimensions API* that expresses navigation in terms of data dimensions rather than explicit graph construction. Second, building on these, we present *Skeleton*, a direct-manipulation authoring environment in which the properties of an accessible navigation structure are translated into visual representations authors can observe and manipulate. Key techniques include a dual-view editor that simultaneously shows the system's navigation model and the end user's spatial experience, a scaffolding engine that automates spatial node placement by repurposing a visualization rendering pipeline, a live label-template editor with real-time screen-reader-output preview, and a testing mode that makes traversal sequence visually trackable. Third, we evaluate *Skeleton* through an in-situ study with 8 practitioners across visualization design, engineering, and research. Making navigation structure visible changed how practitioners engaged with accessible design: they reconsidered the architecture of their own visualizations, attended to a broader range of input modalities, and shifted from treating accessibility as a compliance task to treating it as a design problem.

6.2 Overview

We start this work with a provocation: How might the discipline of visualization help the discipline of accessibility?

Visualization has spent decades developing techniques for a specific class of problem: representing and interacting with information visually. Grammars of visual encoding [114, 136, 175, 178, 195], direct-interaction interfaces [36, 80, 149], and iterative feedback loops between representation and understanding [42, 62, 98] are all methods that enable abstraction and manipulation of the information that underlies the visual representation. We argue these methods have a direct application within the discipline's own accessibility challenges, one that has not yet been explored.

Sighted practitioners who build accessible data visualizations face an unusual authoring problem. The non-visual navigation structures they construct (the nodes, edges, focus states, input bindings, and semantics that govern how assistive technologies traverse a chart) exist only as code. A practitioner can write a navigation hierarchy, but cannot see it, click on a node to inspect what will be announced, observe the spatial relationship between a navigation path and the chart it overlays, and then manipulate its properties through direct interaction. Every other aspect of a visualization has a visible, inspectable representation during authoring: the visual encodings are visible, the layout is visible, the interaction states are visible. And yet, navigation structure is not.

This invisibility has practical consequences. Without a way to see what they are building, sighted practitioners cannot easily catch structural errors, compare design alternatives, and it-

erate. Accessibility becomes downstream of every other design choice, not because practitioners choose to deprioritize it, but because the authoring conditions do not support anything else [85, 146]. The floor and ceiling of non-visual data experiences are constrained by what sighted authors can perceive of their own work.

We engage this space with the following research questions:

R1 (Qualitative, Exploratory): What challenges do sighted practitioners face when designing and engineering navigation structures for accessible visualizations?
R2 (Qualitative, Exploratory): How do sighted authors reason about the non-visual experiences that accompany their visualizations?
R3 (System, Design): How can we make the properties of accessible navigation structure visible and directly manipulable during authoring?
R4 (Qualitative): In what ways does a directly manipulable visual representation of navigation structure change how practitioners find errors and improve upon their designs?

We address these questions through longitudinal co-design with practitioners across cartography, design systems, and open-source visualization, following an action research orientation [61] in which the research team was embedded in each community's active development work. This paper makes three contributions:

First, **we introduce technical advancements** for making the properties of accessible navigation structure visible and directly manipulable during authoring. These techniques are grounded in our co-design collaborations, which produced two foundational pieces of infrastructure: an *Inspector* that renders any navigation graph as an interactive node-link diagram, and a *Dimensions API* that formalizes a declarative grammar for expressing navigation in terms of data dimensions rather than explicit graph construction (**R1**, **R2**, **R3**).

Second, building on this infrastructure, **we present *Skeleton*, a direct-manipulation authoring environment** in which the topology, spatial mapping, semantics, and input logic of an accessible navigation structure are made visible and directly manipulable. *Skeleton* is built on Data Navigator [35], a code-based library for constructing interactive data navigation structures (**R3**). Our intention with *Skeleton* is to continue to develop it towards a usable, practical system beyond the scope of this research.

Third, **we contribute findings from an in-situ interview study** with 8 practitioners across visualization design, engineering, and research. We evaluate Skeleton as a design probe [56, 81] rather than a deployable system, focusing on qualitative shifts in practitioner engagement rather than task performance. We find that making navigation structure visible shifted how participants engaged with accessible design: they reconsidered the architecture of their own visualizations, attended to a broader range of input modalities, and shifted from treating accessibility as a compliance task to treating it as a design problem (**R1**, **R2**, **R4**).

6.3 Related Work

6.3.1 Non-visual Data Experiences

Blind people who rely on assistive technologies interact with data in fundamentally different ways than sighted users who use a direct pointer, like a mouse [92, 110, 144]. A substantial body of research has documented what these experiences look like across modalities, and what it takes to make them meaningful. In the context of "data experiences," this paper focuses specifically on interactive navigation structures, but we briefly survey adjacent modalities to situate our contribution.

Alternative text and natural language. A dominant strand of this work concerns the generation and evaluation of textual descriptions of visualizations [86, 89, 93, 101]. More recent LLM-driven systems and Q/A approaches can caption charts with varying degrees of semantic depth, some at a risk of producing bias [33, 40, 91]. While alt text makes a visualization's message available without sight, it is by nature static: a description conveys what a visualization says, but not how a user might explore or interact with it.

Sonification, haptics, and tactile rendering. Non-visual data experiences extend well beyond text. Sonification encodes data as sound [14, 65, 108], with declarative grammars emerging for authoring these experiences [90]. Haptic and tactile representations offer another channel through refreshable displays, 3D-printed graphics, and multimodal touchscreen interactions [16, 74, 111, 128]. Recent systems integrate multiple non-visual modalities around a single data representation [21, 75, 141].

Interactive navigation structures. The primary focus of our work centers on the state of research related to structured navigation: the traversal of data points, groupings, and interface elements through assistive technologies and keyboard input [35, 152, 171, 194]. Existing systems and interfaces have demonstrated that going beyond static descriptions to support hierarchical, traversable data structures meaningfully improves how blind users can explore and reason about charts [162, 194]. Giving users control over the textual tokens surfaced during navigation improves comprehension and agency [84]. And more recent work has found that perceptually congruent navigation structures for charts and diagrams can improve goal-driven exploration [115].

6.3.2 Authoring Non-visual Data Experiences

Accessible visualization has historically centered the experiences of disabled users, but a parallel and increasingly urgent body of work examines the experiences of the people who build these experiences: visualization designers, engineers, and researchers.

Practitioner challenges. Research consistently finds that sighted visualization practitioners struggle with accessibility [39, 85, 146]. Most visualizations in the wild are inaccessible and designers themselves report lacking guidance, especially for complex and interactive graphics [85, 146]. And screen reader users experience the downstream effects of these gaps: inconsistent structure, poor keyboard support, and information that is present visually but absent in the accessibility tree [39, 144]. The pattern across this work is consistent: the practitioners who build visualizations lack the tools and feedback mechanisms to make non-visual experiences effective, useful, and good.

Across practitioner-centered literature, a recurring finding is that non-visual experiences are treated as downstream of visual design choices, added after the visual representation is finished rather than designed in parallel [85, 102, 146, 196]. This sequencing has consequences: what is navigable and how it is structured is constrained by whatever visual decisions came first.

Authoring-oriented systems. A distinct line of work has focused on building authoring tools and libraries that give practitioners more tractable paths to accessible output. Few visualization tools support the kinds of interactive navigation structures that assistive technology users most benefit from [93]. Of those that do, most rely on code-based approaches [10, 35, 143, 145]. *Umwelt* [196] takes a different and notable approach: it is a structured editing environment where authors specify representations *across* modalities (sonification and visualization) in an integrated interface, where navigation is made available over the visualization using *Olli* [10]. The latest work in this space is *Arboretum*, a tool that provides automatic conversion of diagrams to a tabular structure, navigable structure, and tactile representation [180]. In *Arboretum*, the input visual is treated as the ground truth for the navigation structure, which is treated as downstream output.

Communicating visually, authoring invisibly. There is a revealing irony across this body of related work: research about navigation structures almost invariably communicates those structures visually. Papers such as "rich screen reader experiences" [194], *ChartReader* [162], *Benthic* [115], and *Data Navigator* [35] each use visual node-link diagrams and hierarchical schematics to explain navigation paths to their readers. The same pattern holds in adjacent domains that involve structuring data for navigation, such as PDF remediation [118].

And in authoring-oriented systems, none provide a visual interface through which practitioners can interactively inspect and manipulate navigation structures as a first-class design material. Structure output is either downstream of code or static visuals. Structure is always *derived* or *specified*; indirectly manipulated. Additionally, verification of the structure across all of these systems requires developers to manually navigate using a screen reader after the structure has been authored and rendered, before returning to the upstream visual design space or code.

Navigation structure is understood and communicated visually by sighted researchers and designers, yet built entirely without visual feedback by developers. We seek to address this gap.

6.4 Co-design Foundations

Our research follows an *action research* orientation [61], in which knowledge is generated by engaging with a community to solve a real problem in-situ, alongside them rather than studying it from the outside. These collaborations started from a shared motivation: practitioners needed to make their existing systems accessible to navigational assistive technologies, using Data Navigator [35] as a foundation. Across three projects, we worked with 12 individuals outside our research team. Three blind co-designers shaped the work throughout: CD1 and CD2 (anonymous) and a co-author on this paper, [REDACTED]. CD1's contributions are noted in Section 6.4.1, CD2's are noted in Section 6.5.3. [REDACTED] contributed to early ideation, problem formation, and framing for the project as a whole, helping define the authoring challenges that *Skeleton* addresses, in addition to feedback on study design.

The co-design literature on accessibility has centered people with disabilities as primary de-

sign partners [23, 25, 125, 162, 193, 194], and rightly so. Our co-design takes sighted practitioners as primary partners because the authoring challenges we address happen on the sighted side of the process: it is sighted authors who cannot see what they are building.

Two themes converged across all three engagements, and we present them here before describing the individual projects that surfaced them. First, **practitioners communicated and reasoned about accessible navigation using visual representations**: while designing for language, sound, and structure, our collaborators drew on paper, built wireframes of nodes and edges, and reflected on the design space using visual artifacts. Even when collaborating with blind co-designers, a visual medium was the first language of sighted authors. Second, **development that followed visual design work faced severe iteration barriers**: verifying a navigation experience required building a working code prototype and manually navigating it with a screen reader. The gap between visual design and code-based scaffolding with manual testing produced repeated mistakes, misinterpretations, and abandoned prototypes. Each project below contributed distinct evidence for these themes and surfaced specific requirements that shaped our infrastructure and tooling.

6.4.1 Geologic Map

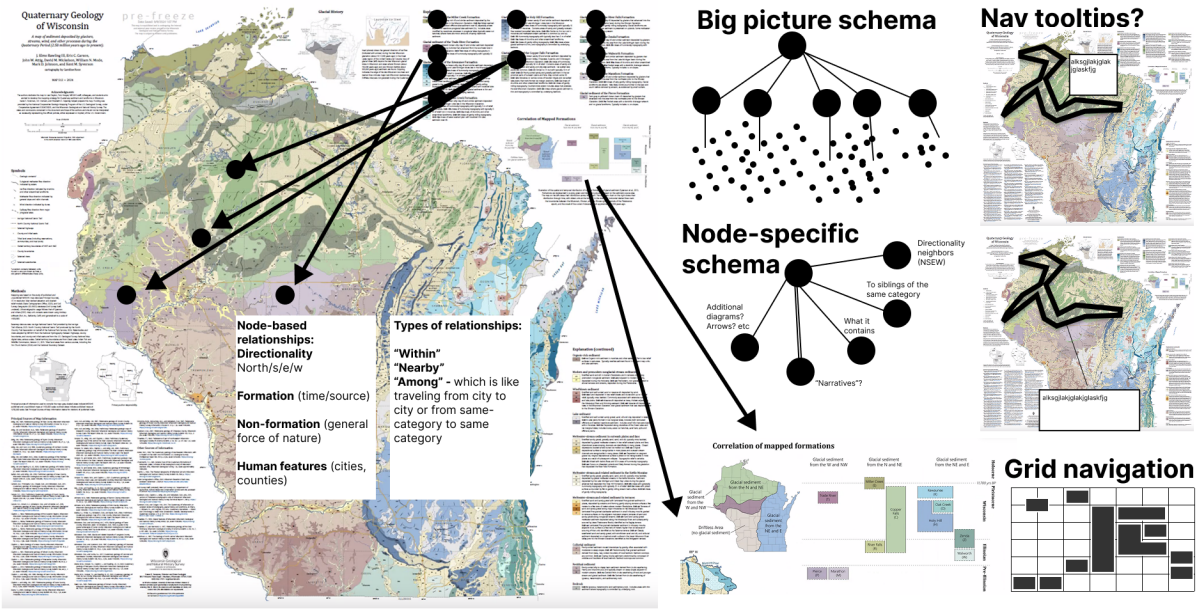

Figure 6.2: Our visual design work in Figma over a static geologic infographic map of Wisconsin. We use visual forms and illustrations over and beside the map to communicate flows, structure, navigation styles, and interaction patterns.

Our longest engagement spanned just over two years, with a cartographer, accessibility consultant, and blind co-designer (CD1) building an interactive version of a quaternary geologic map of Wisconsin [126]. The map combined dozens of irregular geographic regions organized categorically by a legend.

Early design sessions took place in Figma (Figure 6.2), where we laid out node-link diagrams of how a screen reader user might traverse the map, legend, and peripheral information. We annotated each node with text a screen reader would announce and connected them to relevant regions. Drawing the structure made it possible to discuss design choices, catch dead ends, and debate traversal strategies. We were informally doing what *Skeleton* later formalized.

The friction began when we moved from design to implementation. We had no way to verify that our designed structure would be good or ideal, and scaffolding the project into Data Navigator was arduous: the translation from even simple navigational designs to functional code was too complex to hand off or meaningfully iterate on. The collaboration also surfaced a design-space boundary: for within-map spatial navigation, an egocentric audio-game approach [9] fit better than a node-link graph, a distinction CD1 helped surface. Reaching this boundary was only possible because we could see the structure we were trying to build.

6.4.2 Design System Library

Our second engagement, spanning 7 months, was with 6 engineers and designers on Adobe's React Spectrum Charts library, an open-source chart component system. We worked in their codebase and in Miro on navigation design for bar charts, clustered and stacked bars, line charts, and related types. Because we needed generalized, reusable patterns, our design problems differed from the geologic map: we needed to account for use, re-use, and edge cases across chart types.

Our Miro sessions produced two kinds of artifacts: diagrams of navigation structure for specific chart *instances* and *schema* diagrams capturing generalized patterns in dimensional terms. The concept of a "dimension" emerged naturally: common transformations on Data Navigator's graph structure corresponded to properties within a dataset, such as a "categorical" dimension or a "numerical" dimension, where traversal took place within collections of grouped siblings. This dimensional thinking directly foreshadowed the Dimensions API (Section 6.4.4).

The dominant friction was iteration speed. We could sketch and converge on a schema in Miro in an hour, but verifying the design in a functional example required embedding Data Navigator into a large codebase, implementing changes, rebuilding, and manually testing with a screen reader. The collaboration also surfaced a limitation: mobile screen reader navigation uses swipe gestures rather than keyboard input, and our keyboard interaction model did not account for it. Together, these gaps made two requirements clear: a higher-level navigation abstraction and a visual tool for inspecting structures without a screen reader or a fully integrated build.

6.4.3 Open Source Visualization Library

Our third collaboration, spanning nearly two years, was with Quansight Labs and contributors to Bokeh, a Python-based open-source visualization library. We performed an accessibility audit [38] based on Chartability [34] and identified that Bokeh visualizations with interactive chart elements needed to be navigable by assistive technologies.

Unlike Adobe, Bokeh has no native chart types. Its API operates at the level of glyphs, renderers, and data sources, which users assemble freely. There was no standard unit around which to anchor a navigation pattern, and any grammar or tooling would need to accommodate

an open-ended range of encoding combinations. The iteration gap from Adobe was present in a more severe form: making library-wide contributions to a fully open-source project required incremental tooling for the most common cases. For Bokeh, we needed to test functional, data-driven navigation abstractions without fully embedding Data Navigator into the library.

6.4.4 Infrastructure from Practice

The three collaborations converged on two concrete requirements: a way to visually render and inspect navigation structures, and a higher-level abstraction for specifying navigation without hand-wiring every node and edge. We built two pieces of infrastructure to address these requirements. Both were used in subsequent co-design work and became the foundation on which *Skeleton* was built.

6.4.4.1 An Inspector Gadget

We built an *Inspector* (`@data-navigator/inspector`) to render any Data Navigator structure as an interactive node-link graph using D3, with an accompanying console for debugging. Hierarchical structures are colored by level; edges are drawn as directed links; the entry point is visually marked. The *Inspector*'s graph can itself be navigated using Data Navigator, with visual focus tracking during navigation, allowing practitioners to manually verify structure and reachability.

This made structural verification immediate: a practitioner could generate a structure and check at a glance whether the hierarchy had the right levels, whether circular extents produced expected wrap-around edges, or whether a particular path was reachable. The interactive console logs API information and underlying data when nodes are activated, and hovering or focusing logged information highlights the corresponding node in the graph.

The *Inspector* remains a developer tool, however. It requires code familiarity to attach and renders structure as an abstract graph with no connection to the spatial layout of the underlying visualization. It shows topology but not instantiated geometry: a practitioner can see that two nodes are connected but not where their focus indicators will appear on-screen. This gap between navigable structure and its spatial instantiation over a rendered chart motivated *Skeleton*.

6.4.4.2 Alternative Dimensions

The original Data Navigator library requires explicit graph construction: practitioners specify nodes, edges, and navigation rules by hand. This is general but scales poorly.

The *Dimensions API* introduces a declarative abstraction one level above this. Rather than specifying the graph directly, a practitioner describes the *dimensions* of their data, the meaningful axes along which a user might want to navigate, and the API constructs the full node-link structure automatically. A dimension has a type (`categorical` or `numerical`), a data key, and behavioral properties governing traversal. Two properties are central: `extents` determines boundary behavior (`terminal` stops at edges; `circular` wraps around), and `childmostNavigation` determines whether leaf-level nodes are reachable laterally across a dimension's divisions without first returning to a parent.

The generated structure is a multi-level hierarchy: each dimension produces a root node, below which division nodes group the data, below which leaf nodes represent individual data points. Multiple dimensions over the same dataset share leaf nodes, so users navigating via different dimensions reach the same data through different paths. With the Dimensions API, bar chart navigation that would otherwise require constructing every node and edge by hand is expressed as a single dimension declaration:

```
dimensions: {
  values: [
    {
      dimensionKey: 'month',
      type: 'categorical',
      behavior: { extents: 'circular' }
    }
  ]
}
```

The abstraction is chart-type-agnostic: bar charts, scatter plots, line charts, and layered charts all use the same vocabulary, with different combinations producing different navigation topologies. This directly addressed Bokeh's problem: the API mirrors data fields and encoding choices as a set of dimensions.

6.5 *Skeleton*: System Design

With a visual structure renderer (the Inspector) and a declarative abstraction for producing navigation structures (the Dimensions API), the remaining problem was to bring these capabilities into an integrated, direct-manipulation authoring environment where practitioners could not only see structure but manipulate it interactively, test it with real input, and iterate on it with immediate feedback. *Skeleton* is that environment. It integrates the Inspector and Dimensions API into a unified workflow and extends both with a guided preparation phase, a spatial rendering canvas, and a live testing mode. Each of Skeleton's authoring techniques makes visible a specific property of non-visual interaction that sighted practitioners otherwise cannot see during authoring: the topology of what is navigable, the spatial mapping of where navigation lives over the chart, the semantics of what is announced, and the temporal sequence in which a user encounters nodes.

Where the Inspector requires a practitioner to write code first and visualize after, *Skeleton* reverses the direction: practitioners design a navigation structure visually and inspect the code representation as a consequence of that design.

6.5.1 Staging: Input and Preparation for Authoring

The authoring workflow proceeds through four stages: upload, prepare, edit, and test. Making a visualization accessible involves decisions about what is navigable, how navigation is triggered, and where focus indicators appear in space. These decisions are related but distinct, and collapsing them into a single undifferentiated interface, as code-only workflows effectively do, makes

each one harder to reason about. The stage architecture surfaces them as separate concerns. It also preserves a direct correspondence between what practitioners see in the interface and the structure of the Data Navigator API [35]: each stage maps onto a distinct layer of the API, lowering the barrier to moving from visual authoring into code when production deployment requires it.

