Tool-making as an Intervention on the Accessibility of Interactive Data Experiences

Frank J. Elavsky

July 2026
CMU-HCII-26-103

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

Thesis Committee:
Dominik Moritz, Chair, Carnegie Mellon University HCII
Patrick Carrington, Co-Chair, Carnegie Mellon University HCII
Ken Holstein, Carnegie Mellon University HCII
Jennifer Mankoff, External, University of Washington
Tamara Munzner, External, University of British Columbia

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Copyright © 2026 **Frank J. Elavsky**

This research has been partially or fully funded by the Human-Computer Interaction Institute, Apple Inc, Highsoft, Adobe, and the Chan-Zuckerberg Initiative.

Keywords

Visualization, accessibility, data interaction, tangible interfaces, tactile data representations, sonification, graph theory, human-computer interaction, personalization, disability, malleable interfaces, guidelines, standards, heuristics, evaluation, human-studies, qualitative research, quantitative research, assistive technology, data science, software tools, hardware tools, input modalities, voice-controlled interfaces, gesture-based input, natural language system design, programming tools, sensory substitution, software debugging.

Contents

Keywords ii

Abstract xiii

Acknowledgments xv

I Introduction 1

1 Motivation 3
1.1 What do we mean by "accessibility" and "tools?" . . . 4
1.1.1 Is "accessible visualization" really an oxymoron? . . . 4
1.1.2 On *tools*, *tool-making*, and *human-tool* interaction . . . 6

2 Background & Related Work 9
2.1 Tools and the People Who Use Them . . . 9
2.1.1 Studying Practitioners and Their Work . . . 9
2.1.2 How Tools Use Us . . . 9
2.2 The Conversation Between Accessibility and Data Visualization . . . 11
2.2.1 A Short History Across Modalities . . . 11
2.2.2 The Split Between Research and Practice . . . 12
2.2.3 Empirical Depth, but a Lack of Applied Work in Production . . . 13
2.3 Critical Voices and Work Along the Periphery . . . 14
2.4 Why Navigation, Interaction, and Personalization . . . 15
2.4.1 Navigation . . . 16
2.4.2 Interaction . . . 16
2.4.3 Personalization . . . 16
2.5 What to Take Away . . . 17

3 Overview of Contributions 19

II Evaluation: Helping Practitioners Identify Accessibility Barriers 25

4 *Chartability*: Heuristics as a Tool and Resource 27
4.1 Preface . . . 27

4.1.1 Context . . . 27
4.2 Overview . . . 30
4.3 Existing Work in Data Visualization and Accessibility . . . 31
4.3.1 Research Advancements in Data Visualization and Accessibility . . . 31
4.3.2 Accessibility Practices in Data Visualization Tools and Libraries . . . 32
4.3.3 Accessibility in Practice, Broadly . . . 33
4.3.4 Using Heuristics to Break Into Under-addressed Areas . . . 33
4.4 Making Chartability . . . 34
4.4.1 Compromising . . . 35
4.4.2 Assistive . . . 36
4.4.3 Flexible . . . 36
4.5 Using Chartability . . . 36
4.5.1 Visual Testing (Perceivable) . . . 37
4.5.2 Keyboard Probing (Operable, Assistive) . . . 37
4.5.3 Screen Reader Inspecting (Perceivable, Operable, Robust, Assistive) . . . 38
4.5.4 Checking Cognitive Barriers (Understandable, Compromising) . . . 38
4.5.5 Evaluating Context (Robust, Assistive, Flexible) . . . 41
4.6 Validating Chartability . . . 41
4.6.1 Pre-Validations and Flipped Roles: Participants Question *Me* . . . 42
4.6.2 Discovering "Critical" Heuristics . . . 43
4.6.3 Selecting Participants and Projects . . . 43
4.7 Study Results . . . 44
4.7.1 Real Access has more Considerations than Colorblindness . . . 45
4.7.2 Audits are Slow, but Help me Focus . . . 45
4.7.3 Chartability Helps me Remember and Stay Consistent . . . 46
4.7.4 Access is an Experience, not just Compliance . . . 46
4.7.5 Everyone wants More Evaluation Resources and Tools . . . 46
4.7.6 Experts: "Novices will Struggle." Novices: "This was so helpful" . . . 46
4.7.7 What about Auditors with Disabilities? . . . 47
4.8 Extended Results . . . 47
4.9 Discussion . . . 48
4.10 Conclusion . . . 49
4.11 Postface . . . 50

III Navigation: Making Data Structures Traversable 51

5 *Data Navigator*: Low-level Tooling for Creating Navigable Data Structures 53
5.1 Preface . . . 53
5.1.1 Context . . . 53
5.2 Overview . . . 55
5.3 Related Work . . . 57
5.3.1 Accessibility research and standards in visualization . . . 57
5.3.2 Visualization toolkits and technical work . . . 58

5.3.3 Considering assistive technologies and input devices . . . 60
5.4 Data Navigator: System Design . . . 60
5.4.1 Structure . . . 61
5.4.2 Input . . . 64
5.4.3 Rendering . . . 66
5.5 Case Examples with Data Navigator . . . 69
5.5.1 Augmenting a Static, Raster Visualization . . . 69
5.5.2 Building Data Navigation for a Toolkit Ecosystem . . . 70
5.5.3 Co-designing Novel Data Navigation Prototypes . . . 72
5.6 Limitations and Future Work . . . 76
5.7 Conclusion . . . 76
5.8 Postface . . . 78
5.8.1 The library boundary: what ships, and what demos build . . . 79

6 *Skeleton*: Visual Authoring of Non-visual Data Experiences 85
6.1 Preface . . . 85
6.1.1 Context . . . 85
6.2 Overview . . . 87
6.3 Related Work . . . 89
6.3.1 Visualizing Non-visual Structure . . . 89
6.3.2 Non-visual Data Experiences . . . 89
6.3.3 Authoring Non-visual Data Experiences . . . 90
6.4 Co-design Foundations . . . 91
6.4.1 Geologic Map . . . 92
6.4.2 Design System Library . . . 93
6.4.3 Open Source Visualization Library . . . 93
6.4.4 Infrastructure from Practice . . . 94
6.5 *Skeleton*: System Design . . . 100
6.5.1 Staging: Input and Preparation for Authoring . . . 100
6.5.2 Edit: Interacting with Topology, Layout, and Semantics . . . 101
6.5.3 Test: Debugging Interaction Interactively . . . 103
6.5.4 System Implementation: Technical Details . . . 104
6.6 User Study . . . 106
6.6.1 Participants . . . 108
6.6.2 Procedure . . . 108
6.6.3 Analysis . . . 109
6.7 Results . . . 109
6.7.1 Seeing Navigation Made Structural Problems Legible as Design Problems 109
6.7.2 Practitioners Developed a Designerly Interest in What Constitutes Good Navigation . . . 110
6.7.3 Iteration Was Substantive, Self-directed, and Concentrated on Semantics . 111
6.7.4 Seeing Navigation Prompted Practitioners to Reconsider the Architecture of Their Own Charts . . . 112

6.7.5 Experiencing Keyboard Navigation Surfaced a Broader Range of Users and Input Technologies . . . 112
6.8 Discussion . . . 114
6.8.1 Visibility as a Precondition for Iteration . . . 114
6.8.2 From Compliance to Design . . . 114
6.8.3 Bespoke Visualizations as an Unaddressed Accessibility Research Problem 115
6.8.4 What Visualization Owes Accessibility . . . 115
6.9 Limitations and Future Work . . . 116
6.10 Conclusion . . . 117
6.11 Postface . . . 118

IV Interaction: Exploring New Possibilities for Blind Data Science 119

7 *Cross-perception*: Rethinking Input Design Towards Blind Analytical Interaction 121
7.1 Preface . . . 121
7.1.1 Context . . . 121
7.2 Overview . . . 123
7.3 Related Work . . . 125
7.3.1 Cross-Filtering and Interactive Visual Analysis . . . 125
7.3.2 Non-Visual Data Representations . . . 125
7.3.3 Access-Oriented Data Interaction . . . 126
7.3.4 Tactile and Haptic *Input* . . . 126
7.4 Formalizing Cross-Perception . . . 127
7.4.1 A Design Vignette: Fuzzy Search and Dual-Task Comparison in Naturalistic Blind Reading . . . 127
7.4.2 Design Goals . . . 128
7.5 Prototype: The Cross-Feelter . . . 129
7.5.1 Interaction Flow . . . 129
7.5.2 Hardware Design . . . 130
7.5.3 Analytical Environment . . . 132
7.6 Data Exploration Study . . . 134
7.6.1 Study Design . . . 134
7.6.2 Participants . . . 135
7.6.3 Datasets . . . 135
7.6.4 Device Familiarization: Video Interaction Task . . . 135
7.6.5 Introduction to Tactile Histograms and Cross-Filtering . . . 136
7.6.6 Data Tasks . . . 136
7.6.7 Measures and Analysis . . . 137
7.7 Results . . . 137
7.7.1 Objective Results . . . 138
7.7.2 Subjective Results . . . 139
7.8 Discussion . . . 139
7.8.1 Research Question Responses . . . 140

7.8.2 Accuracy and Precision Considerations . . . 140
7.8.3 The Case for Analytical Interaction as a Primary Research Site . . . 141
7.8.4 Toward Hardware and Physicalization in Accessible Data Interaction . . . 141
7.9 Future Use Cases . . . 142
7.9.1 Cross-Perception Without a Powered Display: The Pre-Rendered Interaction Cube . . . 142
7.9.2 Sequential Data Navigation with Continuous Positional Feedback: Feelter-Only Chart Interaction . . . 143
7.9.3 Sonification Navigation via Physical Rail Position . . . 144
7.9.4 Multi-Dimensional Spatial Interaction: Rails as Input and Output . . . 144
7.10 Limitations . . . 144
7.11 Conclusion . . . 145
7.12 Postface . . . 146

V Personalization: System-building for User Agency 147

8 *Softerware*: Enabling Personalization of Interactive Data Representations for Users with Disabilities 149
8.1 Preface . . . 149
8.1.1 Context . . . 149
8.2 Overview . . . 151
8.3 Related Work . . . 152
8.3.1 Data Visualization and Accessibility . . . 152
8.3.2 Systems that Adapt . . . 153
8.3.3 Personalization and Accessibility . . . 153
8.4 Presenting: *Softerware* . . . 154
8.4.1 Defining *Softerware*'s Principles . . . 154
8.5 Prototype: Visualization *Softerware* . . . 155
8.5.1 *Pretty Accessible* by Default . . . 156
8.5.2 Reasoned Constraints: 195 Accessibility Options for Interactive Data Representations . . . 157
8.5.3 Preferences Menu Design . . . 157
8.5.4 What the Prototype Actually Is . . . 159
8.6 Evaluation . . . 160
8.6.1 Preliminary: Visualization Practitioners . . . 160
8.6.2 Study: Blind and Low Vision Users with Accessibility Expertise . . . 160
8.6.3 Procedure . . . 162
8.7 Results . . . 163
8.7.1 Preliminary Findings . . . 163
8.7.2 Prototype-level Feedback . . . 164
8.7.3 System-class Accessibility Findings . . . 165
8.7.4 User-Centered Findings . . . 166
8.8 Conclusion . . . 167

8.9 Postface 169

VI Conclusion **171**

9 Findings **173**
9.1 Access Barriers Are Produced for Practitioners, Both With and Without Disabilities 173
9.2 Tools Can Reduce the Work Required to Make Things Accessible 174
9.3 Making the Invisible Structurally Visible Changes Practice 174
9.4 Friction From a Tool Can Be Productive 175
9.5 Tool-use and Tool-making Are Generative 176
9.6 The Hardest Barriers Required Intervention at the Substrate 177
9.7 What Tool-making Did Not Do 178
9.7.1 State of the Art does not Necessitate Adoption 178
9.7.2 Tooling does not *Solve* Access Friction 179
9.7.3 Authorship by People with Disabilities Remains Unresolved 179

10 Discussion & Future Work **181**
10.1 What is a "tool?" A reflection on the social and material identity of tools 181
10.2 "Applied" accessibility work and *low research* 182
10.3 Who is responsible for repair? 183

11 Biographical Sketch **187**

A The *Softerware* Option Taxonomy **189**

References **195**

List of Figures

4.1 A low contrast chart (left) compared to a higher contrast version (right). A dropper tool is extracting the fill color of the bar and then a contrast ratio has been calculated. Note that the fill color is the same on both bars, but darker borders have been added to ensure the visualization passes contrast tests. 37
4.2 Keyboard navigation paths on a stacked bar chart. The left shows a serial navigation example, typically just a default of rendering order. The right shows both groups (the stack of bars) and categories (the color/texture shared among bars across stacks) as dimensions to explore laterally or vertically. 38

4.3 A mouse cursor is selecting a bar (left, shown with a thick indication border) in a stacked bar chart to filter a dataset (on the right). A system alert (red box) notifies the user of their interaction result. This selection capability must also be provided for the keyboard interface and the alert must be announced to screen readers. 39

4.4 Charts must have a visually available textual explanation provided that summarizes the outcome. "Client Registration Chart" for "Product X" (left) is inaccessible while "New Product Launch a Success" (right) gives a clear takeaway. . . . 39

4.5 An interactive chart displaying only "Image" as semantic information with no feedback provided on selection. The robust semantics given to a screen reader, "toggle button" (middle) as well instant feedback, "selected" (right) are considered proper semantics for an interactive experience. 40

4.6 A line chart (left) with a single line and an accompanying data table (right). This line chart would not provide enough low-level information about each datapoint without the table provided. A table alone however would also be inaccessible. Providing both can satisfy conflicting accessibility needs for different audiences. . 40

4.7 A scatterplot with many points, where a single point within the chart can be accessed by a screen reader (left). Navigating this data piece by piece is unnecessarily tedious, so an annotation callout is provided to help the reader focus on an outlier cluster (right). The callout is being accessed by a screen reader, which is displaying the annotation's summary as well. 41

4.8 A bar chart with categories (left) shown not conforming to Windows High Contrast White Mode. High contrast mode on Windows requires limiting color palettes, using only black or white for most elements (shown on the right). 42

5.1 Data Navigator provides data visualization libraries and toolkits with accessible data navigation structures, robust input handling, and flexible semantic rendering capabilities. 55

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. 59

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. 62

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. 63

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. . . 64

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**). 64

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. 66

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. 67

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 an 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. 68

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. 71

5.11 Our material preparation process involved taking a reference (**A**), tracing it (**B**), and rendering it on a tactile display (**C**). 73

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. 73

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. 75

5.14 The library boundary in Data Navigator. The npm package (right) provides the graph builders, the traversal engine, the rendering layer, and supporting utilities. Each case example (left) supplies the structure design, the modality adapters that translate input events into navigation rule names, the focus lifecycle, and any coupling to the host visualization. 80

6.1 *Skeleton* being used to build a navigation structure over a line chart. **A**: a dual tree visualization of nodes and edges built for a line chart with 3 lines, spanning over 12 months. **B**: a direct-manipulation canvas that resides over a static png image of a line chart. **C**: a skeleton-like structure of nodes, edges, and group outlines that correspond to the schema shown in (A), overlaid on the visual space of the chart in (B). **D**: controls for rapidly scaffolding the nodes, edges, and their shapes shown in (C), using Vega's visualization engine for automatically generating spatial properties. 87
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. 92
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). 101
6.4 Group label pattern builder, including an array of aggregate summary options, template formatter field, and preview. 103
6.5 Computed layout for marks via Vega's visualization engine plus group outlines (left) and with padding estimation using our non-ML computer vision approach (right) for various types of common visualizations. 107
6.6 Re-creation of P8's moment of realization, placing nodes manually: not every element in their voronoi pie chart *should* be navigable. 113

7.1 Interaction and perception in one space while being able to perceive output in a separate space is the cornerstone of cross-filtering. Our prototype *cross-feelter* (left) can manipulate a visual cross-filter on one visualization (middle) in order to produce output in a separate visualization, as a tactile graphic (right). 123
7.2 A. Our prototype *cross-feelter*. B. USB output (to computer). C. 3D printed rail cover. D. 3D printed knobs that consolidate rails into one. E. 2 motorized linear potentiometers. F. Arduino board. G. 3D printed casing. 129
7.3 Wiring schematic of the cross-feelter. A. Teensy 4.0 microcontroller. B. DRV8833 dual H-bridge motor driver. C. Motorized fader, rail 0. D. Motorized fader, rail 1. E. Host computer, connected over USB. Wire colors distinguish 5V motor power, the 3.3V supply, ground, wiper position lines, touch-sense lines, and motor drive lines. 132
7.4 Our analytical environment. A. a 1 to 1 translation between visual and tactile that represents a 60x40 pixel-to-pin array. B. A tactile version of the filter location (if on the chart being focused). C. Cross-filtering controls, including text inputs for screen-reader only manipulation. D. Refreshable tactile display output. E. Example renderings of the other 2 charts. 133
7.5 Results. Completion rate and accuracy are counts. Task time, data queries (computational), and data queries (spoken) are objective, observational measures. Anxiety, stress, cognitive load, and enjoyment are Likert-scale results from 1 ("very little") to 5 ("quite a lot"). 138

7.6 Speculative future use cases for our *cross-perception* approach, contributed by our participants. 143

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). 151

8.2 Our dashboard design on US Energy Consumption in 2017 with a sankey, bar chart, line chart, and preferences menu on the left. 156

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, Skeleton, Cross-perception, and Softerware, 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 mapped the full landscape of accessibility barriers. *Chartability* helped us to identify and prioritize our three latter domains (navigation, interaction, and personalization) as the areas where the most severe gaps remain in practitioner work. *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 ecosystems.

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.

Acknowledgments

This acknowledgments section is quite long because I must admit how many people have enabled me to do something like this PhD.

Thanks first and foremost goes to **Shelby**, whose infinite patience and eagerness to hear about my work has given me life through every step of this journey. Thanks next goes to our dog **Pizzelle** and cat **Squid**, who each have given me such profound delight, joy, and comfort through this stage of life. I love our little family.

As far as **material circumstances, media, and technologies** go, I'm thankful for my espresso machine, which made some 14,000 drinks over eleven years before it finally caught fire and died. It made four to six drinks every day for Shelby, myself, the pets, neighbors, and guests (Breville should have sponsored me). Leaving behind my barista identity was hard. But as of writing this document, I've now been doing data science longer than I was a barista (ten $>$ nine yrs). I'm thankful for d3js, especially. This gave me an outlet and a way to fall in love with data, visualization, and code. This may seem strange, but I'm also thankful for r/dataIsBeautiful for giving me an outlet to share my work (and make it onto the front page of Reddit once or twice). And of course, (I can hear Sanika and Lisa laughing about this as I write it), but I'm thankful for Twitter, RIP (especially between 2017 and April 13th, 2022 before it was bought and then slowly died). Twitter probably landed me this PhD, but it certainly helped me find my grown-up identity and community of like-minded folks. I'm thankful for the Lord of the Rings, Frieren, One Piece, Disco Elysium, Fallout, and Elden Ring (my comfort media). I'm also, despite its shortcomings, thankful for our home. I will miss it when we move from here. I've learned that moisture is the greatest enemy of a comfortable home. And (again, can hear friends laughing at this), but I'm thankful for Uniqlo's t-shirts, which have been an immense staple for years. I've managed to get significant mileage out of every one of these. Thanks to our cast iron skillet, dutch oven, couch, bidet, grill, bed, good set of knives, and gaming computers. Money *can* buy (a little) happiness.