Upload. The upload phase is deliberately permissive. Practitioners can bring a dataset, a chart image, both, or neither. When a dataset is present, *Skeleton* parses its fields and infers dimension types automatically, producing a default, starting configuration for the *Prepare* step. When only an image is present, practitioners proceed directly to editing and construct nodes and edges manually over the image. This was motivated by our geologic map co-design work: many bespoke visualizations do not have a single underlying dataset, and any tool that requires structured data as a precondition for excludes the cases that need it most. *Skeleton* can be applied to any 2D image surface, not only to visualizations in the conventional sense.

Prepare. The prepare stage addresses a hard authoring bottleneck: not placing nodes, but deciding what structure to build at all. A practitioner who has never designed a navigation structure faces an open configuration space with no obvious entry point. The prep stage presents a four-chapter Q&A wizard that moves through authoring decisions sequentially: (1) whether the chart should have a root node and what it should announce, (2) which data fields should be navigable dimensions and how those dimensions should behave at their boundaries, (3) which keyboard interactions each dimension should be assigned to, and (4) what text labels each level of the hierarchy should produce. Each chapter is accompanied by an illustrative schematic diagram of which part of the heirarchy is being edited as well as a diagram showing examples of what these decisions look like when showing on a chart. The wizard's output populates a configuration in the editor that practitioners can then inspect, refine, and revise.

6.5.2 Edit: Interacting with Topology, Layout, and Semantics

Seeing the system, seeing the experience. The editor is *Skeleton*'s primary authoring environment (**??**) and presents two interlinked representations of the same navigation structure. A schema panel shows the structure as an abstract hierarchical tree layout that makes levels and parent-child relationships immediately readable. A graph canvas shows the same structure rendered as geometric elements positioned over the uploaded chart image, representing what an end user would encounter spatially. These two representations are bidirectionally linked: selecting a node in either view propagates the selection to the other, so practitioners can simultaneously hold in mind both the abstract topology of what is navigable and the spatial instantiation of where that navigation will live. The dual-view design makes visible a real conceptual divide between the system's model of navigation and a user's experience of it, one that practitioners recognize once they can see it, even without prior vocabulary for it.

Leveraging visualization as a scaffolding engine. Manually positioning leaf nodes over each data mark is the most mechanical step in the authoring workflow, and it scales poorly with dataset size. In our early pilot sessions, actually placing nodes in the canvas space was the slowest and most tedious part of the process. To address this, *Skeleton* includes a scaffold tool (Figure 6.3) that automates spatial placement by repurposing Vega [136] as a layout engine.

The scaffold renders a Vega chart specification to a hidden, off-screen container and extracts

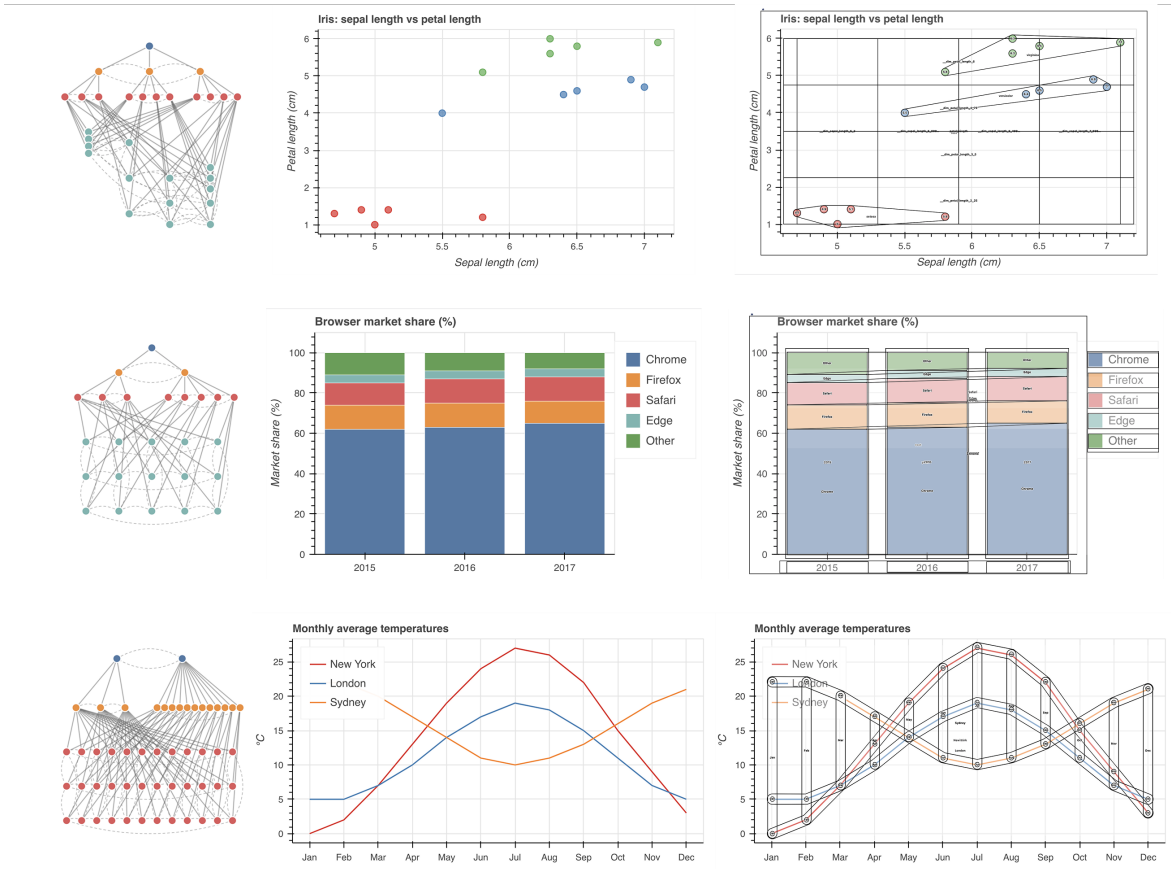

Figure 6.3: Input data transformed into a navigable structure using the *Dimensions API* and visualized with our *Inspector* gadget (left). The input chart (middle). The navigable structure is transformed and drawn over the chart using the *Scaffolding Engine* (right).

node positions via the library's internal scale and view APIs, using the rendering engine purely as a coordinate computation step: no actual chart is ever shown to the practitioner. Coincidentally, none of our co-designers were crafting visualizations using Vega or Vega-Lite (or derivatives), yet the Vega rendering engine could faithfully reconstruct every necessary mark position over the underlying, inaccessible data visualization provided to *Skeleton*.

Additionally, positions and outline strategies for category-level group nodes are computed from the leaf positions using geometric algorithms: a union of child node paths, a convex hull, a grid over numerical bounds, or a bounding rectangle. These designs were consistently produced as ideal treatments during co-design, so we chose them for our starting set of outline strategies.

The scaffold is optional and works by generating synthetic placeholder positions that practitioners can adjust manually. It dramatically improved authoring speed: in light pilot tests, a research team member completed a three-dimension structure without scaffolding in 8 minutes 22 seconds and with scaffolding in 56 seconds. A co-designer completed the task incorrectly (failing to account for one dimension's division nodes) in 13 minutes 7 seconds without scaffolding, and correctly in 2 minutes 44 seconds with it.

Specifying token patterns, editing instances. Selecting any node populates a properties

panel with spatial properties (position, size, shape) and semantic properties (ARIA role, description, and a label template editor; Figure 6.4). The label template allows practitioners to assemble the text a screen reader will announce at that node from tokens drawn from data fields, including aggregate statistics at group-level nodes and precise data values at leaf nodes [84]. Editing labels can apply to all nodes of the same type or to a single instance.

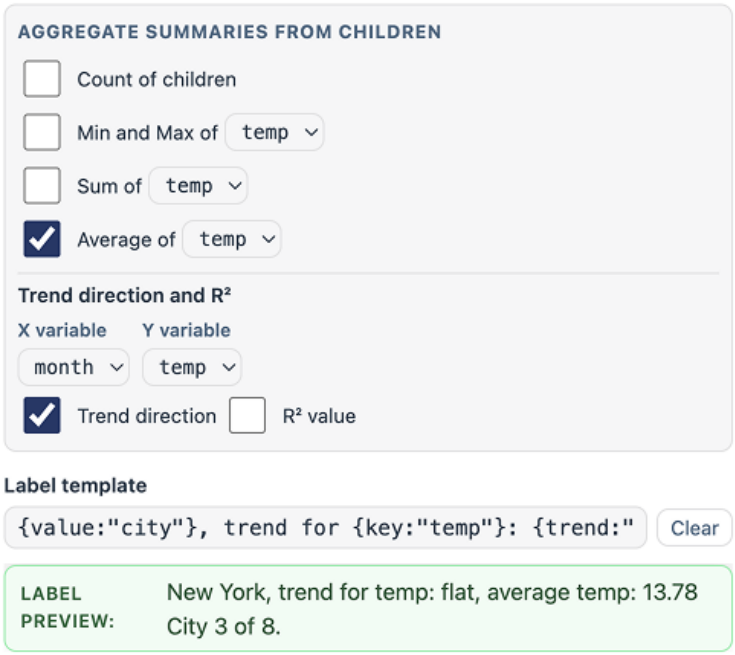

Figure 6.4: Group label pattern builder, including an array of aggregate summary options, template formatter field, and preview.

At the bottom of the semantic section, a live preview displays the full assembled announcement string in the exact form a screen reader would produce: role, semantics, group membership, and label combined into a single rendered output that updates in real time. Prior to this preview, understanding what would be announced at a given node required running a screen reader and navigating to it sequentially. In deeply nested structures, this meant spending several seconds listening to labels and drilling in. This was consistently the main point where bugs were produced and missed during our co-design work.

The preview makes text announcements inspectable as visible, editable objects. This surfaces a class of highly specific, low-level problems that code-only workflows leave invisible: redundancies in announced text, missing contextual framing, and label ordering and punctuation that affects comprehension and reading speed. Of all the authoring decisions *Skeleton* exposes, label templates involve the most degrees of freedom and have the most direct bearing on the quality of the non-visual experience. Getting the structure right ensures navigability; getting the labels right determines whether navigation communicates anything meaningful.

6.5.3 Test: Debugging Interaction Interactively

The testing stage allows practitioners to navigate the structure they have built using the same keyboard input and navigation rules that assistive technologies would use, without leaving the tool. When a practitioner enters testing, the three Data Navigator modules are instantiated in sequence: the structure module rebuilds the navigation graph from the current configuration, the rendering module creates an HTML layer positioned over the chart image at each node's spatial coordinates, and the input module registers keyboard listeners for all navigation rules. The result is a live, keyboard-navigable structure. An event log records navigation events in order, letting practitioners verify that all nodes are reachable and that label sequences make sense when encountered serially rather than read simultaneously.

The abstract graph continues to display during testing. As the practitioner navigates, the focused node is highlighted simultaneously in the canvas (showing its spatial position over the chart image) and in the abstract graph (showing its structural position in the hierarchy). This parallel tracking makes the temporal traversal sequence visible, showing the order and path through which a user encounters nodes, so practitioners can verify at a glance both where focus is and what role it occupies. A text-chat mode is also available, in which practitioners navigate by typing natural language commands, motivated as a design by the Adobe collaboration (Section 6.4.2), to explore interaction alternatives for mobile screen reader users.

The testing stage also served, during development, as the primary debugging interface for *Skeleton*'s own data pipeline. Errors in position computation, label resolution, or structure generation that would propagate silently through code became visible the moment a node highlight appeared in the wrong location or a label read as undefined. A tool for making non-visual structure visible turned out to benefit from the same property during its own construction.

Practice-based validation. CD2, a subject matter expert who professionally evaluates interfaces for screen reader access and is familiar with other visualization navigation systems, used *Skeleton*'s testing stage to evaluate navigation output across several chart types and dimension configurations: line charts (3 configurations), bar charts (2), stacked bar charts (3), and scatterplots (4). This evaluation followed a manual, systematic approach combining standards-based criteria with expert screen reader testing. Scatterplots required the most iteration, surfacing bugs in Data Navigator's core library that were then fixed. CD2 also recommended that we rely on list-based navigation while in the editor (before the testing stage) in case users build themselves into a keyboard trap.

6.6 User Study

Our co-design work followed an action research orientation: the research team was embedded in practitioner communities, and our system work was motivated by the problems those communities faced. This process generated the techniques and infrastructure that comprise *Skeleton*, but we still needed to understand the impact our system had on visualization practitioners more broadly. Our co-designers were deeply familiar with the problem space, having spent months or years working on accessible navigation. We needed to understand what happens when practitioners who are *not* embedded in this process design, author, and debug navigation structure

visually: whether the representations we built are legible to them, whether the techniques change how they reason about accessible design, and what new questions or problems emerge when navigation structure becomes visible. These are empirical questions that required a study.

To evaluate how *Skeleton* influences the way practitioners engage with accessible design, we conducted an in-situ interview study with 8 participants across visualization design, engineering, and research. The study was conducted remotely over video call, took approximately 45–60 minutes per session, and was approved by our IRB. Participants provided verbal consent at the start of each session. Video and audio were not recorded, however some participants consented to share the data/image they brought to the study as well as screenshots of their workflow; data collection was note-based throughout.

6.6.1 Participants

Participants were recruited through snowball sampling within the visualization and accessibility community, and through referrals from co-designers involved in our earlier collaborations. We asked each participant to self-report their primary work role (engineering, research, design, or student) and their existing level of accessibility expertise on a 1–5 Likert scale. We also asked whether the visualizations they build are ever bespoke, that is, custom rather than instances of a recognizable, standard chart type. This distinction mattered because bespoke visualizations represent an especially underserved case in accessible design tooling: no library pattern applies, and every navigation structure must be designed from scratch. Participants were not compensated.

6.6.2 Procedure

Each 45-minute session proceeded in four phases. Before the session, all participants were asked to prepare a chart image they were currently working on or had recently built, for use in the third phase.

Phase 1: Introduction and demographics (5 minutes). After obtaining verbal consent and recording a pseudonym, we collected self-reported role, accessibility expertise level, and whether the participant regularly builds bespoke visualizations.

Phase 2: Generic chart think-aloud [2] (10 minutes). Participants used *Skeleton* on a provided bar chart and dataset of fruit counts (Apples, Pears, Nectarines, Plums, Grapes), asked to design an accessible navigation experience for a screen reader user with no instructions on how the tool worked. The tool loaded with a default structure having both a categorical dimension (`fruit`) and a numerical dimension (`count`) active. This default was intentionally problematic for two reasons: numerical navigation sorts by count value and groups data into subdivided ranges, producing a traversal order different than the visual layout an additional, largely unhelpful level in the hierarchy for such a simple chart. We observed how participants reasoned about what they saw and whether and how they noticed this extra dimension.

Phase 3: Own-chart think-aloud (15 minutes). Participants loaded their own chart image and attempted the same task, except they were also asked to explain their graphic to the research team (purpose, role, data, and domain). This phase was open-ended: charts ranged from standard types to bespoke visualizations, and the goal was to observe how participants reasoned about navigation structure when the context was their own work.

Phase 4: Reflective interview (15 minutes). We conducted a semi-structured interview in which participants reflected on their decisions in Phase 2 versus Phase 3, their experience with the generic versus their own chart, and their assessment of the tool's capabilities and limitations. We asked what felt possible or impossible, what they wanted to do that they could not, and what they found themselves thinking about that they had not considered in Phase 2.

6.6.3 Analysis

Notes from each session were compiled into a shared document. Participant quotes reported in the results are reconstructed from these researcher notes, not verbatim transcripts. We analyzed the data using a combination of thematic analysis [13] and affinity diagramming [59], iterating across both methods to surface recurring patterns while preserving the specificity of individual participant experiences. Analysis attended particularly to differences in how participants engaged with accessible design before and after using the tool, the range of input modalities and user scenarios they considered, and moments when participants reconsidered or wished to redesign their own visualizations.

6.7 Results

We organize our findings into five themes that emerged from thematic analysis and affinity diagramming across all eight sessions. Each theme captures a qualitative pattern in how practitioners engaged with accessible navigation design when its structure was made visible and manipulable. We report these findings descriptively and ground them in specific participant moments; interpretation follows in the Discussion.

6.7.1 Seeing Navigation Made Structural Problems Legible as Design Problems

The generic bar chart in Phase 2 loaded with an intentionally problematic default: both a categorical dimension (`fruit`) and a numerical dimension (`count`) were active, producing overlapping navigation structures with different traversal orders over the same data. This configuration is a poor design choice, arguably a design failure, but one that would be difficult to detect in code alone (**R1**, **R4**).

Participants varied widely in how quickly they recognized the problem. P1 turned off the numerical dimension within seconds of seeing the editor, without commenting on it. Most participants, however, initially struggled to understand what they were seeing, remarking on the unfamiliar structure: "what is this? what are these?" when encountering the numerical divisions for the first time. P8 spent time trying to guess what the extra divisions represented but did not remove them during Phase 2, only realizing during Phase 3 that the additional dimension was "probably bad." P2 and P5 expressed suspicion early: P5 asked, "Is this too much data? This seems like way too much to just navigate through," and P2 noted, "I feel like a lot of data points would be bad, yeah? Like, too many at once is bad?" Both of these remarks were prompted by the visual density of the structure, not by navigating it.

The testing stage (Section 6.5.3) proved critical for resolution. P4, P5, and P7 each removed the extra dimension only after navigating the structure with keyboard input in testing mode, where the traversal sequence made the redundancy experientially apparent. In total, five of eight participants resolved the problem during the session: P1 and P2 during editing, and P4, P5, and P7 after testing.

Beyond the intentional default, participants identified other problems through visual inspection. P8 reacted to a generated node name: "Okay dim_fruit node…that is horrible, what is that?" During Phase 3, P7 looked at the edges of their multi-line chart and asked, "are all these bad? Is it bad that I don't even really know what the takeaway of this [structure] is?" P3, seeing a full hierarchy for their own simple six-item bar chart (during Phase 3), concluded "I should just skip the root and grouping and go straight to the data. This seems like too many steps." In each case, the visual representation of their navigation structure motivated judgment about potential negative design qualities.

6.7.2 Practitioners Developed a Designerly Interest in What Constitutes Good Navigation

The most pervasive pattern across sessions was that participants began asking design questions about navigation quality, unprompted by any instruction or guidance from the research team (**R2**). These questions went beyond identifying errors: participants wanted to know what *good* navigation would be for their charts.

P2, working through the bar chart in Phase 2, deliberated over boundary behavior: "Loop back or stop? I don't think there is a right way. I will just pick *fruit* for now and *loop* and see what this does." P6, who brought a bespoke flower visualization, wondered how to translate the affective quality of their chart: "I think my visualization should be more about the vibes, but I don't know how to make the alt text have good vibes. What is *fun*?" P4 asked fundamental questions about the interaction model itself: "Why do screen readers and keyboards have to work this way? Do people like that?…why do we navigate?" And later after testing, P4 concluded, "I bet we should make this *faster*" before cutting out the additional numerical dimension in Phase 2.

Several participants engaged with the concept of narrative and flow. P8 articulated this as a question about the goal of the visualization: "sometimes I want a big picture, not precision. I may want to drill down a little…" and observed that "we think too much in terms of components…sometimes accuracy isn't the actual goal, it's getting a general sense of something." P4, upon discovering that *Skeleton* supports text-based input, asked, "How do you make that good, though? Like chatGPT, or do people want to, like, interact with the chart [elements]?'

During the interview, several participants explicitly requested guidance. P1, P2, P3, and P6 wanted to see examples of well-designed navigation experiences. P1, P3, P4, P5, P6, and P7 wanted embedded guidelines within the tool. P2 and P4 actually used web search to look for "chart navigation for accessibility guidelines" (P2) and "accessible viz screen reader design" (P4). P3 was interested in automation and heuristics that could suggest reasonable defaults. These requests are consistent with the pattern (**R2**): practitioners who could see the design space wanted orientation within it.

6.7.3 Iteration Was Substantive, Self-directed, and Concentrated on Semantics

Every participant iterated on their navigation designs, and this iteration was neither perfunctory nor prompted by the research team (**R4**). Participants revisited decisions, revised configurations, and in some cases restructured their entire approach after encountering their design in a new stage of the tool.

The most sustained iteration concerned labels and text announcements. Every participant spent the largest share of their authoring time in the label template editor (Figure 6.4), editing the text that a screen reader would announce at each node. This editing was granular: participants wrote full sentences, rearranged the order of data tokens, debated whether to include field keys alongside values or values alone, experimented with how to name groups and individual elements, and considered the length and density of the resulting announcements. At the division and dimension levels, some participants added multiple aggregate statistics (count, sum, average, range, trend) and then returned to trim them. P2, for instance, edited data point labels, left them, and returned to revise them two additional times: "This label is way too complicated, I think." These label iterations were small, fast, and frequent, and they reflected an intuitive grasp of the importance of the textual tokens that constitute a screen reader user's primary interface to data.

A second, distinct pattern of iteration emerged around testing. The editor displays all navigation nodes simultaneously, showing the full structure at a glance. The testing stage (Section 6.5.3), by contrast, shows only the currently focused node, highlighting it in the canvas one at a time as the practitioner navigates. This difference consistently prompted participants to revise. Several returned to the editor after testing to adjust group node outlines, because outline strategies that looked distinct when displayed simultaneously (such as bounding rectangles vs. convex hulls) became harder to differentiate when encountered one at a time. Others adjusted their dimension configurations: P3 and P5 restructured their dimensions after testing, and P4, P6, and P8 experimented with different key bindings and navigation rules. P2 used the testing stage specifically to identify labels that needed revision at the dimension and division levels. In these cases, testing was not only treated as a bug-finding activity but also as a way to encounter, reason about, and then improve a design that had looked adequate in the editor but felt inadequate in sequential traversal.

The most extensive iteration came from P1, who returned to the preparation wizard after reaching the testing stage and re-took the entire wizard from scratch. Seeing the full structure in testing motivated them to re-evaluate their earliest decisions. They reduced their navigation from three dimensions to one, producing a navigation experience they described as deliberately minimal. In the reflective interview, P1 explained that they had been thinking about trimming excess, joking that they were "working on a data-to-word ratio."

The scaffolding tool shaped the pace and character of these iterations. Participants who used the scaffold (Figure 6.3) generally required only minor manual adjustment of node positions, spending one to two minutes refining placements in-aggregate before moving on. Iteration on spatial layout was primarily about the appearance of group-level outlines rather than individual node positions: because the scaffold computed leaf positions from the chart's encoding, the main remaining spatial decision was how division and dimension nodes should visually indicate their grouping. When participants returned to scaffolding, it was most often because they wanted to

change the chart type used for coordinate computation or to try a different group outline strategy.

6.7.4 Seeing Navigation Prompted Practitioners to Reconsider the Architecture of Their Own Charts

An unexpected finding was that working with *Skeleton* prompted several participants to question the design of the visualization they had brought, not just the navigation structure they were building over it (**R4**). Five participants (P4, P5, P6, P7, P8) expressed a desire to simplify their own charts after seeing what the corresponding navigation structure required. Three (P1, P5, P6) considered whether a different chart type might serve the same communicative goal with a simpler navigational architecture. And in 2 cases (P1, P6), participants questioned whether a chart was the right medium at all.

P5, who brought a scatter plot with over two thousand data points, initially tried to build a navigation structure over the full dataset, recognized that the result was unmanageable, and then asked: "do I even need visualization?" They went on to wonder whether the information could be communicated as "a few sentences or like, some data [users] can prompt" (referring to using a large-language model). P6, who brought a bespoke flower visualization in which each petal encoded a different variable, initially tried to express the chart's full complexity in navigation (grouping by flower and then navigating by petal) but then reversed course: "okay, what if we actually just treat this like a bar chart?" Using the scaffold tool, they produced a simple list-style navigation that worked well for their data, and reflected: "my visualization has a lot, but actually, this could be pretty easy to navigate, I think." P6 also wondered whether there was a non-chart way to "tell their story," and in further discussion described something closer to a scrollytelling article or guided walkthrough than a single interactive chart.

P8's case was the most striking. They brought a voronoi pie chart (Figure 6.5) used to communicate to students that 26 assignments make up 45% of their grade, the largest slice of the pie. Working through *Skeleton*, P8 first considered making all 26 voronoi cells navigable, then considered making only the three main slices of the pie navigable, and then questioned whether navigation was needed at all. The "point of this," as they described it, was to communicate a single ratio; students did not need to traverse individual cells to understand it. P8 concluded that well-written alternative text was sufficient for the screen reader experience and chose not to build a navigation structure. They also reflected on the design of the visualization itself, concluding that the voronoi treatment served a visual purpose (it is visually striking) that was separable from the informational purpose (communicating a grade breakdown). They kept the visual design and simplified the non-visual experience accordingly.

6.7.5 Experiencing Keyboard Navigation Surfaced a Broader Range of Users and Input Technologies

The testing stage (Section 6.5.3) provided what was, for most participants, their first experience of keyboard navigation through a data visualization. Only three participants (P1, P2, P8) reported prior experience using a screen reader, and only two (P1, P8) had previously navigated a visualization using keyboard input alone. Yet during the study, every participant navigated using

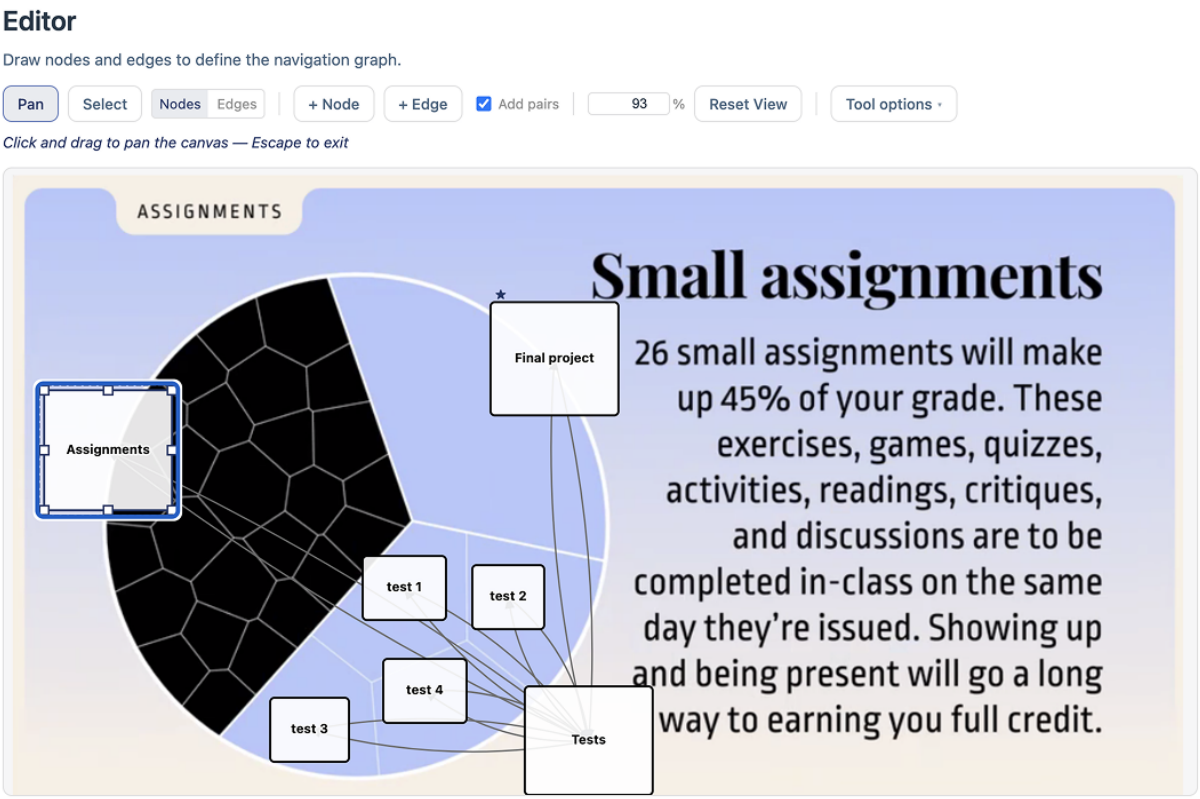

Figure 6.5: Re-creation of P8's moment of realization, placing nodes manually: not every element in their voronoi pie chart *should* be navigable.

a keyboard.

Several participants responded to this experience with genuine engagement. P4 reacted with enthusiasm: "Woah, this is incredible. Wait, what other visualizations can do this?" and later, "Maybe you could make a visualization game, where you can move around the data?" P7 said, "I love this. I want to make charts with this." P5, who had not previously encountered keyboard-navigable charts, said, "I didn't know you could do this." These reactions were not about the tool; they were about the interaction modality itself, about the experience of traversing data spatially using directional input.

The experience also expanded participants' sense of who these structures serve (**R2**, **R4**). P8 adjusted a node's spatial dimensions and said, "Let's make the hitbox as big as possible here…for my neighbor with Parkinson's to click these," treating the navigation structure as relevant to motor accessibility, not only screen reader access. P3, observing the visual focus indicators that appeared during scaffold-based placement, remarked that "this is for more people than [someone] blind," recognizing that sighted keyboard users and users with partial vision also rely on visible focus states. During the interview, P4 asked sustained questions about what kinds of technologies different people with disabilities use, and P4 and P7 both asked why focus indication was designed to be visible rather than invisible.

P4's curiosity extended to alternative input modalities. Upon learning that *Skeleton* supports text-based navigation and (due to leveraging Data Navigator) also supports a wide array of other

input modalities, they asked, "Can you talk at it, too? Is that what some people do?" and followed up with, "How cool is that? How do you make that good, though?" P8 raised a concern about discoverability in the current interaction model: "I've never used j or w to drill out, always shift arrow or option arrow…" and worried that a user might not know how to exit a nested level. These observations reflect an broad mental model of the user, from an abstract "screen reader user" to a person with multiple capabilities, preferences, and interaction patterns (**R2**).

6.8 Discussion

6.8.1 Visibility as a Precondition for Iteration

The central finding of this work is not that *Skeleton* taught sighted practitioners how to design accessible navigation, but that it gave them something to react to. When navigation structure was invisible, our co-designers would only seek to verify whether navigation followed our design documentation. Once visible, questions shifted to whether it was good, why, how it could be improved, and what else it could do. Visibility encouraged iteration, and iteration is enabled continuous design improvements.

This mechanism is consistent with Schön's account of reflective practice [140]: designers iterate by externalizing a representation, perceiving its properties, and responding to what they see. The representation talks back, and the designer adjusts. But this loop requires a representation. In our co-design work, we assumed that the gap was hand-off between design and development, and the slow process of manual verification. Instead, the gap is that development was not participating in, and encouraging, further design reflection. In a developer's code-only workflow, navigation structure has no externalization that supports this kind of perceptual engagement. A practitioner can read the code that specifies a navigation graph, but they cannot see the graph, cannot perceive its topology at a glance, cannot notice that a label is redundant or that a hierarchy is too deep by looking at it. The reflective loop is disrupted and occluded at the first step.

Skeleton encourages this loop by rendering navigation structure in a form that supports the same kind of perceptual judgment that visualization practitioners already apply to every other aspect of their work. The results show what happened when this loop was available: every participant iterated, substantively and self-directedly (Section 6.7.3). They revised labels repeatedly, restructured dimensions after testing, adjusted group outlines when sequential traversal revealed problems that simultaneous display had hidden. P1 restarted the entire preparation wizard after seeing their structure in testing mode. These are not the behaviors of practitioners following a specification; they are the behaviors of practitioners negotiating with a design material.

The co-design work reported in Section 6.4 provides complementary evidence: in each collaboration, sighted practitioners naturally reached for visual representations when reasoning about navigation, through node-link diagrams in Figma, schema sketches in Miro, and annotated wireframes on paper. They were already thinking visually about non-visual structure; the development tooling simply had not caught up. This has a practical implication for the broader field: if sighted authors depend on visibility, then any authoring workflow that keeps navigation structure invisible limits design iteration. Making structure visible does not guarantee good design, but it is a precondition for the kind of sustained, judgment-driven refinement that good

design requires.

6.8.2 From Compliance to Design

A consistent pattern across the accessibility literature is that practitioners frame accessibility as a compliance problem: a set of requirements to satisfy, a checklist to complete, a legal or institutional obligation to meet [34, 85, 146]. The framing matters because compliance and design orient practitioners toward fundamentally different activities. Compliance asks: "does this pass?" Design asks: "is this good?" Our results suggest that *Skeleton* produced a partial shift from the first orientation to the second. Participants' initial instincts clustered around compliance-oriented responses: provide alternative text, follow guidelines, ask an expert. But alongside that desire for guidance, they began doing something compliance framing does not typically produce: they encountered complexity and then *iterated*. They revised labels repeatedly, restructured dimensions after testing, debated boundary behavior and hierarchical depth. This sustained, self-directed refinement is the behavioral signature of design, not compliance. As P4 put it: "I'd love to have someone blind actually just with me while I make this, but I also understand that I should learn what makes a good experience too."

The shift extended beyond the non-visual experience itself. As reported in Section 6.7.4, five participants reconsidered the design of the visualization they had brought, not just the navigation structure overlaid on it. Making the accessibility consequences of visual design choices visible prompted practitioners to question whether a different chart type, a simpler encoding, or a non-chart medium might better serve their communicative goals. This interrelation between non-visual and visual design suggests that the widespread treatment of accessibility as a compliance activity may be partly a consequence of tooling that offers no legible, manipulable design surface. Auditing frameworks are valuable, but they are *evaluative* tools, not *authoring* tools. The field needs both.

6.8.3 Bespoke Visualizations as an Unaddressed Accessibility Research Problem

Diagrams, infographics, and data-driven illustrations are often one-off, custom designs with bespoke symbols, layouts, and visual languages. These representations are increasingly common in journalism, scientific communication, personal projects, art, and public-facing data work, and they represent the cases where accessibility-focused tools are needed most and available least. *Skeleton*'s image-based workflow (upload any 2D image, place nodes manually) provides a starting point, but the study made clear that bespoke visualizations need more than node placement. They need support for reasoning about what navigational structure (if any) is appropriate when no template applies, a research problem that remains largely unexplored.

6.8.4 What Visualization Owes Accessibility

Our approach, to make visual non-visual experiences, should not be limited to data visualization's own accessibility challenges. Navigation structure is a foundational component of acces-

sible experience across domains: PDF and document reading order, web page structures, and software application layouts. In each of these areas, sighted practitioners author non-visual experiences without visual feedback, and in each, the same gap between design intent and verifiable outcome constrains quality. Testing is slow, error prone, and requires expertise in assistive technology use. Visual tooling for authoring, inspecting, and debugging non-visual structure (**R3**) is a tractable and high-value problem across application domains.

There is also a deeper question about what visibility can accomplish in principle. Work on multi-modal authoring environments has argued for de-centering visual representation and treating modalities as equal partners in the design process [196], an important ethical commitment. *Skeleton* does not do this: it re-centers visual representation as the medium through which sighted practitioners engage with non-visual structure. We believe this is justified pragmatically, because sighted authors need to articulate navigation design in their own perceptual language before they can reason about it at all, and this paper provides evidence that they do. But articulating a design in one's own language is not the same as understanding how it will be experienced in someone else's. The structures participants built during our study were never evaluated by blind users, and visibility alone cannot substitute for that evaluation. The risk we want to name is that making non-visual structure visible to sighted practitioners could be mistaken for making it *understood*, when in fact it makes it *designable*, a real but bounded gain. The fuller design process requires collaboration with disabled users, not as an occasional supplement but as a regular practice. *Skeleton* can make that collaboration more productive by giving both parties a shared representation or space of translation between representations, but it cannot replace it.

What visualization owes accessibility, then, is not simply better output but authoring tools that better stimulate reasoning, both individually and collaborative, about design.

6.9 Limitations and Future Work

Skeleton makes navigation structure visible to sighted authors, but it cannot reassure those authors whether the structure they have built is good for the people who will use it. The tool surfaces design questions; it does not answer them. Several participants asked what constitutes good navigation, and *Skeleton* had nothing authoritative to offer. Our study with sighted practitioners evaluated whether making navigation structure visible stimulated design consideration, not whether the designs sighted practitioners produced were actually good. These are related but distinct questions, and the second remains open. CD2's expert screen reader evaluation of *Skeleton*'s navigation output (Section 6.5.3) provided practice-based validation for several common chart types and surfaced concrete bugs, but this evaluation was neither comprehensive nor controlled: many configurations remain untested, and expert review is not a substitute for evaluation with a broader population of end users.

Additionally, we treated *Skeleton* as a design probe [56, 81] rather than comparing it to a controlled condition: the goal was to elicit qualitative insight about how the tool elicits engagement, not to measure performance differences.

Our approach using *Skeleton* as a design probe only with sighted participants is both a limitation and, we believe, the right sequencing: *Skeleton* improves the intentionality and iterability of what sighted practitioners produce, which is a necessary precondition for a subsequent evaluation

or future collaborative design work between sighted and blind authors. Further research and evaluation should close this loop, ideally to engage how mixed-ability teams co-design multi-modal data experiences.

Skeleton is also a prototype with substantial work remaining. Not within the scope of the paper, but our participants provided ample feedback on the functionality of the prototype itself. The most urgent gap is export functionality: practitioners can design and inspect navigation structures in the tool but cannot yet produce deployable output.

6.10 Conclusion

Accessible navigation structure has long occupied an awkward position in visualization practice: known to matter, difficult to design, and invisible to the people responsible for building it. The invisibility was not incidental. Without a way to see what they were making, sighted practitioners could not catch errors, could not iterate, and could not develop the kind of considered judgment that good design requires. Accessibility remained downstream of every other decision not because of any single failure, but because the authoring conditions did not support anything else.

Skeleton demonstrates that those conditions can be changed. Making navigation structure visible and manipulable, as an interactive graph rendered over the spatial layout of a real visualization with live label previews and testable traversal, shifted how practitioners engaged with and reasoned about accessible design. They began asking qualitatively different questions than someone seeking compliance: whether the features and design of their structure was good, despite not having readily available answers.

If any conclusive take-aways can be gleaned from this project: for researchers interested in engaging accessibility, this would mean future projects might explore translational spaces between visuals and non-visuals that help sighted partners engage blind designers. For practitioners who build, design, or audit this would mean that more visualization is needed in current tooling, to enhance and make multi-modal existing non-visual methods of authoring and evaluation.

What the paper leaves open is more than what it closes. We used *Skeleton* as a design probe with sighted practitioners; we did not evaluate the navigation structures they produced with the end users those structures are meant to serve. And the broader disciplinary conversation, about what visualization research owes accessibility and what methods might transfer between them, has more questions than answers. We offer *Skeleton* not as a solution to these problems but as evidence that engaging them directly, with the full weight of visualization's methodological tradition, is both possible and fruitful.

Part V

Personalization: System-building for User Agency

Chapter 8
Softerware: Enabling Personalization of Interactive Data Representations for Users with Disabilities

---

This chapter was adapted from my published paper:

F. Elavsky, M. Vindedal, T. Gies, P. Carrington, D. Moritz, and Ø. Moseng, 'Towards *softer*ware: Enabling personalization of interactive data representations for users with disabilities', *Computer Graphics and Applications*, 2025 (to appear at *IEEE VIS 2026*).

8.1 Abstract

Accessible design for some may still produce barriers for others. This tension, called access friction, creates challenges for both designers and end-users with disabilities. To address this, we present the concept of softerware, a system design approach that provides end users with agency to meaningfully customize and adapt interfaces to their needs. To apply softerware to visualization, we assembled 195 data visualization customization options centered on the barriers we expect users with disabilities will experience. We built a prototype that applies a subset of these options and interviewed practitioners for feedback. Lastly, we conducted a design probe study with blind and low vision accessibility professionals to learn more about their challenges and visions for softerware. We observed access frictions between our participant's designs and they expressed that for softerware's success, current and future systems must be designed with accessible defaults, interoperability, persistence, and respect for a user's perceived effort-to-outcome ratio.

8.2 Overview

There is a significant and relatively unacknowledged problem in emerging work on accessible visualization: a single design cannot satisfy all users. People with disabilities, even those who share the same category of disability, often have different experiences, capabilities, and needs. As experienced practitioners and researchers who have been working to make data visualizations more accessible (some of us for more than a decade), we have each observed this persistent problem in our own practice.