Next, I want to give thanks to the **DatavizA11y community members** and my foundational family in accessibility and visualization: Larene Le Gassick (just a collaborator at the time but now someone whose friendship is dear to me), Liz Hare, Silvia Canelón, Léonie Watson, Emily Kund, Sarah Fossheim, Ted Gies, Melanie Mazanec, Amanda Makulec, Amy Cesal, Øystein Moseng, Chris DeMartini, Ryan Shugart, Jennifer Zhang, Amber Thomas, and all our friends and members not listed. This group has given me immense support over the years. And special thanks to Doug Schepers of Fizz Studio for sponsoring Chartability and for your gracious encouragement, support, and feedback over the years. It has been strange to meet such a kindred spirit at such a small intersection. I'm immensely thankful for your friendship.

I also want to thank all of the **industry folks** who have been encouraging and supportive of

me, many of you have let me know you're using my work or it has helped you in some way (which has helped me carry on through all of this): Alan Wilson (such a wonderful friend and encouraging collaborator), Tony Fast (thank you so much for your honesty, passion, and truly genius experiments), Amanda Makulec (for always connecting me, giving me opportunities, and encouraging me), Jasmine Mithani, Lindsay Silver, Anna Cook, Sara Elizabeth Temby, Tobias Kauer, Saara Kamppari-Miller, Elle Waters, James Nurthen, James Craig, MJ Jawili, Ben Jones, Alli Torban, Johny Cassidy, Travis White, Kai Chang, Kent Eisenhuth, Fanni Melles, Jim Lane, Julia Wolfe, Taha Hassan, Jen Strickland, Patrick Garvin, Gabrielle Merite, Maxene Graze, Miska Knapek, Vinicius Sueiro, Sharron Rush (and all the Knowbility folks), Michael Jordan (no, not that one), Ian Smith, Nicolas Kruchten, Jim Webber, Sam Aaron, and everyone else at Apple, Google, Visa, Adobe, Highsoft, T-Mobile, Jupyter, Quansight, Celonis, Voilà, Data Literacy, Finn.no, GreenInfo, Feedzai, Canva, Observable, Figures and Figures, and many others not already listed. You know who you are. Thank you so much for your support and for not only using my work, but for letting me know and giving me feedback too. Special shoutout to Eric Bailey for being such a great supporter of my work and fantastic conversation partner.

Immense thanks to the folks who have used Chartability in **governmental and policy** contexts. Chartability was intended to be a gift to everyone, so socially contributing to public good was the ideal. Thanks to the immortal legend (and very early hero of mine) Moritz Stefaner who used Chartability to help the World Health Organization develop their Data Design Language. It is hugely flattering for someone I look up to as a pillar to leverage my work. Big thanks to Maarten Lambrechts for letting me know how central Chartability was to the European Commission's Data Visualization Guide, especially their Accessibility section. Thanks to Francis Gagnon (not only for being such a great steward of our field but also for your support) as well as for getting the Quebec government to use Chartability for their design system. Special shoutout (again) to Melanie Mazanec for getting Chartability connected to the State of New Jersey's Open Data Center and State of California's Office of Innovation. Thanks to Emily Vontsolos, Blake Valenta, and Lauren Jong for our wonderful collaboration while y'all were with the City of San Francisco. And thanks to the anonymous folks at the Congressional Budget Office, NIH Common Fund (RIP), Office of Science and Technology Policy, 18f (RIP), USGS, USAID (RIP), and United States Web Design System (RIP?).

Thanks to all the **academics and researchers** who have supported me over the years. I first want to thank Alberto Cairo. You were an early and foundational inspiration for me. I am so thankful you've been a mentor and friend (here's to many more game nights together). Thanks to Arvind for being like an academic "uncle"-advisor and generous conversation partner. Thanks to Vetria Byrd, whose vision led to the REU that first exposed me to research in data visualization. Thanks to Steve Franconeri for letting me, an absolute gremlin, sit in on your lab meetings at Northwestern. Thanks to Gabriel Hankins for your mentorship. Thanks to Jon Froehlich, Jeff Heer, Anhong Guo, Kim Marriott, Matt Kay, Dan Fan, Alexa Siu, Niklas Elmqvist, Danielle Albers Szafir, Derya Akbaba (a truly splendid human!), Miriah Meyer, Evan Peck, Nam Wook Kim, Yea-Seul Kim, John Thompson, Leona Holloway, Matthew Butler, Jim Smiley, Sam Reinders, Bongshin Lee, Ria Khan, Sandra Bae, Arran Wang, Fumeng Yang, Abhraneel Sarma, JooYoung Seo, Rachel Wood, Shira Abramovich, Meredith Broussard, Liz Jackson, Garreth Tigwell, Jasmine Lu, Aashaka Desai, Rahaf Alharbi, Leya Breanna (Bre!) Baltaxe-Admony, Kathryn E. Ringland, Anastasia Schaadhardt, Lawrence Weru, Ather Sharif, Andreas Stefik, Sile

o'Modhrain, Taliesin Smith, Volker Sorge, Rupesh Vyas, Jonatan Hildén, Joel Chan, Laura Garrison, Krisha Mehta, Emma McDonnell, Jesse Martinez, David Widder, Steve Haroz, Alex Kale, Jaylin Herskovitz, Josh Pollock, Sanchita Kamath, Elsie Lee-Robbins, Cindy Xiong Bearfield, Bill Limpisathian, Vanessa Knoppke-Wetzel, Sehi L'Yi, Thomas C. Smits, Isabella Pedraza Pineros, Matt Brehmer, Nur Yildirim, Fred Hohman, Donghao Ren, Mary Beth Kery, James Craig, Leah Findlater, Jackie Milhans, Aaron Geller, Shane Larson, Amy Ko, Torsten Möller, Mols Sauter, Stephanie Valencia Valencia, Péter Gyarmati, Attila Batorfy, Daniel Hajas, Catherine Mei, Lauren Klein, Catherine D'Ignazio, Jess Rauchberg, Jason Wu, Alice Wong, Bess Williamson, Jacqueline Wernimont, Yotam Sechayk, Jessica Hullman, and Richard Ladner.

I want to give special thanks to Pramod Chundury and Naimul Hoque for assembling Keke Wu, Laura South, Brianna Wimer, Stacy Hsueh, and me into our little workshop community. I consider all of you like my **academic siblings** at this little intersection of visualization and accessibility research. I'm so thankful we are part of a generation of researchers together.

I am hugely thankful for the **inspiring practitioners** who helped me fall in love with our craft: Federica Fragapane (I *do* have a favorite!), Shirley Wu, Giorgia Lupi, Valentina D'Efilippo, Nadieh Bremer, Tiziana Alocci, Stefanie Posavec, Sonja Kuipers, Kim Albrecht, Matt Daniels, Jan Diehm, Jennifer Christiansen, Nicholas Rougeux, and Mike Bostock to name a few.

Thanks to my **CMU colleagues** who have either been supportive of my work or inspired me to be better in some way: Lea Albaugh, Scott Hudson, John Stamper, JZ, Sarah Fox, Hiro Shirado, Haiyi Zhu, Jeffrey Bigham, Andy Begel, Pranav Khadpe, Luke Guerdan, Ningjing Tang, Hank Lee, Hyunsung Cho, Seyun Kim, Frederic Gmeiner, Shivani Kapania, Kimi Wenzel, Yi Fei Cheng, Yi-Hao Peng, and Wesley Deng. Immense thanks to Catarina Fidalgo for being encouraging to me about my comm talk (before we discovered I was one of only two who failed!).

Thanks to my **advisors** Dominik and Patrick and to my **committee members**, Jen Mankoff, Tamara Munzner, and Ken Holstein. Dominik, your advice that a PhD "should feel a bit selfish if you're doing it right" (I'm paraphrasing) was perfect. Thanks for letting me pretty much do whatever I wanted and opening doors for me, while also keeping me on track. Patrick, more than anyone you've helped me to chill out the most; I am thankful for your strength. Thank you for creating the AXLE lab and giving us little weirdos and dorks a home. Jen, your work has shifted and re-shifted my thinking many times and I feel like I got away with some kind of trick having you on my committee. You're also just a wonderful person and set an example as the sort of advisor I think I'd like to be someday. Tamara, I've told you this already but I want it on this record (for official reasons), but of all the credit for why I got into data visualization, the most goes to you and your book, which I picked up hot off the press in 2015. That really started everything. And Ken, I don't think there is a professor at CMU who has more consistently put up with my nonsense than you. I love your curiosity and willingness to let me really wax philosophical in class discussions. I hope to grow up to be a teacher like you someday.

Thanks to ***the holy trinity*** from the MIT Visualization Group: Alan Lundgard, Crystal Lee, and Jonathan Zong. All three of you really inspired me to get into visualization and accessibility research but also helped me feel like I was welcome in this space too. Crystal, you have been so generous and supportive over the years, thank you for that. And Jonathan, I swear I just think you're the coolest person ever. I've enjoyed every single chat we've had over the years and cannot thank you enough for really helping me to find my voice and vision time and time again. I have no idea how Arvind managed to assemble a lab with all of your skill and talent in a single

place, but you all really did incredible things together.

Thanks to all of my **collaborators and co-authors** on the projects, papers, and articles we have had together: Venkat Sivaraman, Yunzhi Li, Joon Jang, Cindy Xiong-Bearfield, Lucas Nadolskis, Cindy Bennett, Øystein Moseng, Marita Vindedal, Ted Gies, Jess Hammer, Erik Harpstead, Noor Hammad, Jessie Chen, Seyoung Lee, Lilian de Greef (special thanks for inviting us to be part of your wonderful D&D campaign, too), Tania Allard, Pavithra Eswaramoorthy, Gabriel Fouasnon, Mateusz Paprocki, Emily Barone, Mark Lee (and the whole community at LAMP that you've let me continue to hang with), and Caroline Rose. Some of you had to put up with me more than others. I'm immensely thankful for all of you being so patient.

Thanks to the **financial enablers** who opened doors for collaborations (Dominik aside, of course, since he has been the main patron of my work these past five years): Jørgen Tistel for being so kind and generous with our Highsoft collaboration (and also being a wonderful host during our time there in Vik), Alan Wilson (for all of your legwork over at Adobe, both building bridges slowly and quickly), Tania Allard (for making our stupendous collaboration with Bokeh possible), Jonathan Schwabish (for putting that report together for the Urban Institute and basically letting me invite a bunch of my friends to get paid to be co-authors), Alberto Cairo, Julia Wolfe, Caroline Rose, and other folks who I can't name (but you know who you are). I would love to work with all of you again. Thanks for making things happen.

Thanks to my **labmates** (at DIG and AXLE) not mentioned elsewhere: Adam, Will, Katelyn, Alex, Franklin, Arpit, Atieh, Catalina, Esther, Vijay, Pranav, Scott, James, Daesung, Seokweon, Donny, and Eric. You're all brilliant people. I cannot believe I was able to occupy the same space as all of you. You're very accomplished and will surely go on to do great things. Thanks for your feedback, encouragement, engagement, ideas, and laughs over the years. Thanks also to my favorite **undergrad mentees and friends** for putting up with me: Iman, Chieri, Jillian, Areen, Julia, Ellie, and Ihita. I can't wait to see your adventures unfold.

Thanks to the **foundational people** in my life who pulled (or pushed) me out of terrible places or were otherwise there for me when I really needed it. Thanks, mom. I wouldn't love writing and teaching, if not for you. Thanks to Craig and Beth, for being like parents to me in my younger years and inspiration to Shelby and me in our more-grown-up years. I wouldn't be the person I am today without you two. Thanks to David Montgomery for your insistence in my value as a person. Thanks to the Powells for being a home away from home many times while I grew up. There were countless nights I needed a bed and food and you didn't think twice to take me in. Thanks to Jeff D'Angelo for firing me from your pizza shop and then to Gus Gosanko and Liz Jenkins for telling me I should apply to college. Thanks to Liz for introducing me to Jeff Mallinson and to Mallinson for giving me that huge scholarship (I still have no idea why you did it). The Dean's Circle changed my entire life. Thanks to Stacie too, of course. We love both of you very much. You two showed me how cool it is to be a "sexy murder poet" style of philosopher. Thanks to Annemarie Russell for exposing me to queer and feminist thought and for putting up with my nonsense in the Dean's Circle. Thanks to Erik Samuelson for not only exposing me to community organizing but giving me multiple funded opportunities to learn and put organizing into action. Those experiences radicalized me and fully established my sense of justice and political identity to this day, beyond any doubt. Thanks to Michael DeLashmutt for your mentorship and being the one who enabled me to get into computer stuff, which ultimately sent me on this trajectory to a PhD. I am so thankful for your wit, mentorship, and perspective.

Thanks to the **teachers** in my life, Mr. Donaldson (for being the first person, perhaps in my life, to insist that I was worth something), Mr. Buchanan (teaching me that living the dream is possible), Mrs. Ricksen (helping me realize the importance of my influence over others), Mr. Dellutri (for teaching me to love art, which has carried me through life), Mrs. Strohschein (for loving my fiction - someday I'll send you a copy to sign like you asked), and Mr. Burnett (for giving me a love of theater and performing, and also the first class that Shelby and I shared together). Thanks to Mario Guimaraes, Bruce Grigsby, Betsi Little, Yeheil Curry, Patrick Reyes, Callid Keefe-Perry, Marty Folsom, Liz Colver, Andrea Ide, Andrea Paul, Ryan Torma, Tucker Fitzgerald, David Schulz, and Dave Ellingson for exposing me to entirely new ways to think about the world and understand myself in it.

This final section is the hardest to write briefly, but I want to thank **my friends and fam** who have, in some way shape or form, made this PhD possible. First: I don't want people to think that if you're not here, you're *not* my friend! This section is more reserved for folks who aren't part of a previous category. Most folks already listed are friends. So to our friends from the bay, thank you: Jaime and David (I miss you both so much), Kevin, Tica, Layla, Ross, Sandy+Chris+Lilly, Kat+Phil+Phin (hoo boy does Pizzelle miss Phin and Lilly!). Thanks to our guildmates (I am so proud to we set world records and had so many wonderful adventures): Matt and Kate, Karen, Resa, Ham, Ang, Grun and Cel, Leru, Tabby, Bael, Eliabell, Daniel, and Kata. Thanks, as always, to my best man Robert. I am so proud of your tabletop RPG and how far it has come. Thanks to Erickson, my longest-running friend. Thanks to Kirk, Marlin, CJ, Kelly, and Joe D. Thanks to Oscar and Chancey Fleet, Varsha, Jenni, Andy, David S, Katie M, and Rosemary. Thanks to family: Lea, Leah, Sarah, Jeannie, Kate, Rose, Dave and Linda, and Rick and Sari. Thanks to my friends from our days in Everett who have stayed in touch over all these years: Joe M. and Tess, Tucker and Hannah, Michael and Brianna, Matt W, Sammy "White Ice," Stephen, Mallayana, Sarah, Rachael, Dwyer, Shannon, Judith, Alisha, Bree, Jeff F, Michelle, Tuugi, Doug, Drew, Josef, Curry, Derek, Esca, Amy, Colter, Libi, Dana, Misael, and all of you little gems in the fantasy league (Pete, James, Phil, Cliffy, and Zach). Thanks to Damian and Larene (overseas besties!). Thanks to Kirsty and Simon, Katja, Jim, Charles, Roy, Anouk, Kat, Aaron, Brunty, Kevlin, and Sarah. I am also immensely thankful for folks who have been good friends during our latest season of life: Cella and Tucker (I love you two so, so much), Grant Stoner (the legend!!), Franky, Naomie, Alicia, Jordan, Fatima, Noor, Aimen, Venky, Erica, Lisa, Hwei-Shin, Joon, Sanika, Ezra, Lukas, Anna K., Anna F., Christine, Jessie, Alice, Kaycee, Adinawa, Peya, Faria, Sireesh, Abina, Rua, Katta, and Ruben and Em. Thanks to our neighbors for all your help and friendship: the Dugans, Manchowskis, and Honaths.

I want to end with a few **un-acknowledgements** too: No thanks to my (literal) disabled ass, which made all of this harder. The struggle did not make me stronger or give me an inspiring story. No thanks to gen AI and LLMs. Even if these models were ethical, I'd still be a luddite. I write because I have things I want to say and thoughts I want to feel form (in the same way I dance because I, selfishly, want to feel my own body). Why give the gift of my own living to a machine? GenAI is to self-actualization what frozen microwave meals are to culinary arts.

And despite the noble-sounding reasons for my research, ultimately I did it for myself because I wanted an adventure. And I'm so glad I have proved my haters wrong in the end: I had a wonderful time and it was worth it. I have been so supported, challenged, loved, and cared for in my life by so many people. This compels me to live well. I won't waste your gifts. Thank you.

Part I

Introduction

Chapter 1
Motivation

Technology grows, develops, and adapts alongside humanity. Sometimes, we adapt or appropriate existing technologies to our needs. And in the last two decades, "data" (as a social and cultural vehicle for transferring information through computers) has become a way for everyday people to gain a greater understanding of the world they participate in, even *deepening* their participation.

Data is used to help people make better personal financial decisions, to learn about their health, to learn about the health of societies, to pick who to vote for, to understand trends, wars, and nearly every major issue of daily life. The magnitude to which data now participate in, or are fundamental as a substrate, for a person's health, happiness, and agency cannot be understated.

Data, and access to it, is paramount. Knowing how to read data, read visualizations, interact with data systems, manipulate data interfaces; this unlocks many opportunities for people, such as gaining job opportunities, having more confidence, and communicating more effectively.

This thesis aims to address a broader injustice that has arisen alongside data as an emerging source of agency in people's everyday lives: that some people, namely people with disabilities, often face barriers to accessing data, data interfaces, and data visualizations in particular.

People with a broad range of visual disabilities face the most pervasive forms of exclusion when interacting with data, because *visualizations* are the most common form of data representation. Visual disabilities include people who are "blind" (which may be congenital, progressive, total, partial, "legally" as a designation, and so on; a whole spectrum), people who are "low vision" (which may include forms of night blindness, light sensitivity, visual obstruction, limited field of view, general lack of acuity, color vision deficiency, and so on; another whole spectrum), as well as those who might have visual-cognitive disabilities, such as epilepsy, vestibular vision disorders, and more. All of these people may experience partial or total exclusion from data visualizations and interactive data interfaces and experiences. Some may have access *technically speaking*, but the access is arduous, painful, and nearly unusable in practice.

And, as we unpack later in Chapter 4, many other people with disabilities other than those related to vision also experience barriers when interacting with data.

To no fault of these people, they are not given equal, equitable, or just agency to use data in their lives. We refer to this broader phenomenon as "inaccessibility."