Data visualizations that are produced by a designer for an audience tend to be designed in a way that is relatively *unchangeable*. As a material, we use the metaphor that the creator of a visualization manipulated their design while it was in a *softer* state, like clay. And eventually, the clay is *hardened* into a state that is presented to the user. Often visualization design artifacts cannot easily be altered by an end-user after they are created. Pixels cannot be moved, graphics

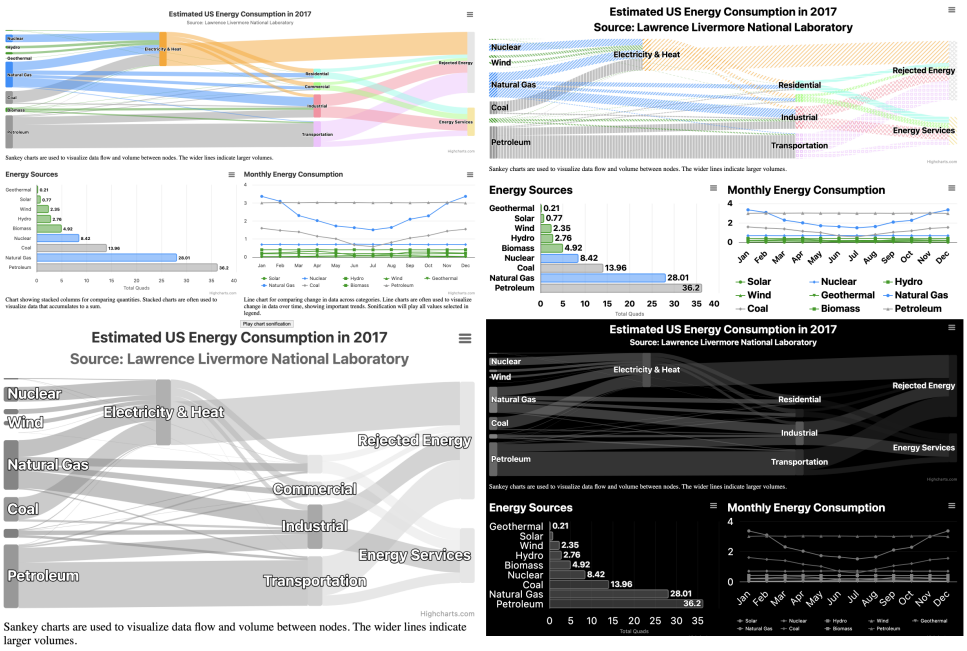

Figure 8.1: Sometimes one design is not enough. Our design (upper left) and three different designs by low vision users. All low vision users chose larger text, but then diverged: redundant-encoding enabled (upper right), high zoom and greyscale on white (bottom left), and then dark mode (enabled externally) with greyscale (bottom right).

cannot be re-embedded. The clay has been fired and the visualization is now *baked*. We intend to make visualizations easier to change and manipulate for end-users. We want a material that is softer than software. But we also don't want a fully malleable interface, like in end-user programming, either. We propose to call this space of design "softerware."

We wanted to explore material softness *with constraints*. We conjecture that advancements in malleable interfaces and end-user programming are too system-centric and open-ended. End-user programming puts too much burden on end-users to know the language and symbols of interface-building in order to build their own interfaces. Instead, we chose to enable end-users to have agency over a data visualization interface by exposing a preferences-driven menu of options built from our existing knowledge of visualization and accessibility.

Exploring the *softerware* gap in data visualization became our primary research focus, which led us to formulate the following qualitative exploratory research questions:

- **R1**: What constraints and capabilities should we provide end-users to give them meaningful agency over interactive data visualizations?
- **R2**: What qualities, challenges, and design opportunities do designers and engineers envision for a data visualization *softerware* system?
- **R3**: What qualities, challenges, and design opportunities do blind and low vision users envision for a data visualization *softerware* system?

We contribute our findings from this research to the larger accessibility and visualization communities in hopes that we can inform and inspire future work that investigates *softerware*

systems, end-user design, preferences-based user experiences, and fluid and malleable interfaces focused on end-users with disabilities. Our future work will focus on completing a finished version of our prototype and deploying it at scale within Highsoft's Highcharts ecosystem [70].

8.3 Related Work

Our contribution is an attempt to bridge the gap between the knowledge we have on accessibility for visualization (as a complex space of design and engineering) with research and practice that centers on users with disabilities being able to adust, change, or control the interfaces they interact with. We intend to frame our work towards the benefit of data visualization designers, system engineers, and end-users of data visualizations. We believe that more flexible data visualization systems that enable user preferences will require a careful approach to architecture and thorough consideration for the burdens placed on end-users.

8.3.1 Data Visualization and Accessibility

Data visualization accessibility has come far in recent years. But little work has been done to explore what disability scholars call "access friction" - a tension that arises when access must be negotiated [57, 77]. This friction is often a result of static barriers in shared spaces: one artifact or approach designed to include some people may end up excluding others.

In general, accessibility concerns itself with a broad spectrum of barriers that people with different disabilities face. And while most literature focuses on visual disabilities [110, 179], there are growing resources on areas such as cognitive/intellectual disabilities [186, 188], neurodivergence [164], and both research and systems exploring epilepsy and vestibular/motion inaccessibility in visualization [151, 153].

Yet despite these resources, making data visualizations more accessible remains a difficult task for practitioners [85, 146]. Some accessibility guidelines even conflict, for example on the topic of patterns and textures used in charts. One side stresses that patterns are harmful to cognitive and visual accessibility [138] while another stresses that redundant encoding strategies are necessary [34]. Understanding how to make the correct design decisions may sometimes be impossible. Either existing guidelines are incorrect or it is possible that access friction becomes inevitable the more we know what different barriers look like for different people with disabilities.

8.3.2 Systems that Adapt

One angle of exploration that has been engaging this issue already focuses on systems that can adapt. Work on adaptive systems for people with disabilities, such as in *ability-based design* [182], stresses the importance of design alleviating burdens placed on users. Users who don't fit initial system designs are often expected to adapt to fit the system. This means that they may have to acquire an assistive technology, learn a peripheral skill, hack the system, or wait on a design fix. This places the burden on the user to fit the system. Ability-based design in-

stead stresses that systems should be capable of automatically adapting, in order to reduce these burdens placed on the system's users.

However, building data visualizations that automatically adapt to users via some form of data collection often do so through means such as monitoring live biometric data and input patterns, collecting a user's self-declared conditions and cognitive ability, parsing a user's history, and sensing a user's environmental or situational context [189]. We argue that these methods for an adaptive system raise questions of end-user agency, trust, privacy, and awareness in regards to the system decision-making [122]. They may not be sufficient for addressing a user's needs while also preserving their privacy and agency.

8.3.3 Personalization and Accessibility

Lastly, we researched broader spaces where users have more design agency and explicit awareness of a system that is built to be adapted. We were interested in literature and projects that explore ways end-users can enact meaningful change on an interface, with special attention paid to accessibility and disability.

One specific project has emerged at the intersection of accessibility, visualization, and customization which focuses on screen reader users adjusting the content of textual tokens when navigating data visualizations [84]. While this is excellent work, we still have larger questions about when preferences, options, and customizations are appropriate and in what contexts as well as other ways of conceptualizing end-user agency over a system. It remains unclear when, why, and how customization and personalization can be used effectively when designing a system.

In the field of meta-design, meta-designers consider these end-user manipulations of a system to be one facet of "end-user design" and "continuous co-design" between a system and a user [104], which helps give us some meaningful language to refer to our system goals.

Recent work on the influential factors for personalization and adoption of accessibility settings [187] also informs our work in 2 key ways: conceptual mismatching between a system and user can contribute to a system's under-use while features that propose value, are time-saving, or reduce cognitive load for a user can contribute to positive perception and use of personalization of a system.

8.4 Presenting: *Softerware*

Softerware is a vision for software design that is not just based on giving a user the ability to set preferences or personalize. *Softerware* is about the intentional design of a software system that enables people with disabilities to have meaningful, opinionated, and persistent agency over that system.

We contribute the concept of *softerware* to the larger community of researchers and practitioners because we argue it is a useful construct that can help us categorize past work, improve existing projects, and inspire new directions. *Softerware* systems have been part of existing work for decades, but we lack a cohesive way to refer to designing and engineering experiences that enable end-users to have agency over malleable interfaces without entering into the territory of end-user programming and end-user development.

8.4.1 Defining *Softerware*'s Principles

Here we present the principles that define *softerware* before demonstrating an example instantiation in the context of online, interactive data visualization.

8.4.1.1 Principle: Has Reasoned, User-centered Constraints

An important aspect of *softerware* is that it is *softer* than software (which is already-baked) but not quite as *malleable*, free, and potentially low-level as systems that facilitate fully realized end-user programming and development [17, 95].

End-user programming is still a form of *programming* in the end. It focuses on taking constructs, functionalities, and reasoning from software programming and development and presenting these elements to users in ways that may suit a user's natural language or mental models, such as through no-code, visual-only, or low-code approaches. We anticipate that many users, especially those experiencing accessibility barriers, will have difficulty interacting with software paradigms based on end-user programming and development.

Instead *softerware* engages this limitation through reasoned constraints that leverage conceptualizations and language focused on overcoming anticipated user barriers. Providing constraints and then framing and presenting those constraints in ways that have vocabulary correspondence to user needs is what separates *softerware* from existing work and literature on end-user programming and development.

To accomplish this, the *softerware* system designer must work to anticipate not only what their system should do in a default or beginning state but also which ways that system will potentially fall short and require fitting by the end-user. The system designer should motivate all of the capabilities of a *softerware* system based on what they anticipate users will want to change, how users can discover that change is possible, and then how best to enable users to enact that change easily.

8.4.1.2 Principle: Facilitates End-user Agency

Softerware is ultimately about the process of architecting and implementing a system that enables an end-user to be able to easily express meaningful changes to that system's appearance and behavior.

Accessibility has been framed as a tension between fit and scale [69], where *fit* refers to a system that is perfectly complimentary and synchronized to a user and *scale* refers to a system that is capable of reproducing functionality for many different users. We believe that the tension between fit and scale, in addition to *access friction*, can both be alleviated when a system is designed to facilitate end-user agency.

The cornerstone goal of a *softerware* system is an attempt to facilitate *self-fitting* at a minimum, and in ideal circumstances also facilitate social methods of sharing fitting (such as loading profiles or ingesting metadata from others).

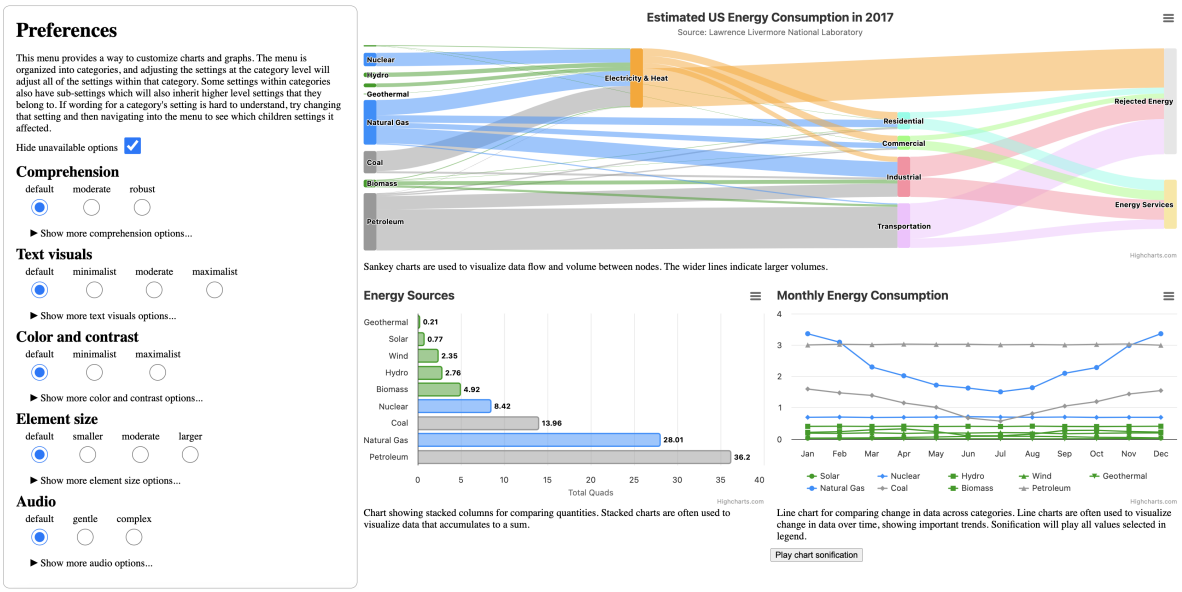

Figure 8.2: Our dashboard design on US Energy Consumption in 2017 with a sankey, bar chart, line chart, and preferences menu on the left.

8.4.1.3 Principle: Demonstrates Value

Existing literature makes one thing particularly clear when it comes to personalization and end-user design: it has to be worth it [187]. Users must be able to recognize barriers, issues, or shortcomings of a system and then discover and utilize capabilities provided to them to eliminate or alleviate those barriers.

This entire process must not be too burdensome and the payoff should establish an expectation that future use of the system will be improved. The time and effort it takes for a user to fix a problem should be less than the time and effort generated by that problem. This means that *softerware* systems can likely be optimized and improved significantly over time, as better techniques are developed to perform tasks as quickly and easily as possible.

The user should also be able to validate the value of their interaction with a *softerware* system through the continued use of that system. If something was a painful experience and they took action to alleviate that issue, they should be able to observe the effects easily.

8.5 Prototype: Visualization *Softerware*

We first built a data visualization dashboard (see **Figure 2**) that would allow us to build a *softerware* system prototype that we could demonstrate to designers, engineers, and evaluate with end-users with disabilities. You can view and interact with our prototype, including view our open source code and dataset on user preference options, on github.

We chose a relatively clean and standard dataset that would be relevant to our target end-users, who we would recruit from the United States, on 2017 US energy consumption. This dataset afforded us enough complexity to build multiple data visualizations in one dashboard (including

an uncommon type like the sankey). This choice also allowed us to explore more ideas for user interventions and investigate broader questions in our eventual study. Our dashboard was built in JavaScript and laid out in a wide format for interacting with on a desktop machine.

We designed the dashboard to contain some interactivity, but nothing highly complex. Each chart has tooltips and visual filtering provided on hover/keyboard focus and the line chart has data filtering through the legend as well as sonification.

8.5.1 *Pretty Accessible* by Default

In order to really test *access frictions*, we wanted our dashboard to be considered accessible in its default state. We ran manual and automated tests according to accessibility standards as well as a 50-heuristic manual accessibility evaluation specific to interactive data visualizations [34, 166]. In addition, we chose to use Highcharts to create the visualizations in our dashboard because existing work has demonstrated the breadth of their accessibility capabilities [93].

We ensured that text contrast was strong, text size was well above guidelines, interaction targets were of a minimum size, screen reader access was descriptive and interactive, our DOM was structured into a hierarchy, we provided semantic HTML data tables for each chart, and there were no *critical* issues detected when running automated tests using *axe DevTools* software. Many of the capabilities that made our dashboard accessible were provided out of the box by Highcharts.

8.5.2 Reasoned Constraints: 195 Accessibility Options for Interactive Data Representations

After building our initial dashboard (see Figure 2), our research team collaborated and discussed potential access frictions that might arise from the use of our dashboard. We used our existing experience from accessibility work, more than a decade in the case of the Highsoft and Elsevier co-authors, to assemble a list of concrete, expected user barriers that would be hard to resolve in a single design, and organized those barriers based on common themes.

We then used these themes to brainstorm how we anticipated end-users would identify barriers and then what language they might use to describe an identified barrier and overcome it. One example might be under the theme "hard to see": "I can't read this text" as a barrier with the final language for them overcoming that barrier as an expression similar to, "I wish this text size was larger" or "make this text bigger."

From these solutions, we re-framed the language into interactive option categories. All categories were given binned options and not more than 7 choices, to avoid overwhelming users. For example, "text size" was an option category and had 7 binned choices from "small" to "large" available. We then created a hierarchy of option categories underneath these higher-level categories which could allow for more element-level specificity within a data visualization. So "text size" as a higher level category with options also contained children that had more specificity, such as "title text size," "legend text size" and so on. The final stage of our language and options design was in line with prevailing industry wisdom, which was to avoid organizing the language of our configurations based on categories of disability [4] and instead focus on higher-level cate-

gories of identifiable elements of a user's experience, such as "text visuals," "audio," and "color and contrast."

At the end, we produced 195 option categories and 774 total option choices. Using the combinatorial rule of product, we calculated 6.83e114 possible unique end-user design configurations from these choices, which is more than the estimated number of atoms in the universe.

However, to ensure the scope of our user study was feasible, we reduced our initial working option categories down to 33 with 137 total options (and 9.35e19 possible design combinations), all focused on options we believed would be most relevant for users who are blind or low vision. These options and our subset are viewable in our live, interactive demo online.

8.5.3 Preferences Menu Design

We then iterated on visual and functional designs to allow users to actually interact with and enact these design configurations. Our early ideas included a natural language interface (since we used a relatively "natural language" centered process to develop these categories), direct manipulation of the elements in a visualization (through focus, hover, click, or selection methods), and eventually settled on the user interface of a separate, visually nearby menu with nested options (see Figure 2). We designed our menu so that manipulating higher level options in the hierarchy would enact downstream options to follow suit, but any manipulation to downstream options would override higher controls, following common patterns used in systems that implement hierarchical specificity.

We justified our user interface as a menu for our final choice because it provides a place for metadata from the other design ideas (natural language and direct manipulation) to live, in case we develop those down the line as well. We anticipated that a menu not only provides a means of interaction but also storage of the state of a system. In addition, this type of user interface is common and relatively recognizable.

8.6 Evaluation

Our first research question for this project ("What constraints and capabilities should we provide end-users to give them meaningful agency over interactive data visualizations?") focused on our thematic collation and compilation of anticipated access frictions, but our following two research questions would require outside evaluation: "What qualities, challenges, and design opportunities do designers and engineers envision for a data visualization *softerware* system?" and "What qualities, challenges, and design opportunities do blind and low vision users envision for a data visualization *softerware* system?"

8.6.1 Preliminary: Visualization Practitioners

The preliminary step in our evaluation was to investigate what qualities, challenges, and design opportunities data visualization engineers and designers envision for a *softerware* system.

8.6.1.1 Recruitment

We recruited 4 data visualization practitioners, each with roles as a current or former visualization software engineer (3) or designer (1). We recruited participants from our existing network of engineers and designers, requesting participation via email. Our practitioners were not compensated for their participation and we asked them up front if they would be willing to volunteer their time for us.

8.6.1.2 Procedure

We conducted 30-minute, semi-structured, qualitative interview sessions either over Zoom or in-person. The session consisted of a 5 minute explanation, 5 minute demo of our prototype's capabilities, and a series of open-ended, semi-structured questions for 20 minutes. Our questions started with getting their thoughts on the idea, what they anticipated other developers and designers would think, what aspects of a visualization they believe end-users will want control over, issues they believed end-users would face, and what new opportunities they envision our prototype and underlying design concept of *softerware* enables.

8.6.2 Study: Blind and Low Vision Users with Accessibility Expertise

Our primary study was focused on our third research question on the qualities, challenges, and opportunities that users with disabilities, in this case users who are blind and low vision, envision for a *softerware* experience of interactive data visualizations. To explore this, we used our prototype dashboard as a design probe to stimulate concrete feedback and ideation on both the details of our prototype as well as our larger design concept of *softerware*, borrowing from Noor Hammad's method used when exploring accessibility preferences of users in novel streaming software [56].

Our study is intended to contribute qualitative knowledge, largely because we believe that statistical generalizations or controlled experiments about a particular group or subgroup of people with disabilities may actually produce knowledge that reinforces the existing problems we are trying to address. Instead, we are explicitly interested in knowledge and experiences that might exist on the margins, even knowledge as specific as a single individual's preferences. We want to explore ways that broader guidelines and design knowledge are capable of producing artifacts that still retain barriers for some individuals with disabilities. This larger challenge (of general guidelines that do not provide a meaningful fit for individuals living with disabilities) is not new to accessibility and assistive technology research [124].

And to this end, we designed our study to maximize the production of knowledge that is considerate and careful of individual differences, challenges, preferences, and envisioned opportunities.

8.6.2.1 Recruitment

Our study involved 9 total participants who are blind or low vision (see **Table 1**), all of whom are also professionals with accessibility expertise (either currently or formerly employed in an accessibility-specific role as subject matter experts). 5 of our participants self-identified as male,

Table 8.1: Study Participants

PID	Age	Gender	Disability
P1	39	F	Totally Blind
P2	38	F	Totally Blind
P3	46	M	Legally Blind
P4	28	F	Low Vision
P5	34	M	Legally Blind
P6	52	M	Totally Blind
P7	56	F	Low Vision
P8	36	M	Low Vision
P9	55	M	Totally Blind

4 as female. Average age of our participant group was 42.67 (SD = 9.99). We initially recruited 6 participants using an existing, compensated research relationship between Highsoft and an external consultancy. In addition, we recruited 3 more participants from our existing network of accessibility consultants, who were each compensated 100 USD for their time.

We anticipated that recruiting participants who not only have lived experience with a disability but also are subject matter experts in accessibility would contribute to the depth of our qualitative study as well as general breadth of considerations. We wanted to maximize the value of feedback on our work.

We reached out to all participants via email with a call for participation and participants were screened according to whether they are blind or low vision. Participants were notified in advance of compensation and that consent to participate is voluntary.

8.6.3 Procedure

Our qualitative study sessions were recorded and conducted over zoom in 3 primary phases (plus a break) during one 90 minute session. Our phases were: early interview, task-evaluation of our dashboard (menu hidden) with discussion, a break, and task-evaluation of our dashboard (menu shown) with final discussion.

8.6.3.1 Introduction and Early Interview [20min]

Our session opened with an introduction to the research team and gathering verbal consent from participants for participation. We gathered demographic information from participants and asked them about their current assistive technology use. We followed up with questions related to whether or not they customized their technology in any way, through adjusting settings, modifications, adding scripts, getting extensions, or equivalent. We then ended the opening session with ice-breaker questions about whether they can recall a chart or graph they have experienced in the past and what their favorite way to experience a chart is.

8.6.3.2 No Menu Prototype, Tasks, Discussion [30-35min]

The next phase of our session involved showing participants our demo environment (see Figure 2), except that our preferences menu was hidden. We explained what the dashboard was and entailed, including explaining each chart type shown (sankey, bar, and line) and how to read them. We gave users a short amount of time to explore the dashboard, and then notified them that in order to evaluate the effectiveness of our technology, we would be asking them to perform 2 data tasks, one elementary and one synoptic [1]. Our intention for performing tasks was not to measure speed or accuracy of the participants, but simply as a probe for eliciting feedback on the usability and effectiveness of our prototype and design.

Our first question was to answer an elementary analytical task (direct or indirect lookup), "Does petroleum or nuclear contribute the most to Electricity and Heat?" ("nuclear" was correct). Our second was a synoptic task (pattern identification or multi-value comparison), "Which energy type has the highest use in Dec *and* Jan?" ("natural gas" was correct). We gave participants a limited time (5mins total) to answer the questions and upon answering, we gave them the correct answer and asked them to explain their process of finding their answer, step-by-step.

Our final step in this process was to interview them about their perceived challenges and frustrations with the dashboard and whether anything could be changed or adjusted in order to help them complete their tasks. We followed this phase with a 10 minute break.

8.6.3.3 Menu Prototype, Tasks, Discussion [30-35min]

We opened the final phase of our study by sending our participants a new link to a version of our online dashboard that included our preferences menu. We explained the purpose of the menu and gave them 5 minutes to explore the available options.

After participants explored the menu and its effects, we repeated our tasks procedure. Participants were given 2 tasks to complete in five minutes. First, an elementary analytical task, "Where does most coal go?" ("electricity and heat") and then a synoptic task "In the summer, June through August, which energy type has the highest consumption rate?" ("petroleum").

Our final discussion focused on investigating our participant's thoughts on our prototype, the idea of preferences and customization, why they chose the customizations that they did, whether they had any new or additional ideas, considerations for other users with disabilities, and any other concerns, challenges, or feedback. We asked them specifically to consider both their personal, lived experience with their disability and assistive technology in addition to their professional expertise in accessibility.

8.7 Results

We performed two analyses from our studies, first analyzing our findings from our preliminary study from practitioners and then analyzing our results from our study with end-users. We collated our notes and transcript materials, coded them thematically, and then used affinity diagramming to group the themes that emerged from our data [58].

8.7.1 Preliminary Findings

To avoid repeating information between our preliminary study with visualization practitioners and final study with blind and low vision participants, any findings from our end-users that are echoed by our practitioners will be mentioned later. Only the findings unique to our preliminary study will be included here.

8.7.1.1 Alleviating Situational Barriers

3 of our 4 practitioners spoke about the potential benefits of end-user manipulation of a visualization for situational or contextual reasons. One participant gave the example that when giving a presentation using an existing dashboard, having the ability to manipulate features to suit layout, flow, and interactions on-demand would be valuable. Another example given was that at times a user's viewing device (such as a smartphone) can cause barriers, so softerware would be useful to have available.

8.7.1.2 Creating Potentially Harmful Visualizations

The second theme from our practitioners (3) was the concern that end-users would be able to create a misleading or harmful data visualization. For example, we have studies on how encoding area size [99] and aspect ratios [19] can be misleading or deceptive, yet being able to manipulate these for accessibility and contextual barriers (such as viewing a chart designed for desktop on a mobile phone) are important design considerations. Users may accidentally adjust features of a visualization while self-fitting that actually create problematic designs. Being able to design a system to avoid this would be important.

8.7.1.3 Designing via *Softerware*-first

The final theme from our practitioners was around authoring and design-tuning via *softerware*, where 3 participants discussed using a direct-manipulation or LLM-based *softerware* interface to author data visualizations and 1 of the 3 also mentioned that large-scale, privacy-preserving data collection from users could be used to create smarter design defaults in the future. We believe that both suggestions mirror existing work that speaks of the benefits of "design-through-use" and "continuous-co-design" [104].

8.7.2 Prototype-level Feedback

The advantage of a study with participants who had accessibility expertise in addition to their lived experience as people who are blind or low vision is that we were able to get feedback on our existing prototype as well as on our larger idea space for *softerware*.

8.7.2.1 Navigation Structure Options

While not a theme across participants, P2 mentioned that they would like to be able to navigate a data visualization using headings with their screen reader, rather than via regions. ("Regions" are

a type of semantic markup used to create programmatically recognizable organization for screen readers.) This was a suggestion that immediately led to our team iterating in parallel on ideas the next day. More than 71% of screen reader users navigate information via headings when first encountering a new web page [167]. This suggestion made sense to explore as a sensible default.

8.7.2.2 Previewing Change

Our low vision users (P4, P7, P8) requested a feature that we had originally designed but not implemented, which was to directly show what different options would look like in the preferences menu itself. For example, the "text size" options would either have a preview of the text size shown for each option in the menu (like showing "Large" in the actual resulting large text size) or with a nearby preview window that would show the result of a selection as it is being selected. Low vision users in particular often use high levels of magnification and zoom, so the live results shown in the visualization space required users to go back and forth between the menu once an option was chosen and into the chart space to find what had been affected.

8.7.2.3 Language Re-consideration

Some of our participants (P1, P2, P4, P6, P7, P8) noted ambiguity or lack of clarity in the wording we used for our menu's higher level options, such as "Audio" having options for "default," "gentle," and "complex" while lower level options that inherited these were hard to connect to. For example, "Sonification order" under "Audio" had the options "default," "sequential," or "simultaneous." "Gentle" in "Audio" would set the child setting for sonification order to "sequential," but this was unclear initially.

Other participants (P1, P2, P3, P6) noted that while the menu's focus on functional categories was helpful, it might be nice to also have a way to customize the menu itself or view it from a "disability" perspective, so they could get all the screen reader options in a single place. Users were interested in looking at all options relevant to "screen readers" or "low vision" together.

8.7.3 System-class Accessibility Findings

The next set of themes that emerged were considerations that both our end-users and our visualization practitioners shared, which we are calling *system-class* considerations, using the phrase from Chris Fleizach and Jeffrey Bigham [43]. In order for *softerware* to function at scale, certain technical and infrastructure considerations would have to be prioritized to make things possible. This theme emerged thanks to the accessibility knowledge and expertise of our end-users and engineering concerns of our practitioners.

8.7.3.1 Persistence

Every participant (P1, P2, P3, P4, P5, P6, P7, P8, P9) as well as all of our practitioners noted that the ability to create some sort of "profile" or persistent state of their customizations would be one of the most important features that would make *softerware* actually useful.

> "What if I come back to this? Will I lose this? Do I need to do it again?"—P6

We followed up by asking whether certain contexts would make persistence more or less important. We asked users whether a random website or news article with a chart in it would be worth their time, to which most users replied, "no." However, P5 noted that "This is so fun that if it was there I still might play around with it and use it, especially if I had the time." P4 related this issue to an existing frustration with video games, noting that having to set up repeated options for every game was time consuming. It would be nice if they could "do this once and forget it."

8.7.3.2 Profile Sharing

Following this theme of establishing a profile, most of the participants (P1, P3, P4, P5, P6, P8) and 2 of our practitioners also expressed interest in being able to share their own profiles or ingest settings from others, in order to save others or themselves time. In phase 1 of our procedure, we asked users about their existing levels of modification, customization, and preferences setting in their existing use of technology. While all of our participants (except for P9) customize, personalize, or modify their technology to some degree, those who were most interested in customization or spoke the most about it (P3, P4, P5, P8) were also the most passionate about being able to save *other* people time and not just themselves.

> "I customize my tech a lot. If I use something for the first time and it feels off, I find a way to fix it. But most people aren't like that; it takes too long. So I love when I can share [my modifications and customizations] with others."—P3

8.7.3.3 Cross-system Interoperability

Closely related to *persistence* and *profile-sharing* was an idea expressed by several participants (P3, P4, P5, P7, P8) that they wanted to be able to use these settings outside of Highcharts and even outside of the web. "Will this work in Microsoft Excel?" and "I use salesforce for analytics a lot and would love this there," remarked P4. However, cross-system interoperability would require multiple charting libraries being intelligent enough to ingest user settings, when most are currently incapable of even recognizing a system's "high contrast" settings being active. In addition, all 4 of our practitioners suggested that there would need to be a system in place, either at the operating system level or as some kind of service hosted by a platform, where these settings could be recognized and ingested. For cross-system interoperability to be made possible, it would require establishing standards for customization, standards for preserving user privacy, and coordination with the larger community of visualization practitioners and software providers.

8.7.4 User-Centered Findings

The last major set of themes is related to the considerations of end user experience of a *softerware* system applied in practice, including our observations about the differences between users and their choices when personalizing a data visualization interface.

8.7.4.1 Frictions in User Differences

Our first major user-centered finding was that no participant chose the same set of preferences as another. Every user discussed different reasons for justifying their choice of options. Users even chose options that others specifically emphasized were inaccessible to them. An example of this was a tension in preference for and against use of "dark mode" designs.

> "If anything has dark mode? That's great. I wish everything used dark mode."—P4

P4 mentioned that they had "night blindness" (*nyctalopia*), which is why dark mode designs are helpful for them. However, P7 also mentioned that they had progressive nyctalopia, but dark mode makes an interface "virtually impossible" to them.

> "Oh, I can't use dark mode at all. I hate when websites have [dark mode] because it can be virtually impossible to use."—P7

Any one of the designs chosen by a low vision participant would have been insufficient for providing access for any of our other low vision participants (see Figure 1).

Our blind participants also had different justifications and preferences for their text and audio customizations. Some justified their differing preferences with similar justifications, such as cognitive accessibility and text description length. For example, P9 stressed that "I prefer to keep things simple" to "avoid overwhelm" while P2 said, "more information is better than less, when it comes to data." Both P9 and P2 preferred "accessible defaults," but disagreed on what length of textual descriptions should be default.

8.7.4.2 Accessible Defaults are a Necessary Prerequisite

Several participants were concerned that this approach would allow designers and developers to continue to make inaccessible charts (P2, P6, P7) if users have the ability to *self-fit*. Participants emphasized how important it is to have strong accessibility *before* customization is introduced (P2, P4, P6, P7, P9). Even 3 of our 4 of our practitioners expressed worry that *softerware* could put a design burden on users.

P9, our only participant who almost exclusively uses default settings (and avoids mods and extensions) with their current assistive tech, stressed the importance of well-thought out defaults. It is clear that for users like P9 in particular, strong defaults are much more important than customization. Although less common, some assistive technology users are not interested in the work involved in personalization and would prefer technology to suit their needs out of the box.

This leads us to argue that there is a line between ethical use of *softerware*, which is built on top of already-accessible material, and *softerware* that is filling gaps in poor design. Designs that are lacking access that wouldn't cause any friction for someone else if they were present, such as simply having alt text, aren't in need of *softerware*, they're just in need of accessibility.

8.7.4.3 Effort-to-Outcome Ratio

As a playful rephrasing of the visually-centric (and controversial) *data-to-ink* ratio [24], we observed an *effort-to-outcome* ratio among our participants, in line with previous results from existing work [187]. Nearly all participants (P1, P2, P3, P4, P6, P7, P8, P9) noted that the work

required to interact with this menu wouldn't be worth the effort if they had to do it every time they interacted with a data visualization.

Most of our participants who used screen readers (P1, P2, P3, P6, P9) also mentioned that the menu itself was too cumbersome for navigating within and back and forth with the dashboard. Setting options was quite slow, and observing the output of a given option change, such as text verbosity or sonification type, was hard to do with accuracy. Keeping what a previous state was like in memory was hard.

One participant was interested in different ways that this process could become easier, suggesting

> "What if I could just tell it what to change while I'm listening? Like right here [navigating a chart element] what if I could just say "keep it short" or maybe "wait, tell me more."—P6

This suggests that there may be a space to explore non-visual direct manipulation *softerware* strategies.

8.8 Conclusion

In an idealized world, designers do their best to produce useful and accessible interfaces. They're concerned with making software as accessible as possible by default. But no single design is capable of perfection. *Access frictions* between accessible defaults and the needs of real individuals might always be present in software interfaces. To that aim, we hope to contribute knowledge that can inform future designers and developers to not only build accessible artifacts, but build *systems* that enable end users with disabilities to have interactive agency over their software experiences.

Our vision of data visualization *softerware* demands more involvement from research and industry. In order to offer as much value as possible to end-users, we need standards set for accessibility profiles and we need data visualization software, libraries, and applications to respect and be able to contribute to those profiles. We want to encourage researchers to investigate further the needs of people with disabilities, designers to imagine new interfaces and interaction paradigms for end users, and engineers to build robust systems that are capable of not only respecting a user's preferences and customizations, but providing persistence, interoperability, and system-class infrastructure.

Part VI

Conclusion

Chapter 9
Discussion & Future Work

9.1 What is a "tool?" A reflection on the social and material identity of tools

In the introduction of this dissertation, I use the example of a hammer: a hammer can destroy and it can construct. So is the *use* of a technology what constitutes it? Do we understand the hammer as the *thing we swing, to destroy and to build*? Should we?

This thesis engages domains of tools and tool-making for accessibility: evaluation, navigation, interaction, and personalization. But these categories for work do not fully characterize the upstream conditions that our software systems and data interfaces inherit.

In my work specifically on accessibility, a larger social reality becomes apparent that shapes the question, "what is a tool?" far more than how an individual might use one, or the domains of work that our tool-making inhabits. My research journey has navigated multiple social and political thresholds, from changes to law in the European Union, to the enactment of Title II as part of the update to the Americans with Disabilities Act. These laws have motivated a significant interest in accessibility research, solutions, guidelines, and technologies. In the midst of this, we have seen the rise of overlays and generative AI solutionism [60] and subsequent lawsuits and grass-roots resistance.

For my work, this is mostly good news. Legal change produces motivation, and even with predatory technology attempting to address real problems, pushback is widespread and active. But this paints a picture of the reality that my work inherits: many tools cannot even be used, or cease to be used, if there is not a social, political, and material set of conditions in place motivating those tools, providing resources for their construction, regulating their use, and examining the outcomes of what they accomplish. Tools and technologies are often a response to social, cultural, political, and legal realities that we first negotiate.

I recently spoke on this at a keynote in Australia, on how a hammer isn't *just* a tool and that the idea that "the only thing that matters is how a tool is used" limits how we really understand tools. Instead, I spoke about how a standard, household hammer requires iron and wood. That alone leads to a whole universe of different questions. Western Australia's conservation efforts were disrupted when a significant amount of natural iron was discovered in a wildlife preserve. So laws were passed and now iron is mined there. That iron is largely exported. And Australia then, whether with Australian iron or not, mostly imports their small tools. Iron is sent out, and through a complex network of trade (likely indirectly related to the iron), hammers are brought in. A "hammer," to even exist at all, relies on multiple levels of human governance, international relations, and complex infrastructures of trade.

And while my metaphor is largely motivated to encourage younger practitioners to consider the "iron mines" in the technologies they use, such as modern generative AI, it is also an area that is not adequately explored and addressed in terms of accessibility research.

Research on accessibility is dependent on funding, which is often dependent on political

priorities and action. Depending on the current social and political state of the world at large, accessibility research itself may never gain the opportunities required in order to innovate and produce new tools at all. And as the US's 2025 federal cuts to research demonstrated, millions of dollars devoted to accessibility research can be lost to political agendas. It is for this reason that engagement with policy recommendation and guidance is essential. Personal political activity and involvement is also essential. Researchers who genuinely believe in accessibility as a human right or as a dignity that all people deserve should work with policymakers to ensure that there are material and structural resources in place for this work to continue. We cannot naively believe that technology, divorced from the realm of social and political forces, is capable of solving accessibility barriers [156]. Without enforcement and threat of litigation, very little accessibility work has been accomplished in the past by technology companies alone.

Not featured in these chapters (as they were merely stapled in research papers from previous publications) is the policy and outreach work involved in seeing that work like *Chartability* and *Data Navigator* are used in real contexts, including by organizations that govern and influence the lives of many people. Immediate incentives to produce novelty may not be enough to sustain the larger socio-cultural and political ecosystems that our work participates in and is downstream from. We must also get involved.

9.3 Who is responsible for repair?

Lastly, I want to revisit one of my opening points, where I argue that the *tool-makers* are first responsible for repair. This is true. However, the most pressing issue I have faced in recent years is mostly unmentioned across these research projects: tool-makers might be responsible, but this is because they are the only ones who have the *power* to make things accessible. Does this always need to be the case? Can we imagine an artifact's authority over the user's ability to access being designed towards self-subversion [50] or de-centralized agency [18, 88, 116], instead? What might that look like, concretely?

In *Softerware*, we begin to engage this larger problem in terms of an idealized state where a user can repair or re-design their own experiences. But to me, this self-repair is like laying down train tracks for yourself as you move a locomotive, but then lifting up your own tracks behind you as you go. You're the only one helping yourself. This is not ideal, for you or others.

What we need are broad, lasting, infrastructural changes. On the web, this problem becomes quite difficult to solve. A personal computer or device? Again, someone can auto-design their interfaces into a better state. But back when I started *Chartability*, the WebAim Million's report showed more than 95% of the top one million website home pages contain at least one critical accessibility error. And now, more than 6 years later, that proportion is unchanged [174].

Some had imagined that generative AI would solve the massive infrastructural repair problems we face. But unfortunately, the latest WebAim Million report shows that since 2020, ARIA usage has increased and correlates to more errors, while use of tabindex on a page has increased nearly 300% and also correlates to more errors on a page. If anything, during the age of generative AI, we have seen existing bad patterns worsen in prevalence and complexity.

I firmly believe that a tools-based approach is not enough on its own. Tool-making cannot be the *only* intervention on inaccessibility. Tools and tool-making, as our thesis argues, have a

powerful role to play. But we simply can't tool our way out of failed infrastructure and inadequate policy when someone else *owns* the tools and tool-making. Visiting a website is like going into someone else's home: arranged according to their effort, tastes, and so on. If you can't access their home, you essentially need to request that they let you in personally. Website repair always falls to the owner and maintainer of a website, and they largely don't take any meaningful action.

Sidewalks outside of homes are a good parallel to this problem. Sidewalk accessibility is a massive infrastructural problem [132], and yet localities treat sidewalk maintenance in different ways: some, like where I presently live in the south hills of Pittsburgh, put the onus on the homeowner whose house and property the sidewalk touches. In other places, sidewalks are considered a public path, similar to a roadway, and are maintained through public tax and resource management. To no surprise, privately-maintained, public-access sidewalks are worse for people in pretty much every way than publicly-maintained ones [184]. This is because private homeowners don't care about sidewalk maintenance unless the city manages to fine them or they get sued.

And the web is a collection of private spaces that you visit privately. There is no truly shared, universally democratic, public space on the web. Centralization is partly to blame: sharing space while scaling leads to consolidation.

So my future work will continue to wrestle with the same tensions of scale, repair, and anti-consolidation of power, motivated by the same WebAim Million report. But now I look to questions of *democratic* and *radical* access to accessibility repair. The barriers I hope to tackle in the future are political and infrastructural. Perhaps tool-making will participate in this work, but it seems clear now from my work that the upstream technical problems and socio-political conditions that tools inherit, will likely not be addressed by tools alone.

What does "democratic" and "radical" infrastructure work look like? It will probably be an extension of *Softerware*, to some degree. I imagine future research that explores public-first spaces, ones where access is socially negotiated and repair belongs to all of us. Is this an autonomous space, like an autonomous zone [8] separate from the web? Above it, looking down into it, like shared annotation tools but capable of sharing the manipulation of websites [131]? A space with ambient co-repair, modeled after projects that bring people together [142] or that allow community "fixing" of misinformation [82]? Perhaps feminist thought on the ethics of care can help us [64, 106]? Or maybe it will be something else entirely; I'm not yet sure. But what made the web fantastic years ago is long gone; most of it has been hedged into corporate spaces that are controlled, maintained, and repaired by corporate power. And these entities are notoriously bad at repair. What I imagine in the future involves reclaiming a sense that the web is *ours*, belongs to *us*, and that ultimately *we* are responsible for making it accessible.