"Inaccessibility" is when a person or system creates something that contains a barrier that excludes a person with a disability. And if a clever reader carefully inspects the previous sentence, we can begin to understand who is first responsible for making something accessible (hint: *not* the person who has been excluded!). Upon recognizing that the creators are first responsible, this then helps us identify which new challenges and problems arise that we must address in order to make accessibility a reality.

In the following section of this thesis' motivation, we add more clarification (as a necessary pre-requisite) for better understanding not just what we are doing in this thesis, but how we frame our work and situate the "researcher" as an actor who has power to enact change towards more accessible data experiences for people with disabilities.

1.1 What do we mean by "accessibility" and "tools?"

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. Our 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.

1.1.1 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 our work. This can be misleading. We 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" [84]. 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 [37, 101, 278]. 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, Frank, have 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, 26% of people self-report living with at least one disability that affects their daily lives [213] and all of us will eventually age into disability (if we are lucky to live a long life).

People with disabilities deserve to participate fully in life. They deserve financial independence. They also deserve loving care and interdependence [18, 138, 200]. 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 [82]. 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 [287]. 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 [138]? Ultimately, our aim is not accessible "visualizations" but equitable and accessible *outcomes*, everything an interactive data experience *accomplishes* for the people who use it.

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 of disability [190]: 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 and 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 [190, 226, 230, 275, 328]. 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.

1.1.2 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 [319] and are sometimes, for this reason, regulated or made proprietary and controlled by powerful entities [107, 312].

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 people are *able* to do [320] and also rules and guardrails about what anyone downstream from that tool's design *should* do [107, 311]. 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 [91].

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 [241], others are lower level but much more expressive [27]. 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 [123].

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

This chapter sets up the story that the rest of this thesis engages. It makes three moves. First, we ground the idea that tools are not neutral instruments: tools shape the work and the workers who use them, which is why tool-making is a meaningful site for accessibility intervention. Second, we survey the long conversation between accessibility and data visualization and its history across modalities, including engaging critical voices that intersect with visualization and accessibility work which inform, support, and hold in tension the work of this thesis. Lastly, we argue that the hardest remaining challenges cluster in three domains, navigation, interaction, and personalization, and we signpost which chapters engage each of these.

2.1 Tools and the People Who Use Them

2.1.1 Studying Practitioners and Their Work

Human-computer interaction has a long tradition of studying the people who make things. Ethnographic and case-study methods immerse researchers in maker, DIY, and assistive technology communities to understand how practitioners actually work, especially when the work is new and unfamiliar to them [137, 139, 140]. Participatory design and co-creation bring practitioners and end users directly into the research process, surfacing how designers shift their thinking when they encounter novel problems [113, 237, 276]. And a growing body of design-based research aims not merely to fill gaps in existing practice but to extend what practitioners are capable of: scaffolding experimentation, supporting learning, and amplifying creative and technical expression [51, 111, 150, 156, 290].

This thesis sits inside that tradition but takes a particular methodological stance: we study practitioners *through their tools*. Each project in this thesis builds a tool, places it in front of working practitioners, and observes what the tool reveals about why accessibility work is hard for them. Sometimes the tool is a resource practitioners apply in their real environments, sometimes a system they build with, and sometimes a probe whose main purpose is to make an invisible part of their work observable. Tool-making here is both an intervention and an instrument of inquiry. That stance only makes sense, however, if tools really do shape work in a deep way. So before surveying the accessibility and visualization literature, we want to engage that claim seriously.

2.1.2 How Tools Use Us

We tend to talk about using tools. We talk much less about how tools use us. A chair is a tool for sitting, but a lifetime of chairs trains a body into the chair's posture. A pickaxe is a tool for extracting ore, but the mine organizes the miner: their hours, their movements, their risks, and what counts as a day's work. Tools come with a direction already built in, and the people who take them up inherit that direction. As a participant in Hohman et al.'s work on model compression in practice remarked, "Better tools make it easy to do correct things and harder

to do incorrect things" [134]. Tools are made precisely with the intention to shape us and our behavior.

Sara Ahmed gives this intuition theoretical teeth. In her phenomenology of orientation, objects are not passive things waiting to be used: they direct bodies, making some actions feel near and natural while other actions fall out of reach, and the social world is full of what she calls straightening devices, arrangements that keep bodies aligned with the paths already laid down for them [2]. In her later work she traces how *use* itself is formative: things become shaped by their use, "fit" for some users and not others, and a history of use leaves a trace that directs how a thing will be used next [3]. Bowker and Star make a parallel argument at the scale of infrastructure: the classification systems and standards embedded in our information systems are largely invisible scaffolding, and every standard valorizes some point of view while silencing another [29]. Assumptions about ability are among the assumptions most commonly baked in: tools encode what their makers believe people are able to do [320].

Visualization research has begun to examine its own tools in these terms. McNutt's survey of 57 JSON-style domain-specific languages for visualization shows that each grammar embodies contested decisions about who its user is and what that user may express, with recurring tensions between formal and colloquial specification and between competing user types [197]. His dissertation develops the larger argument: the design of visualization languages, and of the tools that surround them, directly shapes and can either enhance or constrain end-user agency, motivating systems that offer assistance while respecting the user's intent [198]. In other words, the field's own instruments are straightening devices. Every charting grammar that assumes a sighted reader, every renderer that emits unstructured pixels, quietly directs practitioners away from accessible outcomes, not by forbidding them but by making them feel out of reach.

Tool users are not powerless in this relationship. A rich literature on appropriation documents how people adapt tools beyond their designers' intent, and how that emergent use can be a source of design knowledge in its own right [65, 68, 235, 285], with some work building general theory from the study of emergent and generative tool use [16, 20]. HCI's systems community has likewise developed methods for building and studying tools deliberately: piggybacking on existing systems to study an improvement in context [108, 126], and contributing toolkits whose value is argued through demonstration, usage, technical performance, or heuristics. Ledo et al., analyzing 68 toolkit papers, found these four evaluation strategies dominate, with demonstration by far the most common [169]. That finding will matter later in this chapter: a field can produce many demonstrated toolkits while producing very little maintained infrastructure.

The takeaway from this section is the premise the whole thesis rests on: tools direct work. If the tools of data visualization currently straighten practitioners toward inaccessible outcomes, then re-orienting those tools is not a cosmetic fix. It is an intervention at the place where practice is formed. The rest of this chapter examines what the field has learned about accessible data experiences, and why that learning has so far failed to re-orient the tools.

2.2 The Conversation Between Accessibility and Data Visualization

Accessibility and data representation have been in conversation far longer than the modern visualization community has existed, and that conversation has two important threads. The first is a history of techniques: a steady accumulation of ways to carry data to people through senses and devices other than a sighted eye on a screen. The second is a critical lineage, largely led by disabled scholars and disability studies, that questions how that technical work is framed, who leads it, and who it serves. This thesis tries to take both seriously, so we survey them in turn.

2.2.1 A Short History Across Modalities

Tactile representation is the oldest thread. Tactile maps and graphics for blind readers date to at least the 1830s [112], and tactile sensory substitution for charts has been studied in the computational sciences since the early 1980s [247]. Practice in this modality is unusually mature: tactile and braille standards are robust and well-maintained [14], although their assumptions (embossed paper, fixed layouts) transfer only partially to digital, interactive contexts. Research has worked on that seam, for example by augmenting tactile graphics with audio annotations [12], and industry accessibility teams have published candid experience reports on what it takes to produce tactile graphics inside a real design workflow [59]. Recent work continues to refine the perceptual foundations of non-visual structure [199].

Sonification has a similarly long arc, and because it is a substantial and active research community in its own right, it deserves explicit treatment here. Mansur et al.'s sound graphs demonstrated in 1985 that line data mapped to pitch over time could communicate trends to blind listeners [187]. Subsequent decades built out the empirical base: cross-modal comparisons of auditory and visual graphs [87], non-visual presentation of structured information [33], techniques for scanning and comparing multiple data series by ear [196], and interactive sonification of geo-referenced data [330]. The community consolidated its methods and theory in the Sonification Handbook [125]; Walker and Nees's theory chapter formalizes sonification as a mapping problem from data dimensions to auditory dimensions, emphasizes context cues that play the role axis ticks play in vision, and stresses that interaction, not just playback, shapes what listeners can extract [305]. Sonification research has also been a site of methodological progress, including co-design with blind and low vision collaborators [57].

What is most relevant for this thesis is the recent trajectory: sonification is increasingly integrated into multimodal, interactive systems rather than offered as a standalone substitution. VoxLens adds summaries, sonification, and voice queries to web visualizations through a JavaScript plug-in, and in a study with 21 screen reader users improved information-extraction accuracy by 122% and interaction time by 36% over conventional interaction [257]. Chart Reader, co-designed with screen reader users, combines structured navigation with sonification and author-configurable data insights [289]. Auditory maps have reached usable maturity for wayfinding and choropleth reading [22]. Notably, sonification is also one of the few accessible-visualization techniques to cross into production: Apple shipped Audio Graphs in 2021 as a VoiceOver feature with a public developer API [10]. And Umwelt re-frames the relationship en-

tirely, treating sonification, textual structure, and visualization as coequal representations derived from a shared abstract model in an accessible authoring environment [334]. Sonification's trajectory, from output substitution toward interactive structure and authoring, previews the argument this chapter is building.

Textual description is the most widely deployed modality, and research has moved it well past "write some alt text." Lundgard and Satyanarayan's model of semantic content distinguishes the levels of meaning a description can carry, and shows that blind and sighted readers value different levels [177]. Jung et al. offer evidence-based guidance on description length, plain language, and the ordering of information [154]. Kim et al. collected the questions screen reader users actually ask of visualizations, pointing toward more dialogic, query-driven access [160], and Chundury et al. deepen our understanding of how sensory substitution choices should be grounded in users' practices [50].

Navigation and screen reader interaction is the youngest thread and the one closest to this thesis. Foundational work made statistical and mathematical content traversable by screen reader [102, 271]. Studies of screen reader users' experiences with two-dimensional digital content, including visualizations, documented both the depth of exclusion and the strategies users develop anyway [243, 256]. Zong et al. then gave the area its clearest design vocabulary: structure (how a visualization's entities are organized for traversal), navigation (the operations that move through that structure), and description (what is spoken at each position), showing through co-design that screen reader users want rich, structured exploration rather than a single summary [332]. Olli operationalized part of this insight as an open-source adapter that renders structured traversal for existing charting libraries [25]. A small number of production systems, notably Highcharts, SAS Graphics Accelerator, and Visa Chart Components, have shipped meaningful accessibility functionality [128, 240, 295]; they remain exceptions, and Chapter 4 examines that practitioner tooling landscape in detail.

Across all four modalities the same pattern recurs: each began with converting visual output into another channel, and each has progressively rediscovered that access is not a conversion problem but an interaction problem, turning toward structure, navigation, interactivity, and user control. The research itself has shifted accordingly: where early decades produced perception studies and one-off substitution systems, the last several years have produced co-designed interactive systems, design dimensions, and authoring environments. That convergence is part of why we argue, later in this chapter, that the remaining hard problems are not modality problems at all.

2.2.2 The Split Between Research and Practice

Before turning to the critical literature, it is worth naming that research and practice participate in accessibility at very different speeds, through very different methods, and with very different artifacts.

Accessibility *practice* organizes itself around standards. The Web Content Accessibility Guidelines carry legal and policy weight across much of the world [296], and a profession of accessibility specialists exists largely to translate such guidelines into concrete implementations. One example of this work would be ensuring that interactive elements carry functional semantics that assistive technologies can interpret, which is a foundational, simple, and necessary requirement for accessibility [300].

Standards are powerful precisely because they are consensus artifacts, or *politically mediated documents*, and for the same reason they are inherently reactive: developing, vetting, and formalizing them takes years, so they chronically trail the technologies they govern and are subject to intervention by individuals and organizations. Interactive data visualization sits at a painful point in that lag. Charts are semantically dense, interaction-heavy, and rendered through technologies the guidelines were not written for, so a practitioner who consults the standards in good faith finds little that speaks to their actual artifact. Robust evaluation in practice therefore leans on expert judgment and, in the best cases, direct involvement of people with disabilities, both of which are scarce resources [222].

Technical accessibility and assistive technology *research* meanwhile (outside of the previously mentioned work on modalities and our empirical record), moves quickly and publishes minimally-viable prototypes. These prototypes are an artifact practice largely cannot consume because it requires technical translation and methodological generalization from the small scope of an artifact into a system-wide implementation [86]. A practitioner cannot install a paper; a paper must be read and the artifact carefully dissected before being reconstructed or replicated. The result is a conversation in which researchers accumulate evidence and demonstrations on one side, practitioners wait on standards that lag by a decade on the other, and very little crosses between them. This split is not unique to visualization, but visualization makes it unusually visible, and closing it from the tooling side is the project of this thesis.

2.2.3 Empirical Depth, but a Lack of Applied Work in Production

What exists, first, is empirical breadth. Surveys and meta-analyses have mapped the research space and its biases: accessibility research broadly concentrates on blindness and on computer output [181], and visualization accessibility research mirrors that concentration, with vision-related work dominating while motor, cognitive, neurological, and vestibular access remain nearly unstudied [161, 191, 317]. Even within the well-studied territory the breadth has a shape: blind and low vision people are routinely researched as one population despite using different assistive technologies and different interaction practices [284], and the prototypes the field celebrates are overwhelmingly built for screen reader interaction in particular.

Second, the field has produced exemplary prototypes. The systems surveyed above, such as VoxLens, Chart Reader, Olli, and Umwelt, are rigorous, validated, and often co-designed with disabled collaborators.

What barely exists is the other half of the pipeline: practitioner tools in production environments, and practitioner action at scale. Studies of deployed visualizations and of the people who build them find that accessible outcomes remain rare and that practitioners struggle even when motivated, gathering fragmented guidance themselves with little way to judge its quality [153, 258]. Mainstream visualization tooling has invested its enormous momentum elsewhere, in expressiveness, scale, and interactive speed [27, 123, 241], with accessibility absent from core abstractions and left to individual developers, libraries, or projects. Research prototypes, meanwhile, rarely become infrastructure: toolkit research is evaluated primarily by demonstration [169], and most demonstrated systems are archived with their papers.

This asymmetry is the load-bearing observation of this chapter: **the field has a breadth of empirical work, yet it lacks practitioner tools in production environments and practitioner**

action. A practitioner today inherits strong evidence about what good access looks like but material change remains far behind. The natural next question is to investigate why this is the case, and whether and how tool-making can play a role in practitioner action.

2.3 Critical Voices and Work Along the Periphery

The second register of the conversation is critical, and although it is sometimes treated as peripheral to systems research, it sets the terms under which systems research should be read. We treat it here as load-bearing. This section will be brief, but essential.

The disability rights movement crystallized its central demand in the principle "nothing about us without us:" people with disabilities must hold power in the decisions and designs that concern them [44]. HCI and accessibility research have been working through what that demand means for our field's methods, authorship, and self-conception [133, 275, 279, 280], including calls for a crip HCI and crip visualization that engages disability on disabled people's own terms [138, 315] and concrete practices such as crediting disabled participant-contributors by name [331].

Several concepts from this literature directly shape this thesis. Mingus names *access intimacy*, the felt sense of someone genuinely getting your access needs, to insist that access is relational rather than merely technical and logistical [200]; Bennett et al. build on that disability justice lineage to propose *interdependence* as a frame for assistive technology research, countering the field's fixation on independence [18]. Hamraie and Fritsch's crip technoscience manifesto insists that disabled people are experts and designers of everyday life, and re-frames access not as a finished state but as friction, something contested and negotiated [118]. Hamraie's history of universal design develops *access-knowledge*, showing that what counts as access has always been produced through political struggle rather than neutral technical progress [115, 116]. Bennett et al.'s biographical prototypes push against savior narratives in design by recognizing the prototyping disabled people already do in their own lives [19]. Shew names the broader cultural failure *technoableism*: the belief that technology will "fix" disabled people, held mostly by people who never asked disabled people what they want [260]. And within visualization specifically, Hsueh et al. bring crip technoscience to bear on data experiences, articulating how *access friction* arises when one design that includes some people excludes others [138].

We read this literature first as a set of constraints on what this thesis may claim. It is why we reject techno-solutionist framing outright: no tool in this thesis "solves" accessibility, because access is negotiated among people, institutions, and policy, and tools do not meaningfully redistribute power entirely on their own. It is also why access friction appears throughout this thesis as a design reality rather than an edge case, and why personalization is treated in Chapter 8 as an ethical question and not merely an intellectual problem to solve. And this body of work imposes a boundary we want to name early: a thesis about practitioner tooling mostly serves practitioners building *for* disabled users. The later chapters, Cross-perception and Softerware, begin to shift who holds the power to shape our interfaces. The critical literature is the measure of how far there is still to go.

We also read this literature as relatively divorced from accessibility work in practice. Critical scholarship has even been critical of "access" as a framing and its shortcomings [115, 116, 117, 222, 315]. Yet, practitioners are building systems all over the world that must be made accessible.

And rather than argue that people with disabilities should not be included in the design and development of new projects, instead we argue that, at least in some cases, the empirical record of what people with disabilities want and need is already enough imperative for builders and designers to begin working. For something new and unexplored? Yes, people with disabilities should be included. But what if we have a sufficient breadth and depth of guidance from people with disabilities already? Aren't we then ready to take action and begin practical application? *Chartability* in Chapter 4 takes this stance: people with disabilities have already said enough for us to build out comprehensive guidelines and an evaluation framework. Between standards, research, and community practice, we have an authoritative set of criteria we can test against. This should be our starting point; the bare minimum. And *Data Navigator* in Chapter 5 didn't seek to build a novel interaction but rather build a better system than what currently exists, following existing empirical work that laid the foundation for rich navigation as a necessity [332]. And in Chapter 5 we *do* demonstrate that our system enables novel interaction designs and how practitioners can approach co-design in a simple, effective, and thoughtful way with partners with disabilities. Chapter 6 sought to investigate the barriers that sighted practitioners (without disabilities) face, which is outside of the scope of critical disability literature, and Chapters 7 and 8 then centered on the experiences of blind and low vision people, and how they might have interactive and innovative power over their experiences.

2.4 Why Navigation, Interaction, and Personalization

If the gap were uniform, any contribution would do. It is not uniform. Looking across the empirical record, the prototypes, and the practitioner studies, the places where practitioners most consistently lack the means to act cluster in three domains, and each domain fails for a different kind of reason.

It is worth being explicit about why these three and not others, because the field's other major needs are of a different kind. Description is the best-served domain: practitioners have models, guidance, and concrete heuristics for what to write [154, 177], and what remains is largely a matter of doing it. Evaluation is hard, but it is hard as a knowledge problem, a matter of gathering, judging, and applying scattered guidance, and a well-built resource can carry a practitioner a long way; Chapter 4 contributes exactly such a resource and uses it to map the territory. *Chartability*, from Chapter 4, ultimately ends up framing the rest of this thesis. As we did work alongside our co-designers, clients, and industry partners over the years, applying *Chartability* to their work, it became clear that three major barriers emerged as the most pernicious (or least-resourced) for practitioners to address.