References

[1] W. Aigner, A. Rind, and S. Hoffmann. Comparative evaluation of an interactive time-series visualization that combines quantitative data with qualitative abstractions. *Computer Graphics Forum*, 31(3pt2):995–1004, June 2012. 8.6.3.2

[2] Obead Alhadreti and Pam Mayhew. Rethinking thinking aloud. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–12. ACM, April 2018. doi: 10.1145/3173574.3173618. URL https://doi.org/10.1145/3173574.3173618. 6.6.2

[3] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pages 111–117. IEEE, November 2005. doi: 10.1109/infvis.2005.1532136. URL https://doi.org/10.1109/infvis.2005.1532136. 3

[4] E. Bailey. Accessibility preference settings, information architecture, and internalized ableism — ericwbailey.website. https://ericwbailey.website/published/accessibility-preference-settings-information-architecture-and-interna 2024. [Online]. 8.5.2

[5] Michel Beaudouin-Lafon, Susanne Bødker, and Wendy E. Mackay. Generative theories of interaction. *ACM Transactions on Computer-Human Interaction*, 28(6):1–54, December 2021. ISSN 1557-7325. doi: 10.1145/3468505. URL https://doi.org/10.1145/3468505. 2.1.2

[6] Richard A. Becker and William S. Cleveland. Brushing scatterplots. *Technometrics*, 29 (2):127–142, May 1987. ISSN 1537-2723. doi: 10.1080/00401706.1987.10488204. URL https://doi.org/10.1080/00401706.1987.10488204. 2.2.1

[7] Dan Bennett, Alan Dix, Parisa Eslambolchilar, Feng Feng, Tom Froese, Vassilis Kostakos, Sebastien Lerique, and Niels van Berkel. Emergent interaction: Complexity, dynamics, and enaction in hci. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, pages 1–7. ACM, May 2021. doi: 10.1145/3411763.3441321. URL https://doi.org/10.1145/3411763.3441321. 2.1.2

[8] Hakim Bey. *TAZ: The temporary autonomous zone, ontological anarchy, poetic terrorism*. Autonomedia, 2003. 9.3

[9] Brandon Biggs, Christopher Toth, Tony Stockman, James M. Coughlan, and Bruce N. Walker. Evaluation of a non-visual auditory choropleth and travel map viewer. In *Proceedings of the 27th International Conference on Auditory Display (ICAD 2022)*, pages 82–90.

International Community for Auditory Display, June 2022. doi: 10.21785/icad2022.027. URL https://doi.org/10.21785/icad2022.027. PMCID: PMC10010675. 6.4.1

[10] Matt Blanco, Jonathan Zong, and Arvind Satyanarayan. Olli: An Extensible Visualization Library for Screen Reader Accessibility. In *IEEE VIS Posters*, 2022. URL https://vis.csail.mit.edu/pubs/olli. Accessed: 2026-03-30, https://vis.csail.mit.edu/pubs/olli. 5.4.3.1, 5.4.3.2, 5.4.3.3, 6.3.2

[11] M. Bostock, V. Ogievetsky, and J. Heer. D³ data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, December 2011. ISSN 1077-2626. doi: 10.1109/tvcg.2011.185. URL https://doi.org/10.1109/tvcg.2011.185. 1.2, 5.5.1.1

[12] L.H. Boyd, W.L. Boyd, and G.C. Vanderheiden. The graphical user interface: Crisis, danger, and opportunity. *Journal of Visual Impairment & Blindness*, 84(10):496–502, December 1990. doi: 10.1177/0145482x9008401002. URL https://doi.org/10.1177/0145482x9008401002. 5.3.2.3

[13] Virginia Braun and Victoria Clarke. Using thematic analysis in psychology. *Qualitative Research in Psychology*, 3(2):77–101, January 2006. ISSN 1478-0895. doi: 10.1191/1478088706qp063oa. URL https://doi.org/10.1191/1478088706qp063oa. 6.6.3

[14] S. Brewster. Visualization tools for blind people using multiple modalities. *Disability and Rehabilitation*, 24(11-12):613–621, January 2002. ISSN 1464-5165. doi: 10.1080/09638280110111388. URL https://doi.org/10.1080/09638280110111388. 2.2.3, 6.3.1

[15] Jennifer L. Burke, Matthew S. Prewett, Ashley A. Gray, Liuquin Yang, Frederick R. B. Stilson, Michael D. Coovert, Linda R. Elliot, and Elizabeth Redden. Comparing the effects of visual-auditory and visual-tactile feedback on user performance. In *Proceedings of the 8th international conference on Multimodal interfaces*, ICMI06, pages 108–117. ACM, November 2006. doi: 10.1145/1180995.1181017. URL https://doi.org/10.1145/1180995.1181017. 1.1

[16] Matthew Butler, Leona M Holloway, Samuel Reinders, Cagatay Goncu, and Kim Marriott. Technology developments in touch-based accessible graphics: A systematic review of research 2010-2020. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, pages 1–15. ACM, May 2021. doi: 10.1145/3411764.3445207. URL https://doi.org/10.1145/3411764.3445207. 6.3.1

[17] F. Cabitza and C. Simone. Malleability in the hands of end-users. In *New Perspectives in End-User Development*, pages 137–163. Springer International Publishing, 2017. 8.4.1.1

[18] Katherine Casey. Radical decentralization: Does community-driven development work? *Annual Review of Economics*, 10(1):139–163, August 2018. ISSN 1941-1391. doi: 10.1146/annurev-economics-080217-053339. URL http://dx.doi.org/10.1146/annurev-economics-080217-053339. 9.3

[19] C. R. Ceja, C. M. McColeman, C. Xiong, and S. L. Franconeri. Truth or square: As-

pect ratio biases recall of position encodings. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1054–1062, February 2021. 8.7.1.2

[20] Alex Chaparro and Maria Chaparro. Applications of Color in Design for Color-Deficient Users. *Ergonomics in Design*, 25(1):23–30, 1 2017. doi: 10.1177/1064804616635382. Accessed: 2021-09-06. 2.2.3

[21] Pramod Chundury, Biswaksen Patnaik, Yasmin Reyazuddin, Christine Tang, Jonathan Lazar, and Niklas Elmqvist. Towards understanding sensory substitution for accessible visualization: An interview study. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1084–1094, January 2022. doi: 10.1109/tvcg.2021.3114829. URL https://doi.org/10.1109/tvcg.2021.3114829. 2.2.3, 5.3.1.2, 6.3.1

[22] Pramod Chundury, Urja Thakkar, Yasmin Reyazuddin, J. Bern Jordan, Niklas Elmqvist, and Jonathan Lazar. Understanding the visualization and analytics needs of blind and low-vision professionals. In *The 26th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '24, pages 1–5. ACM, October 2024. doi: 10.1145/3663548.3688496. URL https://doi.org/10.1145/3663548.3688496. 2.1.1

[23] Clare Cullen and Oussama Metatla. Co-designing inclusive multisensory story mapping with children with mixed visual abilities. In *Proceedings of the 18th ACM International Conference on Interaction Design and Children*, IDC '19, pages 361–373. ACM, June 2019. doi: 10.1145/3311927.3323146. URL https://doi.org/10.1145/3311927.3323146. 2.2.3, 6.4

[24] d. akbaba, J. Wilburn, M. T. Nance, and M. Meyer. Manifesto for putting "chartjunk" in the trash 2021! arXiv, 2021. 8.7.4.3

[25] Lilian de Greef, Dominik Moritz, and Cynthia Bennett. Interdependent variables: Remotely designing tactile graphics for an accessible workflow. In *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–6. ACM, October 2021. doi: 10.1145/3441852.3476468. URL https://doi.org/10.1145/3441852.3476468. 5.5.3.1, 6.4

[26] Deque. Automated Testing Identifies 57 Percent of Digital Accessibility Issues. Technical report, Deque, 3 2021. Accessed: 2021-09-06. 3

[27] Alan Dix. Designing for appropriation. In *Electronic Workshops in Computing*. BCS Learning & Development, September 2007. doi: 10.14236/ewic/hci2007.53. URL https://doi.org/10.14236/ewic/hci2007.53. 2.1.2

[28] Dot Pad inc. Dot pad - the first tactile graphics display for the visually impaired. https://pad.dotincorp.com/, 2020. [Accessed 01-04-2025]. 5.5.3.1

[29] Paul Dourish. The appropriation of interactive technologies: Some lessons from placeless documents. *Computer Supported Cooperative Work (CSCW)*, 12(4):465–490, December 2003. doi: 10.1023/a:1026149119426. URL https://doi.org/10.1023/a:1026149119426. 5.6

[30] Sebastian Draxler, Gunnar Stevens, Martin Stein, Alexander Boden, and David Randall.

Supporting the social context of technology appropriation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 2835–2844. ACM, May 2012. doi: 10.1145/2207676.2208687. URL https://doi.org/10.1145/2207676.2208687. 2.1.2

[31] Eloi Durant, Mathieu Rouard, Eric W. Ganko, Cedric Muller, Alan M. Cleary, Andrew D. Farmer, Matthieu Conte, and Francois Sabot. Ten simple rules for developing visualization tools in genomics. *PLOS Computational Biology*, 18(11):e1010622, November 2022. doi: 10.1371/journal.pcbi.1010622. URL https://doi.org/10.1371/journal.pcbi.1010622. 2.2.2.2, 5.3.1.3

[32] Frank Elavsky. Method and system for accessible data visualization on a web platform. *Defensive Publication Series*, 4220, April 2021. 5.4.3.3

[33] Frank Elavsky and Cindy Xiong Bearfield. Playing telephone with generative models: "verification disability," "compelled reliance," and accessibility in data visualization. In *2025 IEEE Workshop on Accessible Data Visualization (AccessViz)*, pages 14–24. IEEE, November 2025. doi: 10.1109/accessviz68666.2025.00008. URL https://doi.org/10.1109/accessviz68666.2025.00008. 6.3.1

[34] Frank Elavsky, Cynthia Bennett, and Dominik Moritz. How accessible is my visualization? evaluating visualization accessibility with chartability. *Computer Graphics Forum*, 41(3):57–70, June 2022. doi: 10.1111/cgf.14522. URL https://doi.org/10.1111/cgf.14522. 2.2.2.1, 2.2.2.2, 2.2.3, 5.3.1, 5.3.1.3, 5.5.1.1, 5.6, 6.4.3, 6.8.2, 8.3.1, 8.5.1

[35] Frank Elavsky, Lucas Nadolskis, and Dominik Moritz. Data navigator: An accessibility-centered data navigation toolkit. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–11, 2023. ISSN 2160-9306. doi: 10.1109/tvcg.2023.3327393. URL https://doi.org/10.1109/tvcg.2023.3327393. 2.1.2, 6.2, 6.3.1, 6.3.2, 6.4, 6.5.1

[36] A. Endert, L. Bradel, and C. North. Beyond control panels: Direct manipulation for visual analytics. *IEEE Computer Graphics and Applications*, 33(4):6–13, July 2013. ISSN 0272-1716. doi: 10.1109/mcg.2013.53. URL https://doi.org/10.1109/mcg.2013.53. 6.2

[37] Will Epperson, Vaishnavi Gorantla, Dominik Moritz, and Adam Perer. Dead or alive: Continuous data profiling for interactive data science. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–11, 2023. ISSN 2160-9306. doi: 10.1109/tvcg.2023.3327367. URL https://doi.org/10.1109/tvcg.2023.3327367. 2.2.1

[38] Pavithra Eswaramoorthy, Frank Elavsky, Tania Allard, and gabalafou. Quansight-Labs/bokeh-a11y-audit: v1.0.0, February 2025. URL https://bokeh-a11y-audit.readthedocs.io/. 6.4.3

[39] Danyang Fan, Alexa Fay Siu, Hrishikesh Rao, Gene Sung-Ho Kim, Xavier Vazquez, Lucy Greco, Sile O'Modhrain, and Sean Follmer. The accessibility of data visualizations on the web for screen reader users: Practices and experiences during covid-19. *ACM Transactions on Accessible Computing*, 16(1):1–29, March 2023. doi: 10.1145/3557899. URL

https://doi.org/10.1145/3557899. 1.1, 2.2.2.1, 2.2.2.2, 2.2.3, 5.3.1, 5.3.1.2, 5.3.1.3, 6.3.2

[40] Ali Mazraeh Farahani, Peyman Adibi, Mohammad Saeed Ehsani, Hans-Peter Hutter, and Alireza Darvishy. Automatic chart understanding: A review. *IEEE Access*, 11:76202–76221, 2023. ISSN 2169-3536. doi: 10.1109/access.2023.3298050. URL https://doi.org/10.1109/access.2023.3298050. 6.3.1

[41] Stephen Few. Visual Business Intelligence - Data Visualization and the Blind. https://www.perceptualedge.com/blog/?p=1756, 2013. [Accessed 05-04-2025]. 1.1

[42] Danyel Fisher, Igor Popov, Steven Drucker, and m.c. schraefel. Trust me, i'm partially right. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1673–1682. ACM, May 2012. doi: 10.1145/2207676.2208294. URL https://doi.org/10.1145/2207676.2208294. 6.2

[43] C. Fleizach and J. P. Bigham. System-class accessibility: The architectural support for making a whole system usable by people with disabilities. *Queue*, 22(5):28–39, October 2024. 8.7.3

[44] John H. Flowers, Dion C. Buhman, and Kimberly D. Turnage. Cross-Modal Equivalence of Visual and Auditory Scatterplots for Exploring Bivariate Data Samples. *Human Factors*, 39(3):341–351, 9 1997. doi: 10.1518/001872097778827151. Accessed: 2021-09-03. 2.2.3, 3

[45] Steven L. Franconeri, Lace M. Padilla, Priti Shah, Jeffrey M. Zacks, and Jessica Hullman. The science of visual data communication: What works. *Psychological Science in the Public Interest*, 22(3):110–161, December 2021. ISSN 1539-6053. doi: 10.1177/15291006211051956. URL https://doi.org/10.1177/15291006211051956. 1.2

[46] Emden R. Gansner and Stephen C. North. An open graph visualization system and its applications to software engineering. *Software: Practice and Experience*, 30(11): 1203–1233, September 2000. doi: 10.1002/1097-024x(200009)30:11⟨1203::aid-spe338⟩3.0.co;2-n. URL https://doi.org/10.1002/1097-024x(200009)30:11<1203::aid-spe338>3.0.co;2-n. 5.4.1.1

[47] Stéphanie Giraud and Christophe Jouffrais. Empowering low-vision rehabilitation professionals with "do-it-yourself" methods. In *Lecture Notes in Computer Science*, pages 61–68. Springer International Publishing, 2016. ISBN 9783319412665. doi: 10.1007/978-3-319-41267-2_9. URL https://doi.org/10.1007/978-3-319-41267-2_9. 5.3.3

[48] Michele E. Gloede and Melissa K. Gregg. The fidelity of visual and auditory memory. *Psychonomic Bulletin & Review*, 26(4):1325–1332, August 2019. ISSN 1531-5320. doi: 10.3758/s13423-019-01597-7. URL https://doi.org/10.3758/s13423-019-01597-7. 1.1

[49] A. Jonathan R. Godfrey, Paul Murrell, and Volker Sorge. An accessible interaction model for data visualisation in statistics. In *Lecture Notes in Com-*

puter Science, pages 590–597. Springer International Publishing, 2018. ISBN 9783319942766. doi: 10.1007/978-3-319-94277-3_92. URL https://doi.org/10.1007/978-3-319-94277-3_92. 2.2.3, 5.2, 5.3.1.1, 5.3.2.1

[50] David Graeber and Charlie Rose. Never cower to authority, 2006. URL https://www.youtube.com/watch?v=qt5op6wft-8. Accessed [2026]. 9.3

[51] David Gray Widder, Sarah West, and Meredith Whittaker. Open (for business): Big tech, concentrated power, and the political economy of open ai. *SSRN Electronic Journal*, 2023. ISSN 1556-5068. doi: 10.2139/ssrn.4543807. URL https://doi.org/10.2139/ssrn.4543807. 1.2

[52] Catherine Grevet and Eric Gilbert. Piggyback prototyping. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 4047–4056. ACM, April 2015. doi: 10.1145/2702123.2702395. URL https://doi.org/10.1145/2702123.2702395. 2.1.2

[53] Nathan Hahn, Joseph Chang, Ji Eun Kim, and Aniket Kittur. The knowledge accelerator. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI'16, pages 2258–2270. ACM, May 2016. doi: 10.1145/2858036.2858364. URL https://doi.org/10.1145/2858036.2858364. 2.1.1

[54] Jen Hale. Extensive digitization of tactile map collection. https://www.perkins.org/extensive-digitization-of-tactile-map-collection/, 2016. Perkins Archives Blog, Perkins School for the Blind. July 2016. Accessed: 2022-02-23. 2.2.3

[55] Foad Hamidi, Patrick Mbullo, Deurence Onyango, Michaela Hynie, Susan McGrath, and Melanie Baljko. Participatory design of diy digital assistive technology in western kenya. In *Proceedings of the Second African Conference for Human Computer Interaction: Thriving Communities*, AfriCHI '18, pages 1–11. ACM, December 2018. doi: 10.1145/3283458.3283478. URL https://doi.org/10.1145/3283458.3283478. 2.1.1

[56] Noor Hammad, Frank Elavsky, Sanika Moharana, Jessie Chen, Seyoung Lee, Patrick Carrington, Dominik Moritz, Jessica Hammer, and Erik Harpstead. Exploring the affordances of game-aware streaming to support blind and low vision viewers: A design probe study. In *The 26th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '24, pages 1–13. ACM, October 2024. doi: 10.1145/3663548.3675665. URL https://doi.org/10.1145/3663548.3675665. 6.2, 6.9, 8.6.2

[57] A. Hamraie. Making access critical: Disability, race, and gender in environmental design. https://belonging.berkeley.edu/aimi-hamraie-making-access-critical-disability-race-and-gender-enviror 2019. [Online]. 2.2.3, 3, 8.3.1

[58] G. Harboe and E. M. Huang. Real-world affinity diagramming practices: Bridging the paper-digital gap. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2015. doi: 10.1145/2702123.2702561. 8.7

[59] Gunnar Harboe and Elaine M. Huang. Real-world affinity diagramming practices. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Sys-*

tems, CHI '15, pages 95–104. ACM, April 2015. doi: 10.1145/2702123.2702561. URL https://doi.org/10.1145/2702123.2702561. 6.6.3

[60] Parker Hartman and Tim Gorichanaz. Evaluating ai-powered website accessibility overlays. In *Proceedings of the 27th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '25, page 1–4. ACM, October 2025. doi: 10.1145/3663547.3759761. URL http://dx.doi.org/10.1145/3663547.3759761. 9.1

[61] Gillian R. Hayes. The relationship of action research to human-computer interaction. *ACM Transactions on Computer-Human Interaction*, 18(3):1–20, July 2011. ISSN 1073-0516. doi: 10.1145/1993060.1993065. URL https://doi.org/10.1145/1993060.1993065. 6.2, 6.4

[62] Jeffrey Heer and Maneesh Agrawala. Design considerations for collaborative visual analytics. In *2007 IEEE Symposium on Visual Analytics Science and Technology*, pages 171–178. IEEE, October 2007. doi: 10.1109/vast.2007.4389011. URL https://doi.org/10.1109/vast.2007.4389011. 2.2.1, 6.2

[63] Jeffrey Heer and Dominik Moritz. Mosaic: An architecture for scalable & interoperable data views. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–11, 2023. ISSN 1077-2626, 1941-0506, 2160-9306. doi: 10.1109/tvcg.2023.3327189. URL https://doi.org/10.1109/tvcg.2023.3327189. 1.2, 2.2.1, 3

[64] Ana O Henriques, Anna R. L. Carter, Beatriz Severes, Reem Talhouk, Angelika Strohmayer, Ana Cristina Pires, Colin M. Gray, Kyle Montague, and Hugo Nicolau. A feminist care ethics toolkit for community-based design: Bridging theory and practice. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, page 1–26. ACM, April 2025. doi: 10.1145/3706598.3713950. URL http://dx.doi.org/10.1145/3706598.3713950. 9.3

[65] Thomas Hermann, Andy Hunt, and John G Neuhoff, editors. *The Sonification Handbook*. Logos Verlag Berlin, Berlin, Germany, December 2011. ISBN: 978-3-8325-2819-5. 6.3.1

[66] Jaylin Herskovitz, Yi Fei Cheng, Anhong Guo, Alanson P. Sample, and Michael Nebeling. Xspace: An augmented reality toolkit for enabling spatially-aware distributed collaboration. *Proceedings of the ACM on Human-Computer Interaction*, 6(ISS):277–302, November 2022. ISSN 2573-0142. doi: 10.1145/3567721. URL https://doi.org/10.1145/3567721. 2.1.2

[67] Jaylin Herskovitz, Andi Xu, Rahaf Alharbi, and Anhong Guo. Hacking, switching, combining: Understanding and supporting diy assistive technology design by blind people. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, pages 1–17. ACM, April 2023. doi: 10.1145/3544548.3581249. URL https://doi.org/10.1145/3544548.3581249. 2.1.2

[68] Jaylin Herskovitz, Andi Xu, Rahaf Alharbi, and Anhong Guo. Programally: Creating custom visual access programs via multi-modal end-user programming. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, UIST '24, pages 1–15. ACM, October 2024. doi: 10.1145/3654777.3676391. URL https:

//doi.org/10.1145/3654777.3676391. 2.1.2

[69] L. Hickman and A. Hagerty. Standardised access: the tension between scale and fit. https://www.adalovelaceinstitute.org/blog/standardised-access-tension-scale-fit/, 2021. [Online]. 8.4.1.2

[70] Highsoft. Highcharts - interactive charting library for developers. https://www.highcharts.com/, 2009. [Online]. 8.2

[71] Highsoft. Stacked column demo. https://www.highcharts.com/demo/column-stacked, 2018. Accessed: 2022-12-31. 5.5.1

[72] Highsoft. Highcharts accessibility module: information and demos. https://highcharts.com/docs/accessibility/accessibility-module, 2018. Accessed: 2021-09-06. 5.3.2.1

[73] Highsoft. Highcharts: Render millions of chart points with the boost module. https://www.highcharts.com/blog/tutorials/highcharts-high-performance-boost-module/, 2020. Accessed: 2022-11-20. 5.5.1

[74] Leona Holloway, Peter Cracknell, Kate Stephens, Melissa Fanshawe, Samuel Reinders, Kim Marriott, and Matthew Butler. Refreshable tactile displays for accessible data visualisation, 2024. URL https://arxiv.org/abs/2401.15836. 6.3.1

[75] Leona M Holloway, Cagatay Goncu, Alon Ilsar, Matthew Butler, and Kim Marriott. Infosonics: Accessible infographics for people who are blind using sonification and voice. In *CHI Conference on Human Factors in Computing Systems*, CHI '22, pages 1–13. ACM, April 2022. doi: 10.1145/3491102.3517465. URL https://doi.org/10.1145/3491102.3517465. 6.3.1

[76] Jonathan Hook, Sanne Verbaan, Abigail Durrant, Patrick Olivier, and Peter Wright. A study of the challenges related to diy assistive technology in the context of children with disabilities. In *Proceedings of the 2014 conference on Designing interactive systems*, DIS '14, pages 597–606. ACM, June 2014. doi: 10.1145/2598510.2598530. URL https://doi.org/10.1145/2598510.2598530. 2.1.1

[77] Stacy Hsueh, Beatrice Vincenzi, Akshata Murdeshwar, and Marianela Ciolfi Felice. Cripping data visualizations: Crip technoscience as a critical lens for designing digital access. In *The 25th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '23, page 1–16. ACM, October 2023. doi: 10.1145/3597638.3608427. URL http://dx.doi.org/10.1145/3597638.3608427. 1.1, 2.2.3, 3, 8.3.1

[78] Amy Hurst and Shaun Kane. Making "making" accessible. In *Proceedings of the 12th International Conference on Interaction Design and Children*, IDC '13, pages 635–638, New York, NY, USA, 6 2013. Association for Computing Machinery. doi: 10.1145/2485760.2485883. Accessed: 2021-09-03. 2.1.1, 5.3.3

[79] Amy Hurst and Jasmine Tobias. Empowering individuals with do-it-yourself assistive technology. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*, ASSETS '11, pages 11–18. ACM, October 2011.

doi: 10.1145/2049536.2049541. URL https://doi.org/10.1145/2049536.2049541. 2.1.1

[80] Edwin L. Hutchins, James D. Hollan, and Donald A. Norman. Direct manipulation interfaces. *Human–Computer Interaction*, 1(4):311–338, December 1985. ISSN 0737-0024. doi: 10.1207/s15327051hci0104_2. URL https://doi.org/10.1207/s15327051hci0104_2. 6.2

[81] Hilary Hutchinson, Wendy Mackay, Bo Westerlund, Benjamin B. Bederson, Allison Druin, Catherine Plaisant, Michel Beaudouin-Lafon, Stéphane Conversy, Helen Evans, Heiko Hansen, Nicolas Roussel, and Björn Eiderbäck. Technology probes. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 17–24. ACM, April 2003. doi: 10.1145/642611.642616. URL https://doi.org/10.1145/642611.642616. 6.2, 6.9

[82] Farnaz Jahanbakhsh, Amy X. Zhang, Karrie Karahalios, and David R. Karger. Our browser extension lets readers change the headlines on news articles, and you won't believe what they did! *Proceedings of the ACM on Human-Computer Interaction*, 6 (CSCW2):1–33, November 2022. ISSN 2573-0142. doi: 10.1145/3555643. URL http://dx.doi.org/10.1145/3555643. 9.3

[83] Chutian Jiang, Wentao Lei, Emily Kuang, Teng Han, and Mingming Fan. Understanding strategies and challenges of conducting daily data analysis (dda) among blind and low-vision people. In *The 25th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '23, pages 1–15. ACM, October 2023. doi: 10.1145/3597638.3608423. URL https://doi.org/10.1145/3597638.3608423. 2.1.1

[84] Shuli Jones, Isabella Pedraza Pineros, Daniel Hajas, Jonathan Zong, and Arvind Satyanarayan. "customization is key": Reconfigurable textual tokens for accessible data visualizations. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, pages 1–14. ACM, May 2024. doi: 10.1145/3613904.3641970. URL https://doi.org/10.1145/3613904.3641970. 3, 6.3.1, 6.5.2, 8.3.3

[85] Shakila Cherise S Joyner, Amalia Riegelhuth, Kathleen Garrity, Yea-Seul Kim, and Nam Wook Kim. Visualization accessibility in the wild: Challenges faced by visualization designers. In *CHI Conference on Human Factors in Computing Systems*, CHI '22, pages 1–19. ACM, April 2022. doi: 10.1145/3491102.3517630. URL https://doi.org/10.1145/3491102.3517630. 2.2.3, 3, 6.2, 6.3.2, 6.8.2, 8.3.1

[86] Crescentia Jung, Shubham Mehta, Atharva Kulkarni, Yuhang Zhao, and Yea-Seul Kim. Communicating visualizations without visuals: Investigation of visualization alternative text for people with visual impairments. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1095–1105, January 2022. doi: 10.1109/tvcg.2021.3114846. URL https://doi.org/10.1109/tvcg.2021.3114846. 2.2.3, 5.2, 5.3.1.2, 6.3.1

[87] Hyeonsu B. Kang, Xin Qian, Tom Hope, Dafna Shahaf, Joel Chan, and Aniket Kittur. Augmenting scientific creativity with an analogical search engine. *ACM Transactions on Computer-Human Interaction*, 29(6):1–36, December 2022. ISSN 1557-7325. doi: 10.1145/3530013. URL https://doi.org/10.1145/3530013. 2.1.1

[88] Os Keyes, Josephine Hoy, and Margaret Drouhard. Human-computer insurrection: Notes on an anarchist hci. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13. ACM, May 2019. doi: 10.1145/3290605.3300569. URL http://dx.doi.org/10.1145/3290605.3300569. 9.3

[89] Gyeongri Kim, Jiho Kim, and Yea-Seul Kim. "explain what a treemap is": Exploratory investigation of strategies for explaining unfamiliar chart to blind and low vision users. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, pages 1–13. ACM, April 2023. doi: 10.1145/3544548.3581139. URL https://doi.org/10.1145/3544548.3581139. 6.3.1

[90] Hyeok Kim, Yea-Seul Kim, and Jessica Hullman. Erie: A declarative grammar for data sonification. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, pages 1–19. ACM, May 2024. doi: 10.1145/3613904.3642442. URL https://doi.org/10.1145/3613904.3642442. 6.3.1

[91] Jiho Kim, Arjun Srinivasan, Nam Wook Kim, and Yea-Seul Kim. Exploring chart question answering for blind and low vision users. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–15. ACM, April 2023. doi: 10.1145/3544548.3581532. URL https://doi.org/10.1145/3544548.3581532. 2.2.3, 5.3.1.2, 6.3.1

[92] N. W. Kim, S. C. Joyner, A. Riegelhuth, and Y. Kim. Accessible visualization: Design space, opportunities, and challenges. *Computer Graphics Forum*, 40(3):173–188, June 2021. doi: 10.1111/cgf.14298. URL https://doi.org/10.1111/cgf.14298. 2.2.3, 5.6, 6.3.1

[93] N. W. Kim, G. Ataguba, S. C. Joyner, Chuangdian Zhao, and Hyejin Im. Beyond alternative text and tables: Comparative analysis of visualization tools and accessibility methods. *Computer Graphics Forum*, 42(3):323–335, June 2023. ISSN 1467-8659. doi: 10.1111/cgf.14833. URL https://doi.org/10.1111/cgf.14833. 6.3.1, 6.3.2, 8.5.1

[94] Kitware, Inc. *The Visualization Toolkit user's guide*, January 2003. URL http://www.kitware.com/publications/item/view/1269. 2.1.2

[95] A. J. Ko et al. The state of the art in end-user software engineering. *ACM Computing Surveys*, 43(3):1–44, April 2011. 8.4.1.1

[96] Bongshin Lee, Eun Kyoung Choe, Petra Isenberg, Kim Marriott, and John Stasko. Reaching broader audiences with data visualization. *IEEE Computer Graphics and Applications*, 40(2):82–90, 2020. doi: 10.1109/MCG.2020.2968244. 2.2.3

[97] Chien-Yu Lin and Yu-Ming Chang. Increase in physical activities in kindergarten children with cerebral palsy by employing makey–makey-based task systems. *Research in Developmental Disabilities*, 35(9):1963–1969, September 2014. doi: 10.1016/j.ridd.2014.04.028. URL https://doi.org/10.1016/j.ridd.2014.04.028. 5.3.3

[98] Zhicheng Liu and Jeffrey Heer. The effects of interactive latency on exploratory visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2122–2131, December 2014. ISSN 1077-2626. doi: 10.1109/tvcg.2014.2346452. URL https:

//doi.org/10.1109/tvcg.2014.2346452. 2.2.1, 3, 6.2

[99] L. Y. Lo, A. Gupta, K. Shigyo, A. Wu, E. Bertini, and H. Qu. Misinformed by visualization: What do we learn from misinformative visualizations? *Computer Graphics Forum*, 41(3):515–525, June 2022. 8.7.1.2

[100] Panagiotis Louridas. Design as bricolage: anthropology meets design thinking. *Design Studies*, 20(6):517–535, November 1999. doi: 10.1016/s0142-694x(98)00044-1. URL https://doi.org/10.1016/s0142-694x(98)00044-1. 5.6

[101] Alan Lundgard and Arvind Satyanarayan. Accessible visualization via natural language descriptions: A four-level model of semantic content. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1073–1083, January 2022. doi: 10.1109/tvcg.2021.3114770. URL https://doi.org/10.1109/tvcg.2021.3114770. 2.2.2.2, 2.2.3, 5.3.1.2, 5.3.1.3, 5.5.1.1, 5.5.3, 5.6, 6.3.1

[102] Alan Lundgard, Crystal Lee, and Arvind Satyanarayan. Sociotechnical considerations for accessible visualization design. In *2019 IEEE Visualization Conference (VIS)*, pages 16–20. IEEE, October 2019. doi: 10.1109/visual.2019.8933762. URL https://doi.org/10.1109/visual.2019.8933762. 2.2.2.2, 5.3.1.3, 5.5.1.1, 5.5.3, 5.6, 6.3.2

[103] Dor Ma'ayan, Wode Ni, Katherine Ye, Chinmay Kulkarni, and Joshua Sunshine. How domain experts create conceptual diagrams and implications for tool design. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, pages 1–14, New York, NY, USA, April 2020. ACM. ISBN 9781450367080. doi: 10.1145/3313831.3376253. URL https://doi.org/10.1145/3313831.3376253. 5.4, 5.6

[104] M. Maceli and M. E. Atwood. "human crafters" once again: Supporting users as designers in continuous co-design. In *End-User Development*, pages 9–24. Springer Berlin Heidelberg, 2013. 8.3.3, 8.7.1.3

[105] Kelly Mack, Emma McDonnell, Dhruv Jain, Lucy Lu Wang, Jon E. Froehlich, and Leah Findlater. What do we mean by "accessibility research"? In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, pages 1–18, New York, NY, USA, May 2021. ACM. ISBN 978-1-4503-8096-6. doi: 10.1145/3411764.3445412. URL https://doi.org/10.1145/3411764.3445412. Accessed: 2022-02-22. 2.2.2.1, 2.2.3

[106] Els Maeckelberghe. Feminist ethic of care: A third alternative approach. *Health Care Analysis*, 12(4):317–327, December 2004. ISSN 1573-3394. doi: 10.1007/s10728-004-6639-6. URL http://dx.doi.org/10.1007/s10728-004-6639-6. 9.3

[107] R G Maling and D C Clarkson. Electronic controls for the tetraplegic (possum) (patient operated selector mechanisms—p.o. s.m.). *Spinal Cord*, 1(3):161–174, November 1963. ISSN 1476-5624. doi: 10.1038/sc.1963.15. URL http://dx.doi.org/10.1038/sc.1963.15. 3

[108] Douglass L. Mansur, Merra M. Blattner, and Kenneth I. Joy. Sound graphs: A numerical data analysis method for the blind. *Journal of Medical Systems*, 9(3):163–174, June 1985.

ISSN 1573-689X. doi: 10.1007/bf00996201. URL https://doi.org/10.1007/bf00996201. 2.2.3, 6.3.1

[109] Deborah Marks. Models of disability. *Disability and Rehabilitation*, 19(3):85–91, January 1997. ISSN 1464-5165. doi: 10.3109/09638289709166831. URL https://doi.org/10.3109/09638289709166831. 1.1

[110] Kim Marriott, Bongshin Lee, Matthew Butler, Ed Cutrell, Kirsten Ellis, Cagatay Goncu, Marti Hearst, Kathleen McCoy, and Danielle Albers Szafir. Inclusive data visualization for people with disabilities. *Interactions*, 28(3):47–51, May 2021. doi: 10.1145/3457875. URL https://doi.org/10.1145/3457875. 2.2.1, 2.2.2.1, 2.2.3, 5.3.1, 5.3.3, 5.6, 6.3.1, 8.3.1

[111] Kim Marriott, Matthew Butler, Leona Holloway, William Jolley, Bongshin Lee, Bruce Maguire, and Danielle Albers Szafir. From vision to touch: Bridging visual and tactile principles for accessible data representation. *IEEE Transactions on Visualization and Computer Graphics*, 32(1):659–669, January 2026. ISSN 2160-9306. doi: 10.1109/tvcg.2025.3634254. URL https://doi.org/10.1109/tvcg.2025.3634254. 6.3.1

[112] Rubén Alcaraz Martínez, Mireia Ribera Turró, and Toni Granollers Saltiveri. Methodology for heuristic evaluation of the accessibility of statistical charts for people with low vision and color vision deficiency, February 2021. URL https://doi.org/10.21203/rs.3.rs-156959/v1. 2.2.3

[113] David K. McGookin and Stephen A. Brewster. Soundbar: Exploiting multiple views in multimodal graph browsing. In *Proceedings of the 4th Nordic conference on Human-computer interaction: Changing roles*, NordiCHI '06, pages 145–154, New York, NY, USA, 10 2006. Association for Computing Machinery. doi: 10.1145/1182475.1182491. Accessed: 2021-09-03. 2.2.3

[114] Andrew M. McNutt. No grammar to rule them all: A survey of json-style dsls for visualization. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–11, 2022. ISSN 2160-9306. doi: 10.1109/tvcg.2022.3209460. URL https://doi.org/10.1109/tvcg.2022.3209460. 6.2

[115] Catherine Mei∗, Josh Pollock∗, Daniel Hajas, Jonathan Zong, and Arvind Satyanarayan. Benthic: Perceptually congruent structures for accessible charts and diagrams. In *Proceedings of the 27th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '25, pages 1–17. ACM, October 2025. doi: 10.1145/3663547.3746342. URL https://doi.org/10.1145/3663547.3746342. 6.3.1, 6.3.2

[116] Giles Mohan and Kristian Stokke. Participatory development and empowerment: The dangers of localism. *Third World Quarterly*, 21(2):247–268, April 2000. ISSN 1360-2241. doi: 10.1080/01436590050004346. URL http://dx.doi.org/10.1080/01436590050004346. 9.3

[117] Dominik Moritz, Bill Howe, and Jeffrey Heer. Falcon. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 1–11. ACM, May 2019. doi: 10.1145/3290605.3300924. URL https://doi.org/10.1145/3290605.3300924. 3

[118] Peya Mowar, Aaron Steinfeld, and Jeffrey P. Bigham. iTagPDF: Towards finally automating PDF accessibility. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*, CHI '26. ACM, 2026. doi: 10.1145/3772318.3790289. URL https://peyajm29.github.io/itagpdf.html. Best Paper Award. 6.3.2

[119] Jamie R. Nuñez, Christopher R. Anderton, and Ryan S. Renslow. Optimizing colormaps with consideration for color vision deficiency to enable accurate interpretation of scientific data. *PLOS ONE*, 13(7):e0199239, August 2018. ISSN 1932-6203. doi: 10.1371/journal.pone.0199239. URL https://doi.org/10.1371/journal.pone.0199239. 2.2.3

[120] CA Okoro, ND Hollis, AC Cyrus, and S. Griffin-Blake. Prevalence of disabilities and health care access by disability status and type among adults — united states, 2016. *Centers for Disease Control and Prevention MMWR*, 67:882–887, 2018. doi: 10.15585/mmwr.mm6732a3. 1.1

[121] Manuel M. Oliveira. Towards More Accessible Visualizations for Color-Vision-Deficient Individuals. *Computing in Science Engineering*, 15(5):80–87, 9 2013. doi: 10.1109/MCSE.2013.113. Conference Name: Computing in Science Engineering. 2.2.3

[122] G. Pallapa, S. K. Das, M. Di Francesco, and T. Aura. Adaptive and context-aware privacy preservation exploiting user interactions in smart environments. *Pervasive and Mobile Computing*, 12:232–243, June 2014. 8.3.2

[123] Fabio Pittarello and Manuel Semenzato. Towards a data physicalization toolkit for non-sighted users. In *2024 IEEE 21st Consumer Communications & Networking Conference (CCNC)*, pages 1–6. IEEE, January 2024. doi: 10.1109/ccnc51664.2024.10454861. URL https://doi.org/10.1109/ccnc51664.2024.10454861. 2.1.2

[124] Christopher Power, André Freire, Helen Petrie, and David Swallow. Guidelines are only half of the story: Accessibility problems encountered by blind users on the web. In *Proceedings of the SIGCHI conference on human factors in computing systems*, CHI '12, pages 433–442, New York, NY, USA, 2012. Association for Computing Machinery. doi: 10.1145/2207676.2207736. 3, 8.6.2

[125] Lauren Race, Chancey Fleet, Danielle Montour, Lindsay Yazzolino, Marco Salsiccia, Claire Ferrari, Sheri Wells-Jensen, and Amy Hurst. Designing while blind: Nonvisual tools and inclusive workflows for tactile graphic creation. In *The 25th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '23, pages 1–8. ACM, October 2023. doi: 10.1145/3597638.3614546. URL https://doi.org/10.1145/3597638.3614546. 6.4

[126] J.E. Rawling, E.C. Carson, J.W. Attig, D.M. Mickelson, W.N. Mode, M.D. Johnson, and K.M. Syverson. Quaternary geology of wisconsin. Technical report, Wisconsin Geological and Natural History Survey, 2025. URL https://doi.org/10.54915/xqpw9883. 6.4.1