Navigation, interaction, and personalization are different in kind: in each of them, a practitioner who knows precisely what good access requires still cannot build it, because the means do not exist in their tools. Guidance cannot substitute for means. These areas then serve as opportunities to innovate, to build better, more general, or more specific tooling.

We unpack these three tooling gaps more below.

2.4.1 Navigation

The empirical case for structured navigation is now strong. Zong et al. showed through co-design that screen reader users want to traverse a visualization's structure, not only hear a summary: structure, navigation operations, and description together determine the quality of the experience [332]. Systems work has begun to deliver this for specific stacks, with Chart Reader designing navigation experiences with screen reader users [289] and Olli adapting existing charting libraries into traversable structure [25]. What practitioners still lack is twofold. They lack low-level building blocks: a way to give *any* data interface, whatever its renderer or framework, a navigable structure with arbitrary shape, rather than the fixed hierarchies particular systems assume. And they lack any visible, manipulable representation of navigation structure in their workflows: a sighted practitioner can see a color palette and act on it with existing tooling but cannot see a navigation graph and act on it, which makes navigation design incredibly burdensome to iterate on and wholly downstream from other visual work. Chapter 5 (*Data Navigator*) contributes the structural substrate, and Chapter 6 (*Skeleton*) contributes the authoring representation; their chapter-level related work sections cover visualization toolkits and authoring systems more in depth.

2.4.2 Interaction

History shows that input design, not just output conversion, changes what is possible. Kane et al.'s *Slide Rule* contributed multi-touch techniques that let blind users operate touchscreens, outperforming a button-based baseline and prefiguring techniques that later appeared in commercial screen readers [155]. That simple lesson has not yet reached data analysis. Accessible visualization interaction today is overwhelmingly what we call *access-oriented*: it exposes information that is already there, through navigation, sonification, or description. These methods are *interactive*, as existing work as argued [332], but we argue that the interaction is focused on perception of the representation and not enabling analytical interaction with and manipulation of the information that lies beneath the representation. Analytical work of the kind sighted analysts perform with cross-filtered, coordinated views depends on rapid, continuous loops of input and perception [123, 173], and a screen reader's serial, query-at-a-time interaction may be structurally unable to sustain such loops. Fast, complex manipulation of a data space is presently made inaccessible by the primary assistive tool used by blind analysts. Rethinking input for blind analytical interaction, beyond an *access-oriented* framing, is nearly unexplored territory. Chapter 7 (*Cross-perception*) takes this on, formalizing a design framework and contributing a hardware prototype; that chapter's related work covers tactile and haptic input devices in more detail.

2.4.3 Personalization

Accessibility research has long argued that systems, not users, should do the adapting. Put simply, this means that systems must be capable of adaptation. Arguably, this is the "soft" part of *software*. Yet, this softness can create conflict in system design, especially when an entire

field of work must refactor their dominant tooling in order to enable system adaptation by end users. Visualization faces this exact challenge.

Existing work has demonstrated that interfaces adjusted to fit a person's abilities and preferences measurably improved speed, accuracy, and satisfaction for users with motor impairments compared to default interfaces [92], and ability-based design generalized the stance into principles [320]. Good systems are made to fit a user's abilities and remove barriers that they face. It could be argued, in broad strokes, that the field of HCI centers on the goal of *fit* between a user and a computational system.

Yet, visualization toolkits are largely not made to be personalized by end users. They are rigid and *hard*; lacking that softness required for end-user manipulation. Visualizations are typically built with a *universal design* [116] approach, when being made accessible (if being made accessible at all): one design is intended to fit as many people as possible. In visualization, recent work shows screen reader users benefit from customizing even small things, like the textual tokens spoken during navigation [151]. Yet a significant gap exists that enables visualization systems to adapt to personalization at scale and for many different people. How should these systems be built and what would they require? Which dimensions and features and functionality of data visualizations *should* be manipulable and customizable to fit end users? How should systems be constructed in order to make personalization effective and useful? And ultimately: how do we make visualization toolkits *softer*?

Some existing work on customizable accessibility settings identifies what drives people to adopt or abandon personalization features [323], but not all of this work neatly translates into visualization system design and development. Additionally, and fundamentally, we simply lack an empirical record of what exactly different people with disabilities would even want or need to manipulate. Presently, no mainstream visualization library or system offers practitioners infrastructure for end-user preferences over data representations: no shared option vocabulary, no persistence, no way to reconcile preferences with authorial defaults. Most visualizations become completely static in their design. Those with manipulable elements are only manipulable through hacking or manual overrides, placing a significant burden of access on end users, exactly where ability-based design says it should not [320]. Chapter 8 (*Towards Softerware*) engages this domain in collaboration with a production charting library, considered by the empirical record to be the most-accessible toolkit according to present evaluation methods by a wide margin [162]; Chapter 8's related work section covers adaptive systems and personalization research in more depth.

2.5 What to Take Away

Five things from this chapter frame everything that follows. First, tools direct the work and the workers who use them, which makes tool-making a rich potential site for accessibility intervention and research into better understanding practitioner barriers when making and using tools. Second, the conversation between accessibility and data visualization is old, rich, and increasingly self-critical. Third, the field's output is lopsided: empirical breadth, exemplary prototypes, and guidance exist in abundance, while production tooling and practitioner action barely exist. The broader field's critical voices impose real constraints: tools do not solve access, and this

thesis will not claim they do. Instead, critical work implores us to bridge the gap between research and practice. People with disabilities have been saying enough, it's time to get to work and try to apply what we know at scale. Fourth, the gap is concentrated where practitioners' means run out in three specific ways: navigation lacks a structural substrate, interaction lacks input techniques for analysis, and personalization lacks infrastructure. Fifth, the first step in our thesis work should be to collect the empirical record and synthesize it for applied accessibility work in the context of interactive data visualization. This is Chapter 4's *Chartability* project. Given where the field stands, *navigation*, *input*, and *personalization* are the categories of work that follow Chapter 4; the chapters ahead carry this out as the body of this thesis, and our concluding chapters return to what tool-making accomplished and what it cannot accomplish, on the other side.

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, Skeleton, Cross-perception, and Softerware, 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 mapped the full landscape of accessibility barriers. *Chartability* helped us to identify and prioritize our three latter domains (navigation, interaction, and personalization) as the areas where the most severe gaps remain in practitioner work. *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 ecosystems.

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 four domains of work that we 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 [296], although in more robust evaluation work, people with disabilities are actively involved in the process [222].

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 meth-

ods [63, 222]. 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 [153, 258], 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 our 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 seven principles. These three domains did not simply "emerge" on their own; they were evidenced by our applied work. Alongside the research activities that comprise this thesis, in a freelancing role we applied *Chartability* by partnering with industry, government, and non-profit organizations, conducting over 150,000 tests (both manual and automated) and working closely with dozens of designers and engineers. From that body of work, three larger domains stood out as the areas where the most severe and dramatic accessibility barriers remained unaddressed: 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 [185]) 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 [332], 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 want 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 [9]. 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 [87, 123, 204, 306]. 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 [173]. 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* demands better software tooling and visual support and *interaction* requires 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 [115, 138]. In existing work on personalization and accessibility, studies have demonstrated that end-user control is great to have and can alleviate friction [151, 323], 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 II

Evaluation: Helping Practitioners Identify Accessibility Barriers

Chapter 4
Chartability: Heuristics as a Tool and Resource

4.1 Preface

Novices and experts have struggled to evaluate the accessibility of data visualizations because there are no common shared guidelines across environments, platforms, and contexts in which data visualizations are authored. Between non-specific standards bodies like WCAG, emerging research, and guidelines from specific communities of practice, it is hard to organize knowledge on how to evaluate accessible data visualizations.

In this chapter, we present Chartability, a set of heuristics synthesized from these various sources which enables designers, developers, researchers, and auditors to evaluate data-driven visualizations and interfaces for visual, motor, vestibular, neurological, and cognitive accessibility. In this paper, we outline our process of making a set of heuristics and accessibility principles for Chartability and highlight key features in the auditing process.

Working with participants on real projects, we found that data practitioners with a novice level of accessibility skills were more confident and found auditing to be easier after using Chartability. Expert accessibility practitioners were eager to integrate Chartability into their own work. Reflecting on Chartability's development and the preliminary user evaluation, we discuss tradeoffs of open projects, working with high-risk evaluations like auditing projects in the wild, and challenge future research projects at the intersection of visualization and accessibility to consider the broad intersections of disabilities.

4.1.1 Context

Most of the prefaces for each chapter tie the overall narrative together, and help kick start each chapter with an explanation and perspective that help motivate why the problem mattered to address, what interesting elements ultimately shaped the start of the work, and even some caveats or scoping concerns to the work worth mentioning up front.

But *Chartability* started the thesis and all of the work branched from here. So what started *Chartability*? A long time ago, I (Frank) first did a "Research Experiences for Undergraduates" (REU) hosted by the National Science Foundation at Clemson University. This was long ago, before my industry career in visualization. The REU Site focused exclusively on data visualization. It had various projects a student could choose from, one of which was focused on helping a humanities scholar, Gabriel Hankins, visualize a network of letters sent between historical figures. I had just gotten a copy of Tamara Munzner's *Visualization Analysis and Design* [206] and Johanna Drucker's *Graphesis* [69], hot off the press. I felt ready for that project. But because it was an REU *Site*, it had many different areas of expertise besides humanities and visualization in the program. And I was assigned a secondary advisor for the summer, Jill Gemmill, who wanted me to learn more about *graph visualization* and *color vision deficiency*. So my first foray into visualization was a mix of humanities, graphs, and accessibility.

I carried that work with me into a career in visualization. I was interested in accessibility, but only slowly gained little bits of expertise here and there as I matured as an engineer and designer: add alternative text (and keep it short), don't make your text too small or light, use clear, summary titles, and so on. I collected this set of tacit knowledge on accessibility over the years. But at Visa, later in my career, I had a mandate to make a design system library of chart components as accessible as possible by default. And I had just had success integrating motion design into the library, so I was looking for my next project to tackle at a deep level. It was a perfect topic.

But digging in to what it meant to make the most-accessible visualization library possible led to many dead ends and impotent guidelines. In the Web Content Accessibility Guidelines, contrast ratios frustrated me to no end. I knew we had good research in visualization on how to discriminate between elements based on size [277], but accessibility standards didn't align with the models in the latest research. And on broader topics, there is only so much applicable knowledge you can gain from reading 30, 40 articles like "top 5 tips for making your charts more accessible" or from accessibility research papers where they built some kind of obscure dongle and tested it and wrote guidelines for other researchers along the lines of, "ensure your dongle's focus levels are maximum dongle" or some even less-useful collection of knowledge.

I needed to know answers to specific, grounded, concrete questions. And I was shocked: is there really no clear set of low-level, detailed guidelines for how someone should actually do this work? We have so much accessibility knowledge spread out in different places, but none of it was really designed to be used in the context of visualization work.

So, I gathered a group of professionals (now friends) interested in the topic and we met almost monthly for nearly two years on the topic of visualization and accessibility. It was a combination of folks from that group, as well as others who were members of groups that I was a part of but didn't organize (on the topic of accessibility more broadly) that ultimately contributed to the work that became the first version of *Chartability*. The research paper we published attributes authorship to a small subset of the full scope of people who contributed to the project itself. When I talk about *Chartability*, I tend to differentiate between the *project* and the *paper* where we talk about how we made the project and tested it.

And testing it was interesting. We don't go into deep detail in the paper on this, but I wasn't affiliated with CMU at the time and didn't even anticipate ever writing a research paper on *Chartability*. But I wanted to see how people worked with it and designed what I figured was a minimal sort of study to see how people used it and how it influenced their work. We had *many* participants in that and I did quite a lot of work over those two years with random folks, many who will remain anonymous.

But when we got around to writing the paper, we had to go back and ask our old participants: "would you consent to being in our paper as a participant?" and many (most, in fact) said they wouldn't. This is because earnest accessibility work is about finding flaws, not solutioning. And I think that the various lawyers and marketing people who got wind of our work loved the free (or heavily subsidized) labor but didn't want the company or product to look bad in the process of a research paper coming out. I don't really have regrets in this regard, because the paper was still published. But at the time of our first rejection, I had doubts about whether I should just run a whole new study from scratch.

This project does say a bit about how research tends to, by the nature of how it operates as a

productive profession of its own, disincentivize certain kinds of work. I doubt many researchers in the academy would intentionally sign an NDA to run a study and gather data that they can't ever publish (even though this is largely how product research and many professional, non-academic researchers presently operate). But I was able to see the real inner workings of some complex visualization products and ecosystems that I wouldn't have been able to if my goal was to publish a paper from the start. But instead, my goal was to learn about how to make *Chartability* better and learn more about how people used it (which very much seems like the same framing as a research paper!). But because I wasn't actively planning to publish anything at the time, I gained collaborators that I wouldn't have had otherwise (some of whom I still chat and share ideas with to this day).

Another thing that was quite difficult about *Chartability* was finding people with expertise in visualization *and* accessibility *and* lived experience with a disability *and* who were working on a visualization project *and* willing to let me see what they were up to. Recruitment was tough for this. I wouldn't do it again, especially now that I am a researcher. When people first interacted with me, the spirit was generous: I'm giving them this resource and my time and they would give me their time and even potentially company secrets. But ironically, now that I am a researcher and my goal is open dissemination of knowledge, it is more difficult to work with certain people.

But *Chartability* sets the tone for this whole thesis: our work is for the data practitioners, whoever they are. I was one. And at first, *Chartability* was for myself. So, we carry this spirit into all of the projects of this thesis, including many of the risks involved when doing research in this way. *Skeleton* (in Chapter 6) unpacks this further as well: *action research* [121], when conducted true to its spirit, operates first and foremost from the problems, framings, and imaginations of a particular group or community first, and not solely from the outside perspective of a researcher. In that sense, this thesis earnestly engages the barriers that data practitioners face when they design, develop, analyze, and architect data experiences.

This chapter was adapted from our published paper:

F. Elavsky, C. Bennett, and D. Moritz, 'How accessible is my visualization? Evaluating visualization accessibility with Chartability', *Computer Graphics Forum*, vol. 41, no. 3, pp. 57–70, Jun. 2022.

4.2 Overview

26% of people in the United States self-report living with at least one disability [213]. Of those, 13.7% live with a mobility disability and 10.8% with a cognitive disability. Globally, the World Health Organization reports that 29% of the world lives with uncorrected or uncorrectable blindness, low vision, or moderate to severe visual impairment [215]. Access is a significant inclusion effort that has broad international impact, especially for data visualization.

Accessibility is the practice of making information, content, and functionality fully available to and usable by people with disabilities. As part of this process, practitioners need to be able to identify accessibility barriers. While general accessibility standards help, evaluating the inaccessibility of complex data systems can be a daunting and often expensive task. State-of-the-art automated compliance checkers only find 57% of accessibility errors [63], meaning accessible experiences must still be manually designed and checked for quality. And following standards may only account for up to half of the needs of people with disabilities [222] anyway. Additionally, the intended wide applicability of these general standards means they fall short for information-rich systems, such as data visualizations (which use size, color, angles, shapes, and other dimensions to encode information). These specific contexts, communities, and libraries that deal with data visualizations and information-rich interfaces often have their own tools and guidelines for use, but they seldom include accessibility. Finally, research at the intersection of data visualization and accessibility has yet to meaningfully permeate data visualization tools and communities and primarily focuses on blindness and low vision, neglecting diverse accessibility needs of people with other disabilities.

Synthesizing evolving accessibility standards, research findings, and artifacts from communities of practice into usable knowledge for a specific, evolving domain is a wicked problem. To address this, we present Chartability. Chartability is an accessibility evaluation system specific to data visualizations and interfaces which aims to help practitioners answer the question, "how accessible is my data visualization?" Chartability organizes knowledge from disparate bodies of work into testable heuristics based on the functional accessibility principles POUR (Perceivable, Operable, Understandable, and Robust) [303] and 3 novel principles CAF (Compromising, Assistive, and Flexible), which we added to attend to the unique qualities and demands of data visualizations. We refer to these 7 heuristic principles as POUR+CAF. Chartability is a community-contributed project that leverages the governance strategies of open source projects as a way to address the complex dual-evolution of both accessibility and data interaction practices.

We additionally present an initial, light evaluation of Chartability from the experience of practitioners using it. We set out to see if using Chartability reduces the barrier of entry into

this work for accessibility novices and if accessibility experts had any feedback to share about its use. We gave practitioners introductory material for Chartability and instructed them to use it according to their needs. We found that before using Chartability only accessibility experts believed auditing data visualizations to be somewhat easy or easy, while the other group believed auditing data visualizations to be somewhat hard or hard. All novice accessibility practitioners became more confident after using Chartability and believed auditing data visualizations for accessibility to be less difficult. Conversely while the expert accessibility practitioners were already confident in their ability to evaluate accessibility (and all unanimously had no change in their before and after evaluations), they were excited to adopt Chartability into their set of auditing resources.

Our work sets out to acknowledge that data practitioners face significant barriers when first making data visualizations, systems, and experiences accessible. While Chartability contributes to filling gaps and organizing knowledge, it also challenges visualization and data interaction researchers to explore new horizons of possibilities in this space. As such, we conclude with recommendations for future research at the crossroads of data visualization and accessibility.

4.3 Existing Work in Data Visualization and Accessibility

While recent works at the intersection of data visualization and accessibility are promising, they do not provide a consistent and unified methodology for designers to evaluate the accessibility of their work across the broad spectrum of disability considerations.

4.3.1 Research Advancements in Data Visualization and Accessibility

In parallel to Mack et al.'s "What do we mean by Accessibility Research?" [181] when we asked "What do we mean by data visualization accessibility research?" we found that nearly all topics of study were vision-related. 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) [161]. Marriott et al. found that there is no research at all that engages motor accessibility [191]. 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 [324] and seizure risk [273, 274]). 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) [284].

Since the 1990s, the most prominent and active accessibility topic in visualization has been color vision deficiency [43, 170, 193, 211, 214]. Research projects that explore tactile sensory

substitutions have been a topic in computational sciences dating back to 1983 [247], with tactile sensory substitutions being used for maps and charts as far back as the 1830s [112]. Sonification used both in comparison to and alongside visualization and tactile methods for accessibility dates as far back as 1985 [33, 57, 87, 187, 196, 330]. Some more recent work has explored robust screen reader data interaction techniques [102, 271], screen reader user experiences with digital, 2-D spatial representations, including data visualizations [243, 256], dug deeper into the semantic layers of effective chart descriptions [177], and investigated how to better understand the role of sensory substitution [50]. 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 [154]. We find all of this emerging work promising and foundational.

Despite this promising work emerging, we also want to acknowledge a spectrum of other work that exists at the intersection of accessibility and data visualization that does not serve the goals of our project. There is significant research that explores automatic or extracted textual descriptions [13, 45, 46, 47, 167, 212, 223, 255] and haptic graphs and tactile interfaces [5, 12, 26, 35, 38, 93, 147, 148, 168, 247, 248, 261]. These research projects produce artifacts that are high-cost for individual use, some are not robust enough to interpret complex visualizations effectively, and several have not included people with disabilities. Since our goal is to synthesize knowledge for practitioner accessibility work, we also acknowledge that some of these projects did not follow standards during their research project and in their output, such as using Web Content Accessibility Guidelines [296] or The American Printing House for the Blind and Braille Authority of North America [14]. All of these challenges are factors that limit the generalizability of these artifacts and knowledge for practitioner use [178, 203, 256]. We encourage work to continue at the intersection of accessibility and visualization, but stress the importance of practical, disability-led research that either builds on or explicitly challenges standards.

4.3.2 Accessibility Practices in Data Visualization Tools and Libraries

Our research goals are to find what is already being done in data visualization and accessibility and to see if we can enhance that activity. To this aim, our background investigation includes a broad and comprehensive exploration of the field of practitioner and non-academic artifacts.

Some open source and industry contributions have pushed data visualization and related accessibility efforts. Libraries like Highcharts [128] or Visa Chart Components (VCC) [295] and tools like the Graphics Accelerator in SAS [240] have broad accessibility functionality built in, but their documentation is technically specific to their implementation. While these relatively accessible libraries and tools can be helpful for inspiration, their specific techniques and guidance materials are not easily transferable to other environments or applications where data visualizations are created. Practitioners must reverse engineer and deconstruct many of the methods employed by these libraries, and with the exception of VCC (which is open source), this task requires significant effort, given their primarily closed-box nature.

In common charting tools and libraries (apart from those already mentioned) accessibility engineering is often not present, limited in scope, or has only recently become an effort. More established visualization libraries like matplotlib, ggplot2, d3js, R-Shiny, and Plotly have left

most accessibility efforts to developers, with varying levels of documentation and difficulty involved [40, 54, 89, 90, 309]. None of these major tools have a broad spectrum of accessibility options built in and documented.

Community contributors often must fight to make their tools and environments accessible (sometimes even against the design of the tools themselves) with little to no compensation for their contributions. For example, Tableau's first accessible data table was built by a volunteer community member Toan Hoang as an extension [132]. Tableau users more broadly must resort to voting systems to gather attention to accessibility issues [62]. Semiotic's accessibility features were added by community member Melanie Mazanec [195]. For Microsoft's PowerBI, students have organized resources for how to make visualizations built with it more accessible [163] while non-profits like the City of San Francisco's data team have had to build features like keyboard instructions from scratch [236]. Mapbox GL JS is an example of a popular mapping library (over 400,000 weekly downloads) [189] that has no built-in accessibility support by default. The accessibility module for Mapbox GL on GitHub was created and maintained by volunteers but has had less than 10 weeks of work with any activity invested since its first activity in late 2017 [188].

Many community-driven efforts are under-utilized, must be discovered outside of the primary environment's ecosystem, have poor or no core, internal support, and are inconsistently and partially implemented. Accessibility is still an afterthought in data visualization and ad-hoc, specific solutions proposed have not led to widespread improvements.

4.3.3 Accessibility in Practice, Broadly

Accessibility in practice is largely motivated by standards work or assistive technology. We want to acknowledge that tactile and braille standards are robust [14], but have limited transferability to digital contexts currently. For example, whereas tactile graphics guidelines lend insight into information prioritization, layout, and fidelity, the assumption is they will be embossed onto paper or similar physical mediums [23, 178, 256].

In digital contexts, the most influential body for accessibility is the World Wide Web Consortium's (W3C) Web Accessibility Initiative (WAI). WAI's Web Content Accessibility Guidelines (WCAG) [296] influence accessible technology policy and law for more than 55% of the world's population [143]. WAI and WCAG outline 4 types of functional accessibility principles: Perceivable, Operable, Understandable, and Robust, abbreviated as POUR [303]. POUR is the foundation that organizes all 78 accessibility testing criteria in WCAG.

4.3.4 Using Heuristics to Break Into Under-addressed Areas

To summarize the complex problem space to which this paper contributes: Research in data visualization primarily focuses on visual accessibility, accessibility standards focus on a broad range of disabilities but lack deep contextualization for data visualization, and practitioners seem to build a wide array of solutions to fill these gaps, most of which are poorly maintained or adopted. Any time a practitioner wants to embark on a journey learning how to evaluate the accessibility of a data visualization, they must collect and synthesize this complex space of knowledge themselves. We have included (with permission) an exemplary field artifact as an example of this type

of labor in our supplemental materials, which contributed to the United States Government's project, "Improving Accessibility in Data Visualizations" [292, 293].

After gathering information with this breadth and complexity, a heuristic evaluation model was chosen as a way to deliver useful but flexible knowledge. Heuristic evaluation models have a long history in HCI and are cheap to use and require little expertise. They have been shown to be effective methods for practitioners compared to user testing, focus groups, or other evaluative methods that require existing expert knowledge or recruitment, moderation, and compensation of participants [28, 48, 98, 152, 193, 209, 210, 216, 238, 267]. Heuristics are also not new in visualization [56, 88, 232, 249] even among topics related to accessibility (color vision deficiency, specifically) [214, 238].

4.4 Making Chartability

We next present Elavsky's work to develop Chartability as a real-world design process contribution to the larger research community. Our making process does not neatly fit into most design models that divide researchers from practitioners. In Gray's different models of practitioner-researcher relations, our work is some variation of bubble-up, practitioner-led research [106]. This project was initiated by Elavsky while they were an industry practitioner, deeply situated in this work already.

Thus, the following description of Chartability's 10-month creation is written from Elavsky's perspective. The supplemental materials include the data from this stage of the process, a preview of which is available in Table 4.1:

1. **Situate, Survey, and Select Problem Space**: I was situated within the context of accessibility evaluations of data visualizations. From personal experience, I recognized the prohibitively significant labor involved in ensuring I was effectively following accessibility standards while also attending to the complex design considerations of data visualizations. To improve this work both for myself and others in the future, I surveyed existing problems and challenges others faced and selected a solution that I felt equipped to address.
2. **Collect Existing Resources**: I set out to answer, "If evaluating the accessibility of data experiences is hard, what do existing standards miss?" I evaluated my seed knowledge (WCAG criteria) for shortcomings and gaps and collected other data relevant to my goal (academic and industry research, open-source libraries, tools, applications, data products, government guidelines, design guidelines, software documentation, university coursework, and practitioner articles).
3. **Code Resources**: After collating these resources (including relevant WCAG criteria), I loosely borrowed from thematic analysis [31] and qualitatively coded this data. I developed a set of 29 codes starting with WCAG's POUR principles and expanded the codes to account for other concerns that came up in the resources, including what type of accessibility was being addressed (e.g., cognitive, visual), whether a solution was technology-specific or agnostic, and other categories (like "time-consuming" or "user-controlled"). I then divided the resources into codable segments with relatively distinct pieces of information and applied the 29 codes to the information segments. I grouped information with

codes in common, resulting in a representative 45 groups of related information segments.

4. **Synthesize Heuristics**: Since auditing depends on measurable heuristics, I adjusted each of the 45 groupings that resulted from the qualitative analysis into phrasing that could be verified by an evaluation. I then augmented each heuristic with known testing procedures, resources, and tools necessary for applying them in practice. 10 critical heuristics (these were determined top priorities through user feedback) are previewed in Table 4.1, with the full version of this table (and more) provided in our supplemental materials.
5. **Group Heuristics into Higher-level Principles**: I linked each heuristic with relevant web accessibility standards and POUR principles to draw a familiar connection for users who might already be accessibility practitioners. 26 heuristics fit neatly back into Perceivable, Operable, Understandable, or Robust.
6. **Develop Remaining Themes into New Principles**: 19 remaining heuristics with complex codes and overlapping groups demanded new theorizing, as they either did not fit into POUR at all or could arguably belong to multiple principles at once. I analyzed these remaining complex heuristics for similarities and organized them under 3 new themes, which we are contributing as new accessibility principles, Compromising, Assistive, and Flexible, defined below.

			Coding categories	
Heuristic title	**Principle**	**Origin**	**POUR**	**Other**
Low contrast	Perceivable	Standard	P	2
Small text	Perceivable	Research	P	2
Content is only visual	Perceivable	Standard	P, R	3
Interaction has only one input	Operable	Standard	O, R	3
No interaction cues/instructions	Operable	Standard	O, U	2
No explanation for how to read	Understandable	Research	U	1
No title, summary, or caption	Understandable	Research	U	1
No table	Compromising	Research	O, U, R	3
Data density inappropriate	Assistive	Research	P, U	4
User style change not respected	Flexible	Standard	P, O, R	6
. . . and 35 additional non-critical heuristics				

Table 4.1: Previewing Chartability's 10 critical heuristics. The coding categories are split into two parts: which POUR principles contributed to each heuristic, and how many additional coding categories ("Other") were assigned.

4.4.1 Compromising

Compromising is a principle that focuses on Understandable, yet Robust heuristics. These heuristics are based on providing alternative, transparent, tolerant, information flows with consideration

for different ways that users of assistive technologies and users with disabilities need to consume information.

Compromising challenges designs that only allow access to information through limited or few interfaces or processes. These heuristics focus on providing information at a low and high level (such as tables and summaries), transparency about the state of complex interactions, error tolerance, and that data structures can be navigated according to their presentation. Compromising designs have both information and system redundancies in place.

4.4.2 Assistive

Assistive is a principle that primarily builds off the intersection of Understandable and Perceivable principles but focuses on the labor involved in access. These heuristics include categories that encourage data interfaces to be intelligent and multi-sensory in a way that reduces the cognitive and functional labor required of the user as much as possible.

The Assistive principle focuses on what Swan et al. refer to as "adding value" [283] and what Doug Schepers meant by "data visualization is an assistive technology" [245]. We visualize because it is faster and more efficient than munging cell at a time through data. Assistive heuristics ensure that both visual and non-visual data representations add value for people with disabilities.

4.4.3 Flexible

Contrasted with Compromising (which focuses on robust understanding), flexible heuristics focus on robust user agency and the ability to adjust the Perceivable and Operable traits of a data experience. Flexible heuristics all have a tight coupling between a data experience and the larger technological context the user inhabits. The preferences that a user sets in lower-level systems must be respected in higher level environments.

Self-advocacy and interdependent agency are important sociotechnical considerations that engage the conflicting access needs that different users might have in complex technological interactions like data experiences [18, 186]. Some users might want specific controls or presentation, while others might want something else entirely. Designs must not be rigid in their opinions and ability assumptions and should be designed to be moldable by and adaptive to user needs [166, 320].

4.5 Using Chartability

All of Chartability's tests are performed using Chartability's workbook [74] alongside various tools and software (linked in the workbook). For the scope of this paper, we are not including an explanation for how to perform all of these. Both the workbook and supplementary materials with this paper give more details.

While a highly trained auditor may be able to casually evaluate an artifact in as little as 30 minutes or even hold heuristics in mind as they are doing their own creative work, those new to auditing may take anywhere between 2 and 8 hours to complete a full pass of Chartability. Professional audits, which can take weeks or months, often include multiple auditors and provide



Figure 4.1: A low contrast chart (left) compared to a higher contrast version (right). A dropper tool is extracting the fill color of the bar and then a contrast ratio has been calculated. Note that the fill color is the same on both bars, but darker borders have been added to ensure the visualization passes contrast tests.

rigorous documentation and detailed recommendations for remediation, typically in the form of a report. Chartability is meant to serve both quick pass and deep dive styles of audits, so users are expected to leverage it as they see fit.

Below we give an example of what might be a quick pass audit, using Chartability. Which principles are applied in each of these stages are listed in parentheses in each heading.

4.5.1 Visual Testing (Perceivable)

Checking for contrast is the most common critical failure; 87.5% of tests (7 out of 8) from our user study involving this heuristic failed, which supports the WebAIM Million Report's findings (83.9% of the top 1 million websites also fail contrast testing, more than any other WCAG criterion) [308]. In order to evaluate contrast, often a combination of automatic (code-driven) and manual tooling is performed. When manually auditing, practitioners typically use a dropper and a contrast calculator (Figure 4.1). Most auditors find this to be one of the easiest tasks to perform and accomplishes 3 different heuristics in Chartability: ensuring text/geometries have contrast, interactive states for elements have enough contrast change, and the keyboard focus indicator is easy to distinguish.

Perceivable heuristics also include tests and tools for color vision deficiency and ensuring that color alone isn't used to communicate meaning (like the redundantly encoded textures in Figure 4.8). And another common, critical failure from Perceivable is text size. No text should be smaller than 12px/9pt in size.

4.5.2 Keyboard Probing (Operable, Assistive)

The next practice that most auditors should become comfortable with is using a keyboard to navigate and operate any functionality that is provided. Most assistive technologies, from screen readers to a variety of input devices (like switches, joysticks, sip and puffs, etc) use the keyboard API (or keyboard interface) to navigate content. If a data interface contains interactive elements (Figure 4.2, Figure 4.3), those elements (or their functionality) must be able to be reached and

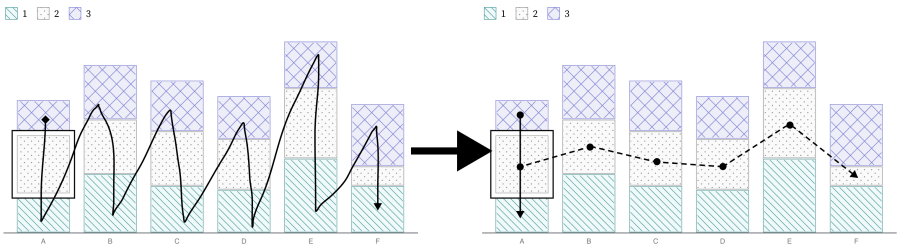


Figure 4.2: Keyboard navigation paths on a stacked bar chart. The left shows a serial navigation example, typically just a default of rendering order. The right shows both groups (the stack of bars) and categories (the color/texture shared among bars across stacks) as dimensions to explore laterally or vertically.

controlled using a keyboard alone. Auditors should be critical of how much work is involved in keyboard navigation, especially (Figure 4.7). All that is required to start is that the auditor begins pressing the tab key to see if anything interactive comes into focus. Arrow keys, spacebar, enter, and escape may be used in some contexts. Generally, instructions or cues should always be provided.

Using a keyboard provides an opportunity to evaluate many different heuristics: checking for multiple inputs (Figure 4.3), whether the data structure that is rendered is navigable according to its structure (Figure 4.2), and whether keyboard navigability across all elements in a data interface is even necessary (Figure 4.7).

4.5.3 Screen Reader Inspecting (Perceivable, Operable, Robust, Assistive)

Closely related to keyboard testing is testing with a screen reader. Some things may work with a screen reader that do not with a keyboard (and vice versa), so both must be evaluated.

Screen readers, unlike more basic keyboard input devices, read out content that is textual (including non-visual textual information like *alternative text*). Using a screen reader to audit is generally the hardest skill to learn. Keeping this in mind, testing whether the meaningful text provided in a visual (such as in Figure 4.4) is accessible with a screen reader is the easiest and most basic test that auditors should first perform.

Next, all valuable information and functionality in a data experience should be tested whether it is available to a screen reader. This includes the individual variables about a mark as well as whether that mark is interactive (Figure 4.5), whether status updates that reflect context change provide alerts (Figure 4.3), and whether summary textual information is provided about the whole chart (Figure 4.4) as well as statistically and visually important areas of that chart (Figure 4.7).

4.5.4 Checking Cognitive Barriers (Understandable, Compromising)

First, auditing for cognitive barriers generally involves checking the reading level and clarity of all available text using analytical tools. But Chartability also requires that all charts have basic text that provides a visually-available textual description and takeaway (Figure 4.4). This alone is

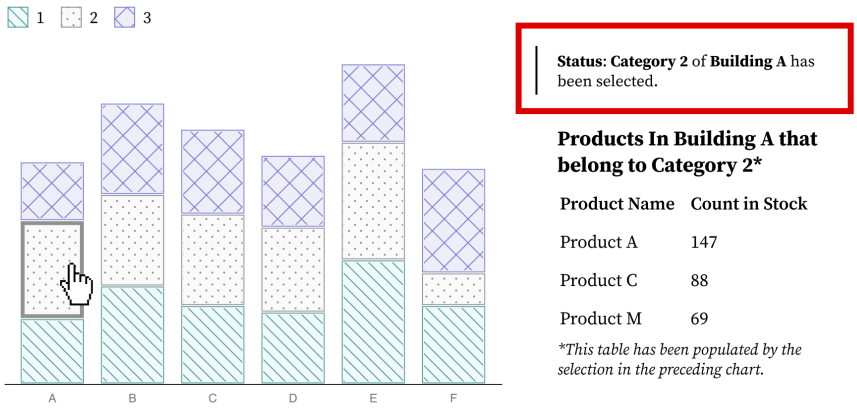


Figure 4.3: A mouse cursor is selecting a bar (left, shown with a thick indication border) in a stacked bar chart to filter a dataset (on the right). A system alert (red box) notifies the user of their interaction result. This selection capability must also be provided for the keyboard interface and the alert must be announced to screen readers.

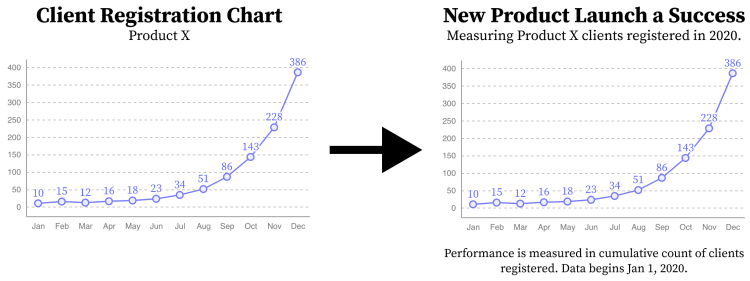


Figure 4.4: Charts must have a visually available textual explanation provided that summarizes the outcome. “Client Registration Chart” for “Product X” (left) is inaccessible while “New Product Launch a Success” (right) gives a clear takeaway.

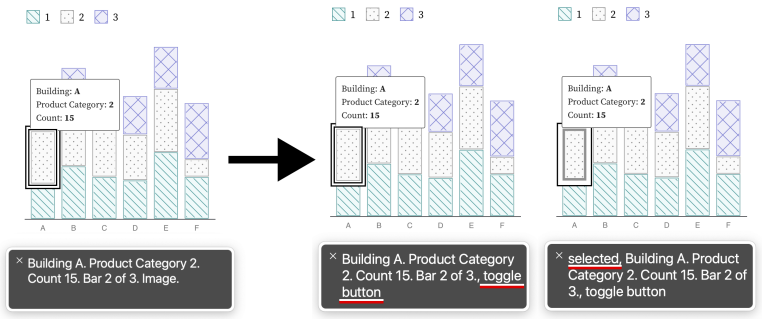
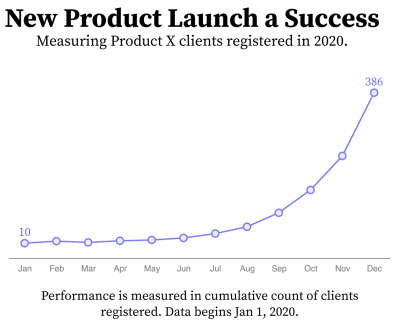


Figure 4.5: An interactive chart displaying only “Image” as semantic information with no feedback provided on selection. The robust semantics given to a screen reader, “toggle button” (middle) as well instant feedback, “selected” (right) are considered proper semantics for an interactive experience.