[127] Blake Ellis Reid. The curb-cut effect and the perils of accessibility without disability. *SSRN Electronic Journal*, 2022. doi: 10.2139/ssrn.4262991. URL https://doi.org/10.2139/ssrn.4262991. 1.1, 5.6

[128] Samuel Reinders, Matthew Butler, Ingrid Zukerman, Bongshin Lee, Lizhen Qu, and Kim Marriott. When refreshable tactile displays meet conversational agents: Investigating accessible data presentation and analysis with touch and speech. *IEEE Transactions on Visualization and Computer Graphics*, 31(1):864–874, January 2025. ISSN 2160-9306. doi: 10.1109/tvcg.2024.3456358. URL https://doi.org/10.1109/tvcg.2024.3456358. 6.3.1

[129] Yvonne Rogers, Jeni Paay, Margot Brereton, Kate L. Vaisutis, Gary Marsden, and Frank Vetere. Never too old. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3913–3922. ACM, April 2014. doi: 10.1145/2556288.2557184. URL https://doi.org/10.1145/2556288.2557184. 5.3.3, 5.4.2.1

[130] Rod D Roscoe, Erin K Chiou, and Abigail R Wooldridge, editors. *Advancing diversity, inclusion, and social justice through human systems engineering*. CRC Press, London, England, December 2019. 1.1

[131] Heather Ruland Staines. Digital open annotation with hypothesis: Supplying the missing capability of the web. *Journal of Scholarly Publishing*, 49(3):345–365, April 2018. ISSN 1710-1166. doi: 10.3138/jsp.49.3.04. URL http://dx.doi.org/10.3138/jsp.49.3.04. 9.3

[132] Manaswi Saha, Michael Saugstad, Hanuma Teja Maddali, Aileen Zeng, Ryan Holland, Steven Bower, Aditya Dash, Sage Chen, Anthony Li, Kotaro Hara, and Jon Froehlich. Project sidewalk: A web-based crowdsourcing tool for collecting sidewalk accessibility data at scale. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–14. ACM, May 2019. doi: 10.1145/3290605.3300292. URL http://dx.doi.org/10.1145/3290605.3300292. 9.3

[133] Antti Salovaara. Inventing new uses for tools: A cognitive foundation for studies on appropriation. *Human Technology: An Interdisciplinary Journal on Humans in ICT Environments*, 4(2):209–228, November 2008. ISSN 1795-6889. doi: 10.17011/ht/urn.200811065856. URL https://doi.org/10.17011/ht/urn.200811065856. 2.1.2

[134] Elizabeth B.-N. Sanders and Pieter Jan Stappers. Probes, toolkits and prototypes: three approaches to making in codesigning. *CoDesign*, 10(1):5–14, January 2014. doi: 10.1080/15710882.2014.888183. URL https://doi.org/10.1080/15710882.2014.888183. 2.1.2

[135] Zhanna Sarsenbayeva, Niels van Berkel, Eduardo Velloso, Jorge Goncalves, and Vassilis Kostakos. Methodological standards in accessibility research on motor impairments: A survey. *ACM Computing Surveys*, 55(7):1–35, July 2023. doi: 10.1145/3543509. URL https://doi.org/10.1145/3543509. 5.3.3

[136] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):341–350, January 2017. doi: 10.1109/tvcg.2016.2599030. URL https://doi.org/10.1109/tvcg.2016.2599030. 1.2, 2.2.1, 5.3.2.2, 5.3.3, 5.4.3.1, 5.5.2.1, 6.2, 6.5.2

[137] Anastasia Schaadhardt, Alexis Hiniker, and Jacob O. Wobbrock. Understanding Blind Screen-Reader Users' Experiences of Digital Artboards. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 5 2021. Association for Computing Machinery. Accessed: 2021-09-06. 2.2.3

[138] D. Schepers and F. Elavsky. "accessible data viz". *The Investigative Reporters and Editors Journal*, 4:8–9, December 2024. 2.2.3, 8.3.1

[139] William Schiff and Emerson Foulke, editors. *Tactual Perception*. Cambridge University Press, Cambridge, England, March 1982. 2.2.3

[140] Donald Schön and John Bennett. Reflective conversation with materials, April 1996. URL https://doi.org/10.1145/229868.230044. 6.8.1

[141] JooYoung Seo, Yilin Xia, Bongshin Lee, Sean Mccurry, and Yu Jun Yam. Maidr: Making statistical visualizations accessible with multimodal data representation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, pages 1–22. ACM, May 2024. doi: 10.1145/3613904.3642730. URL https://doi.org/10.1145/3613904.3642730. 6.3.1

[142] Hanieh Shakeri, Ye Yuan, Benett Axtell, Denise Y. Geiskkovitch, and Carman Neustaedter. Designing smart home technology for passive co-presence over distance. In *Designing Interactive Systems Conference*, DIS '24, page 3389–3406. ACM, July 2024. doi: 10.1145/3643834.3661508. URL http://dx.doi.org/10.1145/3643834.3661508. 9.3

[143] Ather Sharif and Babak Forouraghi. evographs — a jquery plugin to create web accessible graphs. In *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–4. IEEE, January 2018. doi: 10.1109/ccnc.2018.8319239. URL https://doi.org/10.1109/ccnc.2018.8319239. 5.3.1.1, 6.3.2

[144] Ather Sharif, Sanjana Shivani Chintalapati, Jacob O. Wobbrock, and Katharina Reinecke. Understanding screen-reader users' experiences with online data visualizations. In *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '21, pages 1–16. ACM, October 2021. doi: 10.1145/3441852.3471202. URL https://doi.org/10.1145/3441852.3471202. 2.2.3, 5.2, 5.3.1.2, 6.3.1, 6.3.2

[145] Ather Sharif, Olivia H. Wang, Alida T. Muongchan, Katharina Reinecke, and Jacob O. Wobbrock. Voxlens: Making online data visualizations accessible with an interactive javascript plug-in. In *CHI Conference on Human Factors in Computing Systems*, pages 1–19. ACM, April 2022. doi: 10.1145/3491102.3517431. URL https://doi.org/10.1145/3491102.3517431. 5.3.1.1, 6.3.2

[146] Ather Sharif, Joo Gyeong Kim, Jessie Zijia Xu, and Jacob O. Wobbrock. Understanding and reducing the challenges faced by creators of accessible online data visualizations. In *The 26th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '24, pages 1–20. ACM, October 2024. doi: 10.1145/3663548.3675625. URL https://doi.org/10.1145/3663548.3675625. 2.2.3, 3, 6.2, 6.3.2, 6.8.2, 8.3.1

[147] Lei Shi, Idan Zelzer, Catherine Feng, and Shiri Azenkot. Tickers and talker. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 4896–4907, New York, NY, USA, May 2016. ACM. ISBN 978-1-4503-3362-7. doi: 10.1145/2858036.2858507. URL https://doi.org/10.1145/2858036.2858507. Accessed: 2021-09-03. 2.1.2

[148] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *The Craft of Information Visualization*, pages 364–371. Elsevier, 2003. ISBN 9781558609150. doi: 10.1016/b978-155860915-0/50046-9. URL https://doi.org/10.1016/b978-155860915-0/50046-9. 5.5.2

[149] Ben Shneiderman, Christopher Williamson, and Christopher Ahlberg. Dynamic queries. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '92*, CHI '92, pages 669–670. ACM Press, 1992. doi: 10.1145/142750.143082. URL https://doi.org/10.1145/142750.143082. 6.2

[150] Alexandru-Ionut Siean and Radu-Daniel Vatavu. Wearable interactions for users with motor impairments: Systematic review, inventory, and research implications. In *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–15. ACM, October 2021. doi: 10.1145/3441852.3471212. URL https://doi.org/10.1145/3441852.3471212. 5.3.3

[151] V. Sivaraman, F. Elavsky, D. Moritz, and A. Perer. Counterpoint: Orchestrating large-scale custom animated visualizations. In *2024 IEEE Visualization and Visual Analytics (VIS)*, pages 16–20, 2024. 8.3.1

[152] Volker Sorge. Polyfilling accessible chemistry diagrams. In *Lecture Notes in Computer Science*, pages 43–50. Springer International Publishing, 2016. ISBN 9783319412634. doi: 10.1007/978-3-319-41264-1_6. URL https://doi.org/10.1007/978-3-319-41264-1_6. 2.2.3, 5.2, 5.3.1.1, 5.3.2.1, 6.3.1

[153] L. South and M. A. Borkin. Photosensitive accessibility for interactive data visualizations. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–11, 2022. 8.3.1

[154] Laura South and Michelle Borkin. Generating Seizure-Inducing Sequences with Interactive Visualizations. In *Proc. Of the IEEE VIS 2020 posters*, 10 2020. doi: 10.31219/osf.io/85gwy. 2.2.3

[155] Laura South, David Saffo, and Michelle Borkin. Detecting and Defending Against Seizure-Inducing GIFs in Social Media. Technical report, OSF Preprints, 1 2021. Accessed: 2021-08-20. 2.2.3

[156] Katta Spiel, Kathrin Gerling, Cynthia L. Bennett, Emeline Brulé, Rua M. Williams, Jennifer Rode, and Jennifer Mankoff. Nothing about us without us. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, pages 1–8. ACM, April 2020. doi: 10.1145/3334480.3375150. URL https://doi.org/10.1145/3334480.3375150. 1.1, 9.1

[157] Clay Spinuzzi. The methodology of participatory design. *Technical communication*, 52 (2):163–174, 2005. 2.1.1

[158] Sebastian Paul Suggate. Beyond self-report: Measuring visual, auditory, and tactile mental imagery using a mental comparison task. *Behavior Research Methods*, 56(8):8658–8676, September 2024. ISSN 1554-3528. doi: 10.3758/s13428-024-02496-z. URL https://doi.org/10.3758/s13428-024-02496-z. 1.1

[159] Sarit Felicia Anais Szpiro, Shafeka Hashash, Yuhang Zhao, and Shiri Azenkot. How people with low vision access computing devices. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '16, pages 171–180, New York, NY, USA, October 2016. ACM. ISBN 9781450341240. doi: 10.1145/2982142.2982168. URL https://doi.org/10.1145/2982142.2982168. 2.2.3

[160] Pierre Tchounikine. Designing for appropriation: A theoretical account. *Human–Computer Interaction*, 32(4):155–195, July 2017. ISSN 1532-7051. doi: 10.1080/07370024.2016.1203263. URL https://doi.org/10.1080/07370024.2016.1203263. 2.1.2

[161] Yuanyang (YY) Teng and Darren Gergle. Access is not enough: Toward developmental flourishing. In *Proceedings of the Extended Abstracts of the 2026 CHI Conference on Human Factors in Computing Systems*, CHI EA '26, page 1–6. ACM, April 2026. doi: 10.1145/3772363.3798552. URL http://dx.doi.org/10.1145/3772363.3798552. 1.1

[162] John R Thompson, Jesse J Martinez, Alper Sarikaya, Edward Cutrell, and Bongshin Lee. Chart reader: Accessible visualization experiences designed with screen reader users. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–18. ACM, April 2023. doi: 10.1145/3544548.3581186. URL https://doi.org/10.1145/3544548.3581186. 5.2, 5.3.1.1, 5.4.1.1, 5.5.3, 6.3.1, 6.3.2, 6.4

[163] Alexandra To, Angela D. R. Smith, Dilruba Showkat, Adinawa Adjagbodjou, and Christina Harrington. Flourishing in the everyday: Moving beyond damage-centered design in hci for bipoc communities. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference*, DIS '23, pages 917–933. ACM, July 2023. doi: 10.1145/3563657.3596057. URL https://doi.org/10.1145/3563657.3596057. 2.1.1

[164] T. Tran, H.-N. Lee, and J. H. Park. Discovering accessible data visualizations for people with adhd. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–19, 2024. 8.3.1

[165] Visa. Visa Chart Components. https://github.com/visa/visa-chart-components, 2022. Accessed: 2022-12-01. 5.3.2.1

[166] W3C. Web content accessibility guidelines (wcag) 2.1, 2018. 2.2.2.2, 3, 8.5.1

[167] W3C. Webaim: Screen reader user survey #10 results. https://webaim.org/projects/screenreadersurvey10/, 2024. [Online]. 8.7.2.1

[168] WAI. Understanding success criterion 2.4.7: focus-visible. WCAG standard, W3C, 2016. Accessed: 2022-12-11. 5.4.3.2

[169] WAI. Understanding success criterion 4.1.2: name, role, value. WCAG standard, W3C,

2016. Accessed: 2022-03-04. 2.2.2.2, 5.3.1.3

[170] WAI. Accessible rich internet applications (WAI-ARIA 1.1). Technical report, W3C, 2017. Accessed: 2022-12-11. 3

[171] WAI. Understanding success criterion 2.1.1: keyboard. WCAG standard, W3C, 2017. URL https://www.w3.org/WAI/WCAG21/Understanding/keyboard.html. Accessed: 2026-03-20, https://www.w3.org/WAI/WCAG21/Understanding/keyboard.html. 5.4.2.2, 6.3.1

[172] Chris Weaver. Multidimensional visual analysis using cross-filtered views. In *2008 IEEE Symposium on Visual Analytics Science and Technology*, pages 163–170. IEEE, October 2008. doi: 10.1109/vast.2008.4677370. URL https://doi.org/10.1109/vast.2008.4677370. 2.2.1, 3

[173] WebAIM. WAVE, the web accessibility evaluation tool. https://wave.webaim.org/, 2020. Accessed: 2023-01-10. 5.5.1.1

[174] WebAIM. Webaim: The WebAIM Million - An annual accessibility analysis of the top 1,000,000 home pages. https://webaim.org/projects/million/#wcag, 2021. Accessed: 2021-09-03. 9.3

[175] Hadley Wickham. A layered grammar of graphics. *Journal of Computational and Graphical Statistics*, 19(1):3–28, January 2010. ISSN 1537-2715. doi: 10.1198/jcgs.2009.07098. URL https://doi.org/10.1198/jcgs.2009.07098. 6.2

[176] David Gray Widder and Richmond Wong. Thinking upstream: Ethics and policy opportunities in ai supply chains, 2023. URL https://arxiv.org/abs/2303.07529. 1.2

[177] David Gray Widder, Meredith Whittaker, and Sarah Myers West. Why 'open' ai systems are actually closed, and why this matters. *Nature*, 635(8040):827–833, November 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-08141-1. URL https://doi.org/10.1038/s41586-024-08141-1. 1.2

[178] Leland Wilkinson. The grammar of graphics. In *Handbook of Computational Statistics*, pages 375–414. Springer Berlin Heidelberg, December 2012. ISBN 9783642215506. doi: 10.1007/978-3-642-21551-3_13. URL https://doi.org/10.1007/978-3-642-21551-3_13. 6.2

[179] B. L. Wimer, L. South, K. Wu, D. A. Szafir, M. A. Borkin, and R. A. Metoyer. Beyond vision impairments: Redefining the scope of accessible data representations. *IEEE Transactions on Visualization and Computer Graphics*, 30(12):7619–7636, December 2024. 2.2.1, 2.2.3, 8.3.1

[180] Brianna L. Wimer, Ritesh Kanchi, Kaija Frierson, Venkatesh Potluri, Ronald Metoyer, Jennifer Mankoff, Miya Natsuhara, and Matt X. Wang. Nonvisual support for understanding and reasoning about data structures, 2026. URL https://arxiv.org/abs/2601.19168. 6.3.2

[181] Langdon Winner. Do artifacts have politics? In *Computer Ethics*, pages 177–192. Routledge, May 2017. ISBN 9781315259697. doi: 10.4324/9781315259697-21. URL

https://doi.org/10.4324/9781315259697-21. 1.2

[182] J. O. Wobbrock, S. K. Kane, K. Z. Gajos, S. Harada, and J. Froehlich. Ability-based design: Concept, principles and examples. *ACM Transactions on Accessible Computing*, 3(3):1–27, April 2011. doi: 10.1145/1952383.1952384. 1.2, 2.1.1, 8.3.2

[183] Jacob O. Wobbrock, Shaun K. Kane, Krzysztof Z. Gajos, Susumu Harada, and Jon Froehlich. Ability-based design. *ACM Transactions on Accessible Computing*, 3(3):1–27, April 2011. doi: 10.1145/1952383.1952384. URL https://doi.org/10.1145/1952383.1952384. 5.6

[184] Mintesnot Woldeamanuel and Andrew Kent. Measuring walk access to transit in terms of sidewalk availability, quality, and connectivity. *Journal of Urban Planning and Development*, 142(2), June 2016. ISSN 1943-5444. doi: 10.1061/(asce)up.1943-5444.0000296. URL http://dx.doi.org/10.1061/(ASCE)UP.1943-5444.0000296. 9.3

[185] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Transactions on Visualization and Computer Graphics*, 22(1): 649–658, January 2016. ISSN 1077-2626. doi: 10.1109/tvcg.2015.2467191. URL https://doi.org/10.1109/tvcg.2015.2467191. 2.2.1

[186] R. Wood, J. H. Feng, and J. Lazar. Health data visualization literacy skills of young adults with down syndrome and the barriers to inference-making. *ACM Transactions on Accessible Computing*, 17(1):1–1, March 2024. 8.3.1

[187] R. Wood et al. "creatures of habit": influential factors to the adoption of computer personalization and accessibility settings. *Universal Access in the Information Society*, 23(2): 927–953, March 2023. 3, 8.3.3, 8.4.1.3, 8.7.4.3

[188] Keke Wu, Emma Petersen, Tahmina Ahmad, David Burlinson, Shea Tanis, and Danielle Albers Szafir. Understanding Data Accessibility for People with Intellectual and Developmental Disabilities. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, pages 1–16, New York, NY, USA, 5 2021. Association for Computing Machinery. doi: 10.1145/3411764.3445743. Accessed: 2021-09-03. 2.2.3, 8.3.1

[189] F. Yanez, C. Conati, A. Ottley, and C. Nobre. The state of the art in user-adaptive visualizations. *Computer Graphics Forum*, December 2024. 8.3.2

[190] Katherine Ye, Wode Ni, Max Krieger, Dor Ma'ayan, Jenna Wise, Jonathan Aldrich, Joshua Sunshine, and Keenan Crane. Penrose. *ACM Transactions on Graphics*, 39(4), August 2020. doi: 10.1145/3386569.3392375. URL https://doi.org/10.1145/3386569.3392375. 5.5.3.1

[191] Rebecca Yeo and Karen Moore. Including disabled people in poverty reduction work: "nothing about us, without us". *World Development*, 31(3):571–590, March 2003. ISSN 0305-750X. doi: 10.1016/s0305-750x(02)00218-8. URL https://doi.org/10.1016/s0305-750x(02)00218-8. 1.1

[192] Haixia Zhao, Catherine Plaisant, Ben Shneiderman, and Jonathan Lazar. Data sonification

for users with visual impairment. *ACM Transactions on Computer-Human Interaction*, 15 (1):1–28, May 2008. ISSN 1073-0516. doi: 10.1145/1352782.1352786. URL https://doi.org/10.1145/1352782.1352786. 2.2.3

[193] Jonathan Zong. Using real names of disabled participant-contributors to practice citational justice in accessibility. In *2025 IEEE Workshop on Accessible Data Visualization (AccessViz)*, pages 30–33. IEEE, November 2025. doi: 10.1109/accessviz68666.2025.00011. URL https://doi.org/10.1109/accessviz68666.2025.00011. 6.4

[194] Jonathan Zong, Crystal Lee, Alan Lundgard, JiWoong Jang, Daniel Hajas, and Arvind Satyanarayan. Rich screen reader experiences for accessible data visualization. *Computer Graphics Forum*, 41(3):15–27, June 2022. doi: 10.1111/cgf.14519. URL https://doi.org/10.1111/cgf.14519. 2.2.2.2, 3, 5.2, 5.3.1.1, 5.3.1.3, 5.3.2.1, 5.4.1.1, 5.4.3.3, 5.5.1.1, 5.5.2, 5.5.3, 5.6, 6.3.1, 6.3.2, 6.4

[195] Jonathan Zong, Josh Pollock, Dylan Wootton, and Arvind Satyanarayan. Animated vega-lite: Unifying animation with a grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):149–159, January 2023. ISSN 2160-9306. doi: 10.1109/tvcg.2022.3209369. URL https://doi.org/10.1109/tvcg.2022.3209369. 6.2

[196] Jonathan Zong, Isabella Pedraza Pineros, Mengzhu (Katie) Chen, Daniel Hajas, and Arvind Satyanarayan. Umwelt: Accessible structured editing of multi-modal data representations. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, pages 1–20. ACM, May 2024. doi: 10.1145/3613904.3641996. URL https://doi.org/10.1145/3613904.3641996. 6.3.2, 6.8.4