Month	Registered Clients
Jan	10
Feb	15
Mar	12
Apr	16
May	18
Jun	23
Jul	34
Aug	51
Sept	86
Oct	143
Nov	228
Dec	386

Figure 4.6: A line chart (left) with a single line and an accompanying data table (right). This line chart would not provide enough low-level information about each datapoint without the table provided. A table alone however would also be inaccessible. Providing both can satisfy conflicting accessibility needs for different audiences.

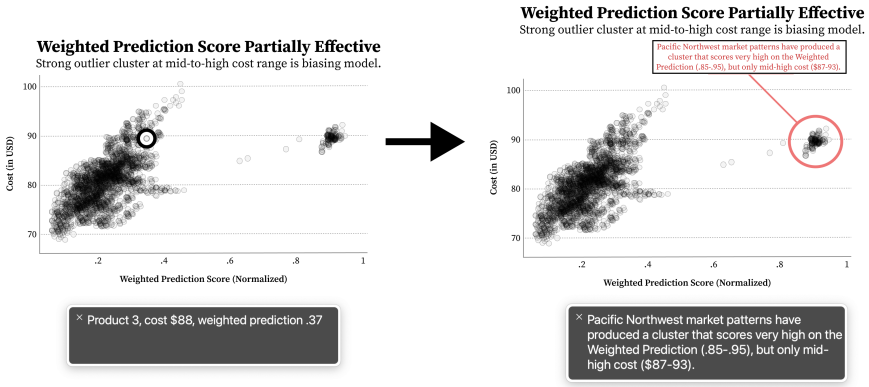


Figure 4.7: A scatterplot with many points, where a single point within the chart can be accessed by a screen reader (left). Navigating this data piece by piece is unnecessarily tedious, so an annotation callout is provided to help the reader focus on an outlier cluster (right). The callout is being accessed by a screen reader, which is displaying the annotation's summary as well.

one of the most important things to check for. In complex cases where a chart has a visual feature with an assumedly obvious takeaway, checking for annotations or textual callouts is important to help avoid interpretive issues [325] (Figure 4.7).

4.5.5 Evaluating Context (Robust, Assistive, Flexible)

The final series of checks an auditor should make involve thinking about the overall work in a design (as it intersects with other considerations) as well as the larger technical context where the user is situated.

Auditors should first try to change system settings (such as toggling high contrast modes) to see whether a data experience respects these settings (Figure 4.8), run automatic semantic evaluations as well as manually check for appropriate meaning (Figure 4.5), and check if dense or highly complex visuals have sonified, tactile, or textual summaries available (Figure 4.7). Auditors should also check whether system updates provide clear feedback textually (Figure 4.3) as well as checking if there are both high and low level representations of information available (Figure 4.6).

Auditors should be especially critical of static designs, such as those that either use textures by default or not (Figure 4.8), which are a high risk of compromising and assistive failure.

4.6 Validating Chartability

Next, Elavsky explains the preliminary user evaluation: I validated whether data practitioners felt more confident and equipped to make their own work accessible with Chartability. Additionally, I also wanted to interview expert accessibility practitioners (including those with disabilities) with the same questions, to see if Chartability had anything to offer in helping them understand and evaluate data experiences better.

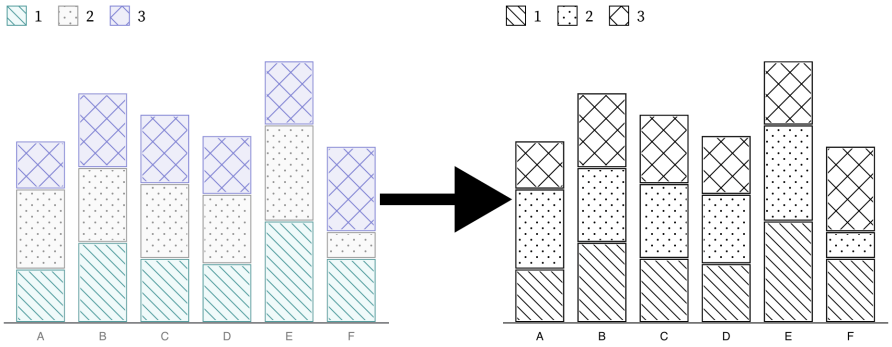


Figure 4.8: A bar chart with categories (left) shown not conforming to Windows High Contrast White Mode. High contrast mode on Windows requires limiting color palettes, using only black or white for most elements (shown on the right).

My secondary goal was to present a tool that can be helpful even in the wild on real projects (with all the weird design and engineering quirks that come with that). I wanted Chartability to be usable on things built with a tool like Tableau and fully bespoke, hand-coded visualizations, like those made with JavaScript and D3. To this secondary aim, I intentionally solicited participants who were working on a variety of different projects, each of their own design.

4.6.1 Pre-Validations and Flipped Roles: Participants Question *Me*

I performed several early, light validations of my work before soliciting and involving participants formally. My early pre-validations #2-4 (below) all focused on practitioners asking me questions and giving feedback.

My 4 pre-validations happened during the process of making Chartability, as well as introducing short iterations back into the making process:

1. **Beta Testing**: I performed several beta tests of Chartability during the process of making. I audited using versions that only had POUR principles, tried versions of Chartability that focused only on standards, and also tried out different iterations of the heuristics as I was forming them. This testing was important to perform early in the process because it helped me test the limits of various possible directions for this tool (standards-only, against standards, building off of standards, etc).
2. **Early Advice**: After the first full pass of making Chartability was complete, I sent Chartability via email to 4 accessibility experts and 6 interested people with disabilities familiar with auditing in order to solicit open feedback.
3. **Professional Workshop**: I held a half-day professional workshop via Zoom on auditing visualizations for accessibility and presented Chartability's heuristics to this select audience of 50 participants. I demonstrated how to audit and then had a chance for feedback and questions.
4. **Deep Feedback Session**: I presented Chartability to 14 experts on data visualization and accessibility, 5 of which are people with disabilities. I presented in two separate sessions

through 2-hour video calls on Zoom (roughly one hour was demonstration and one hour was discussion).

4.6.2 Discovering "Critical" Heuristics

These pre-validations helped me combine and divide some of the heuristics, adjust the language and phrasing, and label 10 specific tests as "Critical," which can be seen in Table 4.1. These critical tests were ones that community members stressed as an important priority for one or more of the following reasons:

- They are prohibitively expensive to fix late.
- The barriers they produce are too significant to ignore.
- They are among the most common type of accessibility failure.
- They affect many parts of a data experience.

All Critical heuristics are based on standards or research.

4.6.3 Selecting Participants and Projects

I was a practitioner and representing myself as a volunteer when I reached out to participants. At this stage in the project, I was still not affiliated with a research institution and was not interested in producing publishable knowledge. I intended to test Chartability in the wild and validate whether it achieved its aims. My priority was to collaborate with folks working on difficult problems or those who had a rare intersection of expertise between accessibility standards and interactive data experiences. To this end, I was highly permissive with potential collaborators in order to maximize the expertise of participants and breadth of environments for testing Chartability.

However, part of being permissive with participants meant that I was willing to collaborate on projects that I cannot share in a research publication and many of my participants must remain anonymous (including interview results that contain sensitive information about intellectual property). Given that auditing is a field of work about identifying failures, there was both a high demand for participation in the evaluation of Chartability in tension with a low motivation to make these failures known in a public venue.

A summary of our selection process:

- **Solicitation**: I reached out via email to 24 individuals in my network to participate in helping to evaluate Chartability. I mentioned that I wanted Chartability to be applied to a current project of theirs and was interested in performing some interviews about their experience before and after using Chartability. I mentioned up front that working with me would be uncompensated and potentially take multiple hours of their time (even multiple sessions) over Zoom meetings.
- **Response**: 16 individuals were interested and shared their project details (2 would require an NDA to be signed).
- **Selection**: I selected 8, based either on the expertise of the individuals, on the robustness of their project, and/or on the opportunity to get feedback about Chartability in team

environments (which I didn't anticipate, but 3 of the 8 represented team efforts).

- **Resulting Group**: I worked with 19 total participants across 8 environment spaces.
- **Publishable Group**: Due to intellectual property concerns, I can publish interview results from 6 participants and discuss the details of 4 audit environments.

Chris DeMartini: a multi-year Tableau Zen Master and recognized expert visualization practitioner. His dashboard of a coin flipping probability game dataset that he produced with his daughter was the subject of his audit [61]. His audit only included criteria labeled Critical in Chartability (which involves only 10 tests instead of the full 45) and his dashboard failed 7 of them. A full audit was later conducted on Chris's behalf. His full audit had a total of 26 failures, 11 of which were considered non-applicable.[1]

Amber Thomas: a data storyteller and technologist credited on 30 of The Pudding's visual essays. Amber has had a growing interest in accessibility challenges related to her line of work designing and developing state of the art, bespoke visual essays. Her article The Naked Truth was still in the early design and development stages when it was fully audited [8]. It failed 22 out of 45 tests, including 6 out of 10 criteria considered Critical. 6 tests were considered non-applicable.[1]

Sam (self-selected pseudonym): a recognized design practitioner in the visualization community who lives with disability. They were collaborating on an interactive data project that would be specifically made to be used by international participants with a broad spectrum of disabilities. Their interactive infographic failed 21 out of 45 tests, 5 of which were considered Critical. 10 tests were considered non-applicable.[1]

Øystein Moseng: Core Developer and Head of Accessibility of Highcharts. Øystein was interested in taking one of Highchart's demo charts not specifically developed with accessibility features in mind [130] and testing it against a full Chartability audit to see how it held up. The demo failed 13 out of 45 tests, 3 of which were Critical. 10 tests were considered non-applicable.[1]

Jennifer Zhang: a senior accessibility program manager at Microsoft with expertise working on enterprise data products.

Ryan Shugart: a blind, screen reader user and disability subject matter expert at Microsoft who has a strong expertise in collaborative accessibility for interactive data systems.

Both Shugart and Zhang were interested in applying Chartability internally and testing its effectiveness and potential with various projects. Their application and use of Chartability (including audits) are not available for publication, but their valuable interviews and evaluations are included with permission.

4.7 Study Results

I asked the 6 participants a series of qualitative and Likert-scale evaluation questions:

1. Have you ever performed an audit of a data experience before?

[1] "Non-applicable:" any test in the auditing process that does not contain content relevant to the test, such as "Scrolling experiences cannot be adjusted or opted out of" for a visualization that does not have a scrolling input control

2. What stage of production is your project in? Analysis, design, prototyping, development, maintenance?
3. How confident are you in your ability to perform an audit of a data experience for accessibility issues? (1-5, 1 being not confident at all, 5 being fully confident.)
4. How difficult do you perceive auditing a data experience for accessibility issues is? (1-5, 1 being trivial, 5 being very difficult.)
5. (After using Chartability) How confident are you in your ability to perform an audit of a data experience for accessibility issues? (1-5, 1 being not confident at all, 5 being fully confident.)
6. (After using Chartability) How difficult do you perceive auditing a data experience for accessibility issues is? (1-5, 1 being trivial, 5 being very difficult.)
7. (After using Chartability) Do you intend to continue using Chartability?

Each of these questions had an open-ended question attached, "Is there anything else you would like to add?" Every participant provided additional input on questions 3 through 7.

None of the 3 participants who only consider themselves expert data practitioners had performed an audit before. All 3 of them reported that they believed auditing to be easier and that they are more confident in their ability to evaluate the accessibility of data experiences after using Chartability.

Of the 3 accessibility experts (all of whom have performed audits of data experiences before), their opinions on these measurements were unchanged after using Chartability. All 6 participants noted that they plan to use Chartability in their own work and would recommend it to their peers.

Below we overview some of the key insights Elavsky received from the open ended responses.

4.7.1 Real Access has more Considerations than Colorblindness

Among the data practitioners, DeMartini wrote after his audit, "I have read a lot about color blindness and could provide meaningful feedback to visualization developers on that topic, but I have come to realize that accessibility is so much more than this and I basically didn't really know where to start when it came to the true scope of accessibility." He ended his qualitative feedback with, "I think this could be a great tool for the masses and really look forward to the impact it can possibly have on the (inaccessible) data visualizations which are being created in huge numbers these days."

4.7.2 Audits are Slow, but Help me Focus

Amber Thomas wrote, "It still takes a while to do a complete audit, but it's not hard! For someone new to the space, all the possible options that can be used to make visualizations more accessible can be overwhelming. [Chartability] helped me to focus." She finished her feedback with, "There aren't really guidelines (at least to my knowledge) that exist to help data visualization creators to ensure their work is accessible... [Chartability] helps to direct users to the most common

accessibility problems with straightforward questions. It really helps to narrow the focus and prioritize efforts."

4.7.3 Chartability Helps me Remember and Stay Consistent

Among the accessibility experts, Zhang wrote, "While I am skilled, depending on the day I might not remember everything I need to look at. I am more confident in consistency between different auditing sessions. For experts it's a good reminder framework." Moseng of Highcharts noted, "[Chartability] did a very good job of highlighting concerns that are often ignored or forgotten when auditing and designing/developing." Shugart of Microsoft added along those lines, "I feel [Chartability] arranges a good set of questions in a user's mind and makes it easier for them to determine if a visualization is accessible."

4.7.4 Access is an Experience, not just Compliance

Zhang offered insight into the design intention of Chartability, "[it is] clearly going for above compliance and focusing on a good experience." Sam expressed their need to make an excellent accessibility experience, "I am not just worried about compliance, but I want to make something really good. Nothing seems to help you go beyond? This is better than WCAG, I can already tell."

4.7.5 Everyone wants More Evaluation Resources and Tools

For constructive feedback, all the data experts noted that they wanted more resources and materials related to learning the skills needed to conduct an audit. Shugart and Moseng both noted that they hope for more tooling and (in some cases) automated tests that can take the burden off the auditor and streamline the design and development process (much like Axe-core [64]). They both also agreed that automation and tooling would help novice practitioners perform this work faster and with more confidence. 2 of the 3 mentioned wanting more examples of failures as well as accessible data experiences. Sam wrote that they felt Chartability was overwhelming at first, but after focusing on just the Critical items, the rest of the framework "became easier."

4.7.6 Experts: "Novices will Struggle." Novices: "This was so helpful"

The accessibility experts all unanimously agreed that Chartability is helpful to their own work, but they are unsure how accessibility novices would do. They all believe that more training and resources are needed to help people who are new, with one noting that Chartability could even be "overwhelming" to someone who has not been exposed to accessibility work before. All of the novices remarked that Chartability was "so helpful," "made this work so much clearer than before," and "made a lot of hard problems not as hard."

4.7.7 What about Auditors with Disabilities?

Shugart's feedback was critical when discussing continuing to use Chartability, "I still feel as a screen reader user, the audit itself would have some unique challenges because I'd be missing a lot and would have problems determining things such as color." He continued, "Auditing anything accessibility-wise as a screen reader user poses challenges because you don't always know what you're missing. In many cases there are workarounds to this but datavis is one area where this is really hard to do now."

4.8 Extended Results

Following calls to ensure accessibility work has practical benefits that exceed publications [139], in April, 2021 Elavsky made Chartability openly available on GitHub. As new research and practices emerge and more community members get involved, Chartability will become an evolving artifact of consensus similar to existing standards bodies [304].

Projects like Turkopticon benefited from the discussion about how a community actually used their tool [144]. In the same vein, we are happy to report some valuable findings from within this last year that we think demonstrate (in a pragmatic way) that Chartability has some merit:

- **It is living and growing**: Chartability has received enough community feedback that it is now on Version 2, with more tests and background resources provided.
- **People are talking about it**: Chartability has been featured in 14 workshops, talks, and podcasts and at least 2 university courses.
- **People are using it**: Chartability has contributed to projects at Microsoft, Highcharts, Project Jupyter, Fizz Studio, FiveThirtyEight, Vega-Lite, UCLA, the City of San Francisco, the Missouri School of Journalism, a Fortune 50 company, two Fortune 500 companies, and community groups (like MiR).
- **It has breadth**: Chartability has evaluated static and interactive data experiences made with Microsoft's Excel and PowerBI, Tableau, JavaScript (D3, Vega-Lite, Highcharts, Visa Chart Components), Python (Altair, Bokeh, and matplotlib), R (ggplot2), as well as design sketches and low/medium-fidelity artifacts (Illustrator, Figma, Sketch).

When considering the analysis by Hurst and Kane about high abandonment rates in assistive technology, [139] we wanted to make sure that we created an artifact (assistive technology or otherwise) that would at least survive its first year of use in the real world.

The greater community feedback as well as new research before and after open-sourcing Chartability has also led to 5 new heuristics being added since our test users performed audits and gave evaluations. The current version of Chartability (v2) has a total of 50 heuristics.

It is important to note that the work of Chartability did not begin and does not conclude with the publication of this manuscript. We want Chartability to become a living, community-driven effort that will adapt and grow as more resources, tools, and research become available.

4.9 Discussion

From our presentation of Chartability and a preliminary user evaluation with data visualization and accessibility practitioners, we learned that Chartability reduced the perception that working on accessibility is difficult and increased the confidence of those new to this work. Chartability shows promise as a useful framework for expert accessibility practitioners because it serves to produce consistency in contexts like the evaluation of dashboards, data science workflows, and other complex, data-driven interfaces.

While our practitioners with novice accessibility experience were initially concerned about doing the audit correctly, most of their audit results were reasonably comparable with that of the authors (although their time to complete was much longer).

We agree with experts that beyond Chartability, more resources are needed which provide examples of both inaccessible and accessible data visualizations as well as how to perform some of the more difficult parts of the auditing process (such as evaluating with a screen reader). We hope that keeping Chartability on GitHub will inspire future improvements to address this gap in examples, and will address future limitations, as we discover them.

Chartability is a valuable tool for auditing. But we also hope that it can inspire researchers to:

1. Examine which heuristics (in our supplemental materials) could use more research attention, particularly those labeled "community practice."
2. Define constraints or requirements on novel projects, ensuring that new explorations still respect established standards, mitigating ethical risks.
3. Explore the intersections of disability in ways yet unaddressed in standards, such as the strong overlaps between understandability and operability (like keyboard navigation patterns across a data structure) or conflicts in understandability and flexibility (how some users need redundant encodings on charts while others find this overwhelming).
4. Consider access barriers in data experiences beyond those related to visual perception.
5. Engage the relationship between labor and access in computing, such as developing more measurements that demonstrate the imbalance of time and effort expected of users with disabilities (even in systems considered to provide "equal" access) and ways to evaluate who is contributing to accessibility efforts in a project (core team members, contractors, or volunteers).

We also want to caution researchers who are considering developing heuristics or auditing tools for use in practitioner environments to consider the tradeoffs between evaluation in rich, authentic professional settings and concerns such as intellectual property and corporate branding. We were able to apply our work in rich and collaborative practitioner settings because we were permissive with our potential participants. However, much of this work exists behind closed doors, similar to the downsides of industry research settings. More work may need to be done in order to encourage rich, cross-industry research projects, such as helping to anonymize the content of intellectual property and not just participants, while retaining data and findings.

4.10 Conclusion

The demand for accessible data experiences is long overdue. The Web Accessibility Initiative's (WAI) Web Content Accessibility Guidelines (WCAG) are over 22 years old and yet little work has been done to synthesize this large body of existing accessibility standards with research and inclusive design principles relevant to the fields of data communication, data science, data analysis, and visualization. Chartability begins to address unique accessibility best practice gaps in these domains with specific heuristics. This synthesis is meant to empower researchers, analysts, designers, developers, editors, and accessibility specialists with a framework to audit the accessibility of data experiences, interfaces, and systems to produce more inclusive environments for users with disabilities. The goal of Chartability is to make this work easier in order to encourage practitioners to regard current practices and resources, some of which have existed for decades.

In addition, Chartability opens the door to more work that remains to be explored in this space. Additional research is needed into many of the topic areas within Chartability's heuristic principles (POUR+CAF) as well as resources, examples, and tools provided for practitioners to perform this work more confidently and efficiently.

The changing landscape of visualization techniques and alternative interfaces (such as sonification and dynamic tactile graphics) may increase the demands for accessibility considerations in this space. The growing technological divide will become an even greater human rights issue as time moves on and we believe that tools like Chartability are necessary for the community of data practitioners to ensure they are including people with disabilities.

4.11 Postface

This chapter could easily produce the longest postface, and arguably the rest of this thesis *is* the postface following *Chartability*. But for the archival record, I (Frank) want to simply add that this project has afforded immense opportunities for future work (and still, even presently many threads are emerging).

On the one hand, *Chartability* was intended to be an in-depth set of heuristics that anyone could use to evaluate accessibility barriers in interactive data experiences. But on the other hand, the level of depth may also have introduced an unexpected outcome: many companies and organizations are interested in simply hiring an expert to help them remediate their systems. I wonder: "is it *too* deep?" But this unexpected outcome has been great for hands-on collaboration, funding opportunities, and for generating the focus areas of this thesis. However, it also points to a larger reality that many teams with the right motivations may simply be under-resourced.

Only a few practitioners in the world specialize in evaluating data visualizations and interactive data experiences exclusively, myself (Frank) being one of them. I helped to recruit among this group, led by Jonathan Schwabish, where we co-authored a report to better democratize the deeper knowledge required to do this work professionally [250]. While that project and my many speaking opportunities (nearly 100 focused on accessibility and visualization, largely thanks to *Chartability*) have been successful in helping to educate practitioners, we believe that significant gaps still remain. Practitioners need more, from somewhere, in order to feel independently empowered enough to begin, and gain expertise in, accessibility work in the context of interactive data experiences. Accessibility work itself still remains largely inaccessible.

And a wide range of recent work has begun to try to use *Chartability* as a prompt or design guardrail for agentic skills, or large-language model based or otherwise intelligent/automatic systems [49, 110, 182, 194, 219, 252]. The motivation here, as we interpret it, is to make *Chartability* more accessible to systems and automated or semi-automated design work. We intend to watch this work closely, because on one hand our goal for this project has been to empower practitioners themselves to make better decisions while they work. But the fundamental way that agentic systems operate is in delegation of agency and decision-making. Perhaps *Chartability*'s future lies in two spaces: narrow, but exceptionally deep human expertise in evaluation, or as guidance for automated systems whose context window has greater working-memory capabilities than an average interface designer or engineer.

However, the successful cases where *Chartability* has improved have been through socialization of the practice: shared expertise, working groups, collaborations, joint ventures, grant opportunities, talks, workshops, and more. Perhaps, as we unpack more in the conclusion of this thesis, what *Chartability* still requires lies upstream, not downstream or laterally: that policy, budgets, priorities, culture, and practice might be the new frontier for the success of this project.

Part III

Navigation: Making Data Structures Traversable

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

5.1 Preface

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.

5.1.1 Context

Data Navigator came about as a project largely motivated by my (Frank's) work in industry, at Visa working to make *Visa Chart Components* [295]. We had a top-down imperative to make our design system library of charts as accessible as possible, even pushing the limits of what that actually means. I led these efforts. It was clear to me from this work that one of the major technical barriers was enabling screen reader and assistive technology navigation of our charts, which is especially important when our charts were configured by our developer-users to have interactivity enabled.

At Visa, we rendered our charts using SVG (scalable vector graphics) in the browser. And it is possible to make SVG navigable to a screen reader by adding ARIA [301]. However, this approach only works for screen readers, meaning someone else who leverages a navigation-based assistive technology (such as a sip-and-puff, switch, keyboard, and so on) wouldn't have access. This was an incredibly difficult problem for us to solve at the time, especially since even ARIA's basic support and adoption varied significantly across devices and browsers at the time. Instead, I decided that we would programmatically render accessible, native HTML above our charts as someone navigated. For our uses at Visa, this approach worked.

Our approach had three major issues that blocked broader adoption: first, our work was tightly coupled to our internal visualization engine that rendered our chart components. It expected SVG. Next, the interactivity we built was *only* capable of perfectly mirroring whatever

interactions we had already built into our chart components for clicking and hovering actions. This limited which interactions were possible. And lastly, our structure wasn't grounded in any theory of data structures, but rather simply coupled to a navigation pattern we applied to nearly every chart in our chart library (with a few exceptions). And the exceptions in our library (for example, a circle-packing chart) made me realize we needed something about our navigation pattern to change fundamentally at a low-level to allow for broader experiences. Adding new features and refactoring existing ones was a brittle, slow experience.

For this project, we started nearly two years after I left Visa. Those lessons still sat in my mind. A better system was possible. I wanted to create a general system that any library rendering charts could use or replicate. Not to spoil one of the major intellectual foundations for this chapter, but we decided to use a graph as the substrate that scaffolds all other structures we might want for discrete navigation. By using a graph and creating a generic and low-level API, we created a tool that could express *virtually any structure*. Of course, we also simply de-coupled our structure from rendering methods entirely as well as de-coupled our interactivity from any set of assumptions about which interactions are possible for end users. These changes enabled *Data Navigator* to become what I like to call a set of "LEGO bricks" with enormous potential for solving existing problems and enabling new designs.

At the end of this chapter, we do a few things for the sake of this thesis: discuss the project in more technical detail and reflect on the successes that followed this publication.

This chapter was adapted from our 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.

5.2 Overview

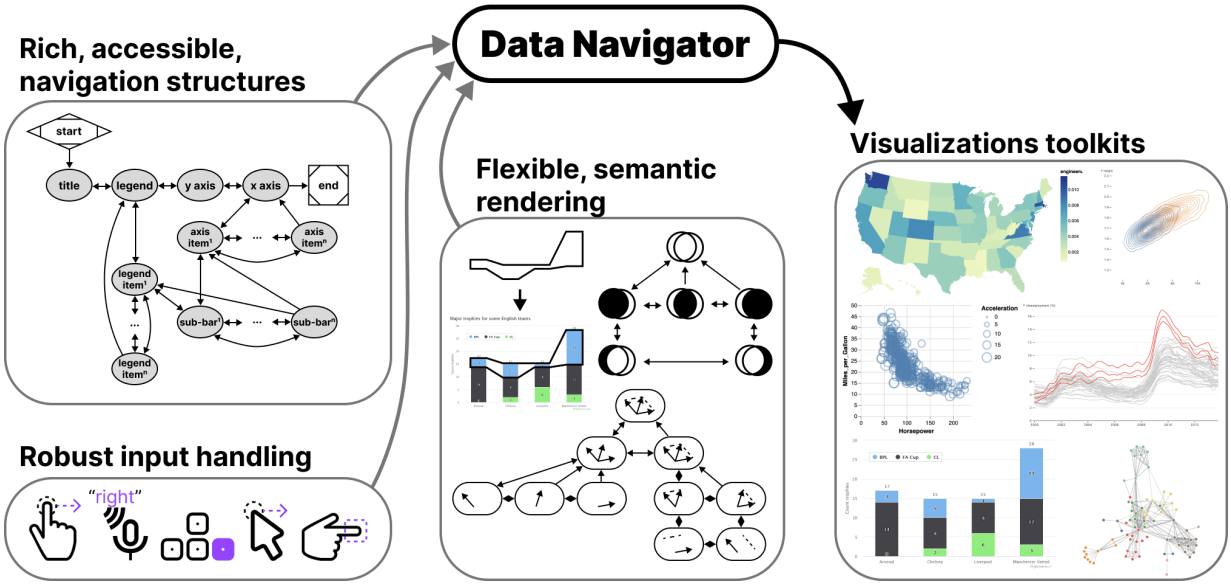


Figure 5.1: Data Navigator provides data visualization libraries and toolkits with accessible data navigation structures, robust input handling, and flexible semantic rendering capabilities.

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.

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 [102, 154, 256, 271, 289, 332].

Some emerging approaches have sought to address this serial limitation of data navigation and provide richer experiences for screen reader users [102, 271, 289, 332]. 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 [332]. 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 [289, 332] 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 [76, 82]. Marriott et al. also found that visual disability considerations are the primary focus of data visualization literature [191], 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 [332].

Chart Reader also engaged these design space considerations in their co-design work on accessible data navigation structures [289]. 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 [102, 255, 257, 271]. 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 [50, 82, 177, 256]. Jung et al. offer guidance to consider the order of information in textual descriptions and during navigation [154]. Kim et al. collected screen reader users' questions when interacting with data visualizations, which could open the door for more natural language data interaction [160].

5.3.1.3 Accessibility standards and guidelines

In the space of research, there has been a growing interest in developing guidelines for practitioners [71, 76] and even applying guidelines as a method of validation alongside human studies evaluations and co-design [82, 177, 178, 332]. 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 [300]. 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 [301]. 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.

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 [332]. Prior research and industry work, with the exception of the *Visa Chart Components* library [295], 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

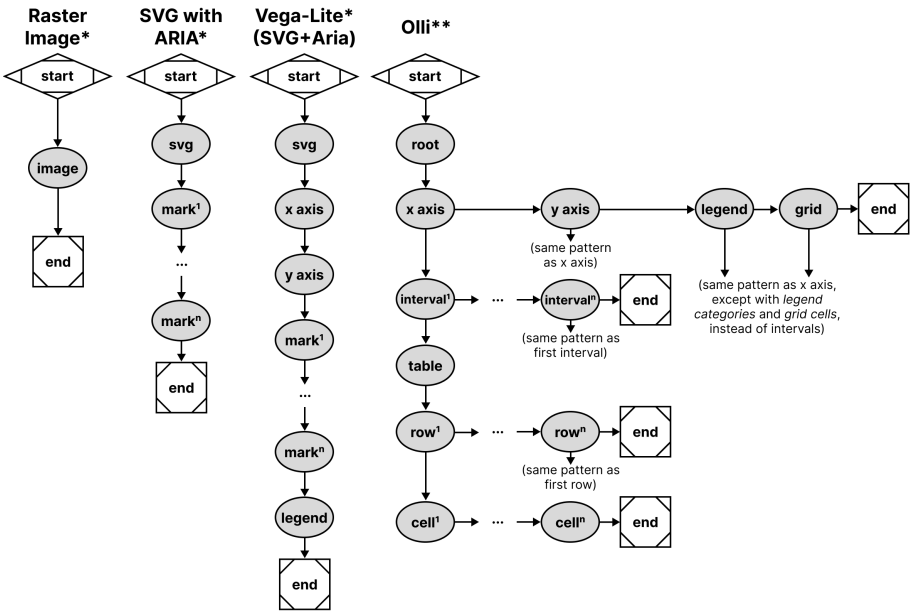


Figure 5.2: Existing accessibility trees and lists, shown using node-edge graph conventions. (*) Denotes only *screen reader* access. (**) Denotes *screen reader*, *keyboard-only*, and *pointer* access as well.

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* [130], *Visa Chart Components*, and *Progressive Accessibility Solutions'* visualization toolkits [102, 271] 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* [241] 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 [30]. 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 [241]. 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 [191].

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 [265], wearables [239], or DIY making and fabrication [139].

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 [172], 3D printed materials for rehabilitation professionals [99], and even using produce-based input (such as bananas and cucumbers) for aging populations [228].

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* [179].

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*) is 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 [94].

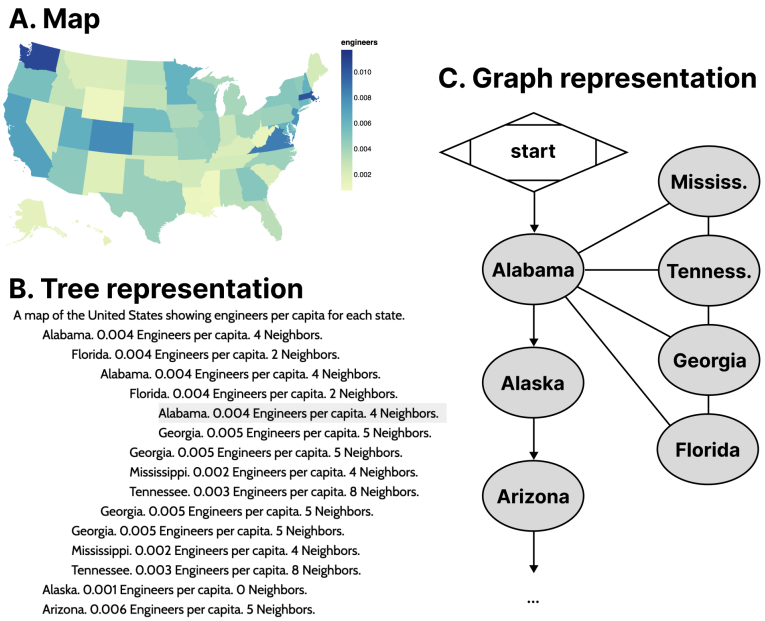


Figure 5.3: **A.** Map of engineers per capita of US states. **B.** Tree representation of the map data where states are listed alphabetically and also include links to neighboring states. The structure repeats itself if users navigate in a loop. **C.** Graph representation with the same navigation potential 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 [289, 332] 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 is used in conjunction with 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

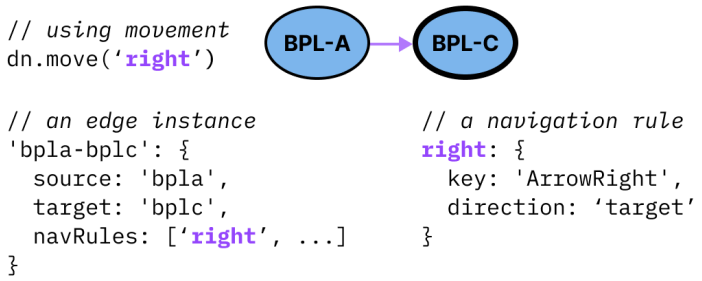


Figure 5.4: An example of how a single edge instance references a navigation rule and can even have multiple navigation rules. A navigation rule can be referenced by multiple edges.

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.

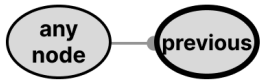
Generic functions for source or target properties can link nodes to other nodes based on changing content, structure, or behavior that may be difficult or impossible to determine before a user navigates the structure.

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.

```
// using movement
dn.move('previous position')
```




```
// a generic edge
'any-return': {
  source: (_d, current, _p) => current,
  target: (_d, _c, previous) => previous,
  navRules: ['previous position', ...]
}
```

```
// a navigation rule
'previous position': {
  key: 'Period',
  direction: 'target'
}
```

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

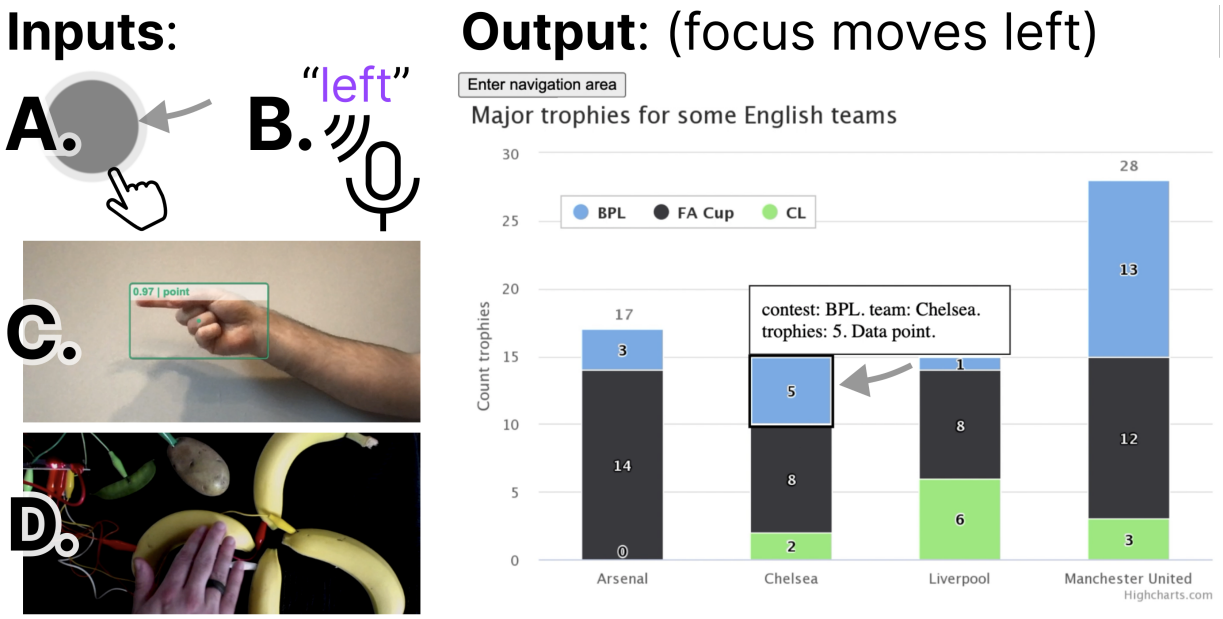


Figure 5.6: An example navigation rule to move "left" can be called as a method by an event from any input modality. Some examples include common modalities such as touch swiping (**A**) or speaking "left" (**B**). This also includes advanced or future modalities such as gesture recognition (**C**) or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

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.

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 as 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 [228], 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 [302]. Desktop 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.

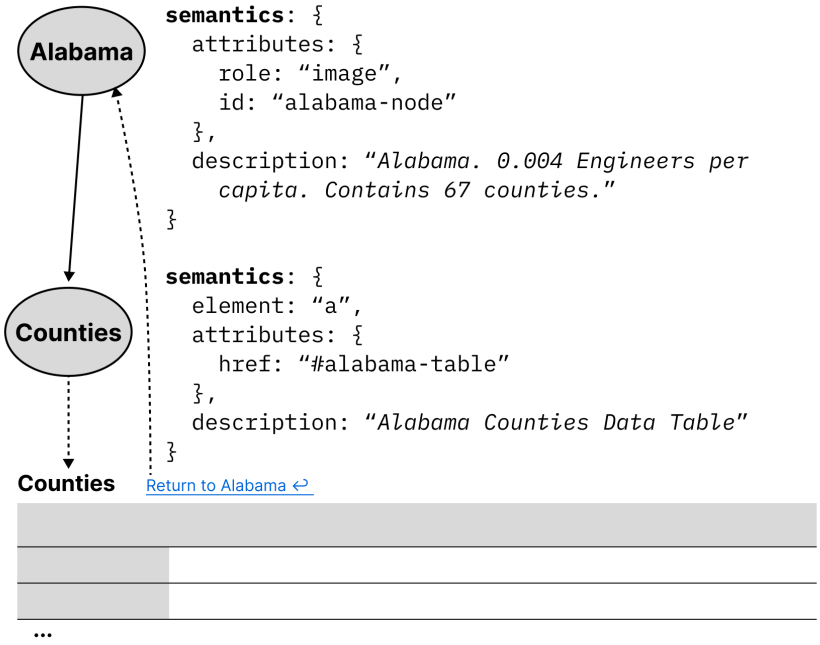


Figure 5.7: An example of how navigation within Data Navigator could use semantic nodes as hyperlinks to provide access to other areas in an application. Alabama has a child node “Counties” which is a semantic HTML link element pointing to a table of counties, outside of Data Navigator’s graph structure.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

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

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 [241] or list items nested within lists [25]. But in Data Navigator, nodes can semantically be buttons, links, or any HTML element. Interactive data visualizations sometimes demand more

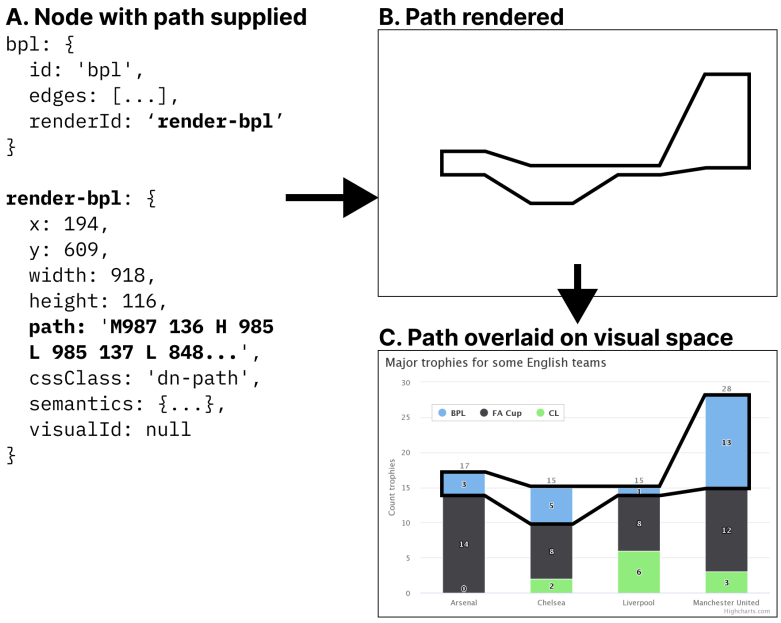


Figure 5.8: **A.** The data specified for a node with a reference to separate data that is used to render that node. **B.** The node will render as a path at the specified Cartesian coordinates. **C.** This rendered node may then be placed over a visual.

flexible node semantics than geometries or lists.

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 [299].

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 [25], 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

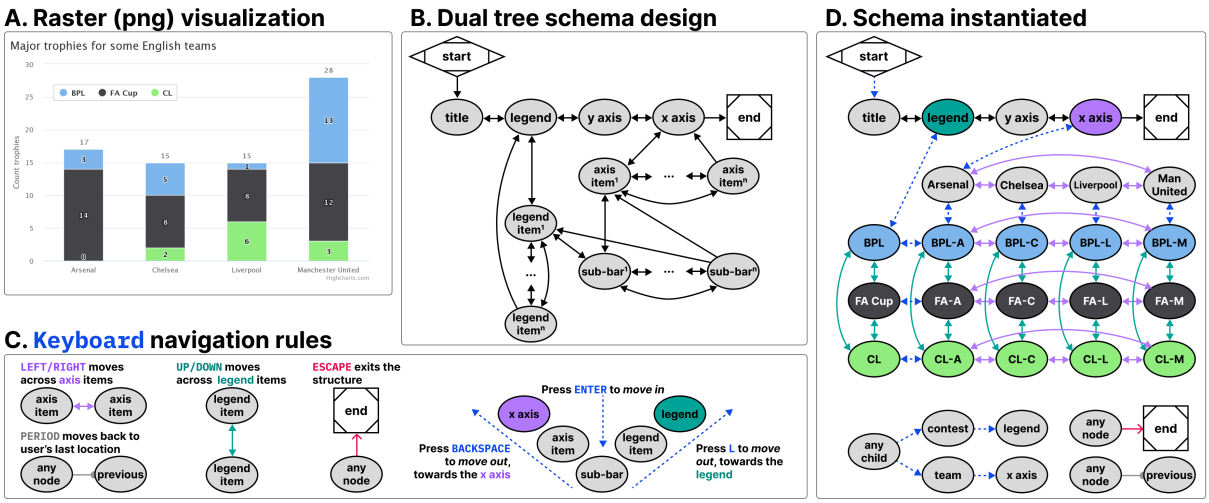


Figure 5.9: **A.** A raster (png) visualization of a stacked bar chart showing how 4 English teams performed across 3 major trophy contests. **B.** An example navigation schema that allows children nodes to have 2 parents (two tree structures intersecting), one for contests and one for teams. **C.** 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 an 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.

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 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 [25, 73, 332].

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 [131]. 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 [129]. 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* [27] 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* [307]. It is important for any technology developed to also meet minimum requirements for accessibility [76, 177, 178, 332], even when following best-practices and research.

5.5.2 Building Data Navigation for a Toolkit Ecosystem

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

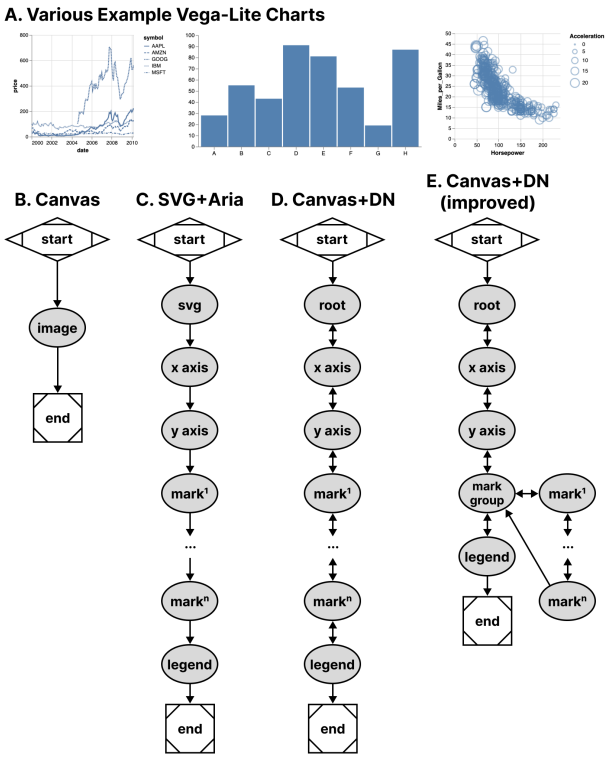


Figure 5.10: **A.** Various charts from *Vega-Lite* share the same general structures with each other when rendered using canvas (**B**) or SVG (**C**). **D.** With Data Navigator, we replicated the existing SVG navigation pattern (**C**) but used a canvas-based rendering for the visualization. **E.**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

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 [263, 332].

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 [241].

Data Navigator allows toolkit 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 $\sim$0.45 to $\sim$1.5ms respectively. Our extraction functions specific to *Vega-Lite* increase initialization between $\sim$4.8 and $\sim$8.5ms respectively. Given that our benchmark testing for *Vega-Lite's* SVG rendering initialized in $\sim$1,800ms for 20,300 data points and canvas in $\sim$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

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 [177, 178, 289, 332].

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

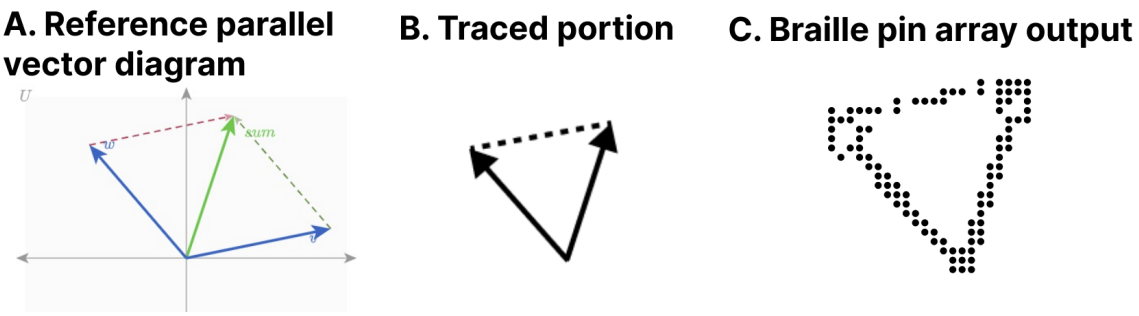


Figure 5.11: Our material preparation process involved taking a reference (**A**), tracing it (**B**), and rendering it on a tactile display (**C**).

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 a 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

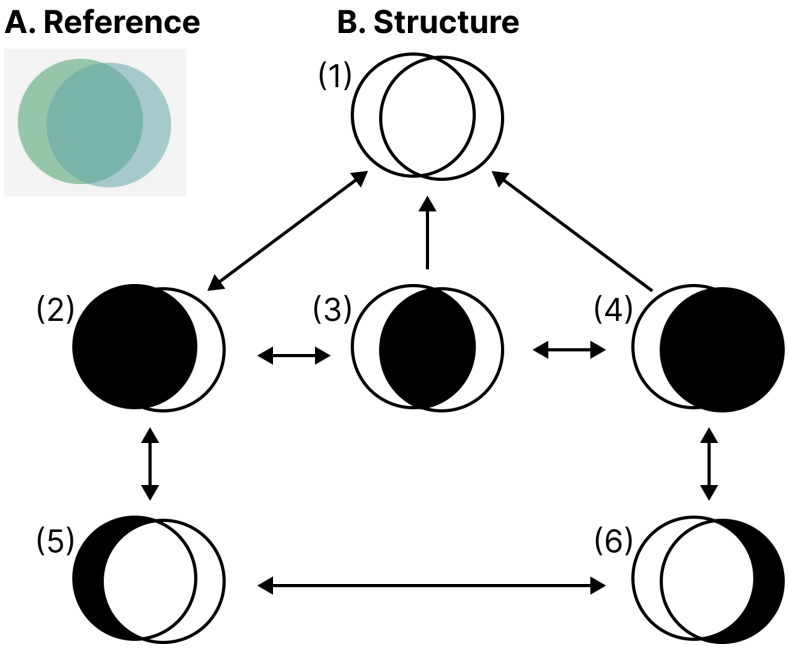


Figure 5.12: **A.** A reference image from *Penrose* of a set diagram containing two sets intersecting. **B.** A diagram of our proposed structure, with three levels of information.

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* [327].

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 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 supplements them.

At our disposal for our co-design session was a *Dot Pad* [66], 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. [59], 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 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

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.

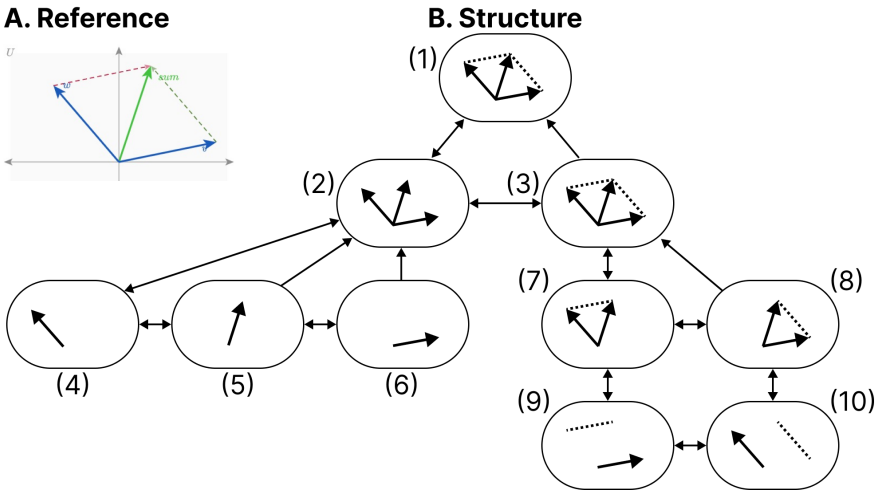


Figure 5.13: **A.** A reference image from *Penrose* of a parallel vectors diagram. **B.** A diagram of our proposed structure, with two main sub-categories of information: understanding the vectors and their parallels.

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 [67] and adaptation [320] in different applied contexts. It is, as Louridas writes, a *technical material*: a technology that enables new and useful capabilities [176]. 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 [177, 178, 226, 332]. 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 [76, 161, 191]. 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 [179]. 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.

5.8 Postface

We unpack this in the next chapter, but our work on *Data Navigator* was never intended to end as a research prototype. We carefully engineered *Data Navigator* to focus on what we believed would be the most-necessary modules required for integrating into any web-based visualization library. Modern libraries render in many different ways, as SVG, HTML, canvas elements, and canvas built with WebGL under the hood. The actual work of rendering is offloaded onto a wide range of ecosystems, from React, to Svelte, to D3, to vanilla JS. Our *rendering* module was intended to work with any of it (or be entirely optional). This, we believed, separated us from every other research prototype the most. *Olli*, while a fantastic project, didn't have an independent rendering engine that could de-couple from the data structures it produces, same with *VoxLens* and other similar projects. These projects handle their own rendering: they produce elements that live on the DOM. Infrastructurally, this can be a problem for many teams when integrating into their production environments because the structure *is* the rendering.

While it certainly isn't hard to imagine how to make a de-coupling possible with those existing research prototypes, doing that work after their papers have been published is typically an additional burden on a research team that isn't actively incentivized. Maintenance and feature improvements are hard to justify when novelty drives technical research production most of all. These systems ultimately must be refactored to work with different rendering methods (SVG/canvas, etc) and rendering frameworks (React, D3, and so on). We intended to face this speedbump from the start and mitigate future refactoring as much as possible.

And in our next chapter, our work with *Adobe* wouldn't have been possible without this de-coupling approach with our rendering module. Coincidentally, the main approach that libraries like *Olli* and *VoxLens* must take is building couplings, wrappers, and pipelines between a separate space where their tech exists and a visualization exists. The spaces aren't integrated neatly. Accessibility exists in one area of an interface while visualizations are in another, and they must be managed separately. But in our work with *Adobe*, *Data Navigator* is part of the lowest-level processes that construct a chart experience, working in tandem with the specifications for their visualization engine. *Bokeh*'s ecosystem, on the other hand, made more sense to remain de-coupled to avoid placing future maintenance in the hands of the open-source *Bokeh* contributors.

And we didn't invent this notion. Vega technically just takes a specification and it produces a view model. How that model is actually turned into visible elements is up to the ecosystem [241]. But for accessibility, there tends to be a philosophy to tightly couple the rendering methods to the access model. We conjecture that this might simply be because many accessibility barriers are a result of an ecosystem's failure to use the right substrate. So an accessible software system *is* one that writes to a substrate.

What we might learn from *Data Navigator* then is that more accessibility projects should seek to abstract out the generation of accessible models and substrates from the means to enact those substrates. The precedent for this in visualization has empowered a single spec like Vega to integrate into the web [241], Python [294], and Rust [165], to name a few. And while *Data Navigator* was instantiated as a JavaScript library, the abstraction of the system as it was designed could lead to implementations in virtually any software environment.

Additionally, because this thesis can afford additional space, we are expanding here beyond the already-published chapter and unpacking some more details about *Data Navigator*, techni-

cally speaking. Every subsequent chapter will also have additional technical details included (beyond their published papers), to help archive and document more about how these systems were made.

5.8.1 The library boundary: what ships, and what demos build

Before describing each subsystem, it is worth being precise about what Data Navigator the *library* provides and what our case examples build on top of it. Previously, this chapter differentiates between the system and the instantiation of our system. Here, we refer to the current-working instantiation of Data Navigator as our *library*. Data Navigator's library is distributed as an npm package, `data-navigator`, written in TypeScript and compiled to both ES module and CommonJS formats with type declarations. Installing the package provides three composable modules, one for each of our design considerations: `dn.structure()`, `dn.input()`, and `dn.rendering()`. These can be used independently or in tandem. The package also ships a small set of supporting utilities, including `describeNode()` for generating default text descriptions from a datum, `createValidId()` for sanitizing identifiers, and a collection of pure geometry functions (convex hulls, unions of rectangles, and bounding outlines) used to compute focus outline paths for groups of elements.

The three modules divide responsibility as follows. The *structure* module produces the graph: given either hand-authored node and edge objects, a flat dataset with a declared set of dimensions, or a *Vega-Lite* view object, it returns nodes, edges, and the navigation rules that edges reference. The *input* module is a pure traversal engine: given a structure and its rules, it answers the question "from this node, where does this navigation command lead to in the structure?" The *rendering* module maintains a semantic HTML layer: a wrapper with appropriate ARIA properties and focusable elements positioned over (or beside, or independent of) the host visualization. The implementation details of each module are described at the end of their respective subsections below.

Just as important is what the library deliberately does *not* do. Data Navigator does not render charts, does not fetch or transform data beyond building its graph, and does not attach global event listeners to the page. It never decides *when* to move; it only resolves *where* a validated navigation command leads and renders what it is asked to render. This restraint is the practical meaning of the "technical material" framing discussed previously at the end of the chapter: like concrete, the library has no opinion about the building.

Our case examples therefore each supply four kinds of application code, and the division of labor is summarized in Figure 5.14. First, the structure itself: in our first and third case examples, every node, edge, and rule was authored by hand, while our second generated them from a *Vega-Lite* view. Second, modality adapters: the keyboard listener, the swipe recognizer, the hand gesture model, the speech recognition grammar, and the text command parser in our first case example are all demo code, and each one reduces an input event to the name of a navigation rule before handing control to the library. (This means that the application level does most of the hard work, parsing a gesture's swipe into the text string, "left" or equivalent.) Third, the focus lifecycle: when the input module returns the next node, the application renders it, focuses it, and removes the previously focused element. Fourth, coupling to the host visualization, such as highlighting the chart region that corresponds to the focused node.

A reader asking "what exactly do I get if I `npm install data-navigator`?" gets the graph builders, the traversal engine, the rendering layer, and the utilities; everything else in our demonstrations is replaceable application code, which is precisely the point.

The package is developed in a public monorepo that has continued to grow since the original publication of this work. At the time of writing it also hosts companion packages built on the core library, including `@data-navigator/inspector`, a visual graph debugger for navigation structures, and the *Skeleton* authoring application. We return to these in the following chapter, since this chapter concerns the core library.

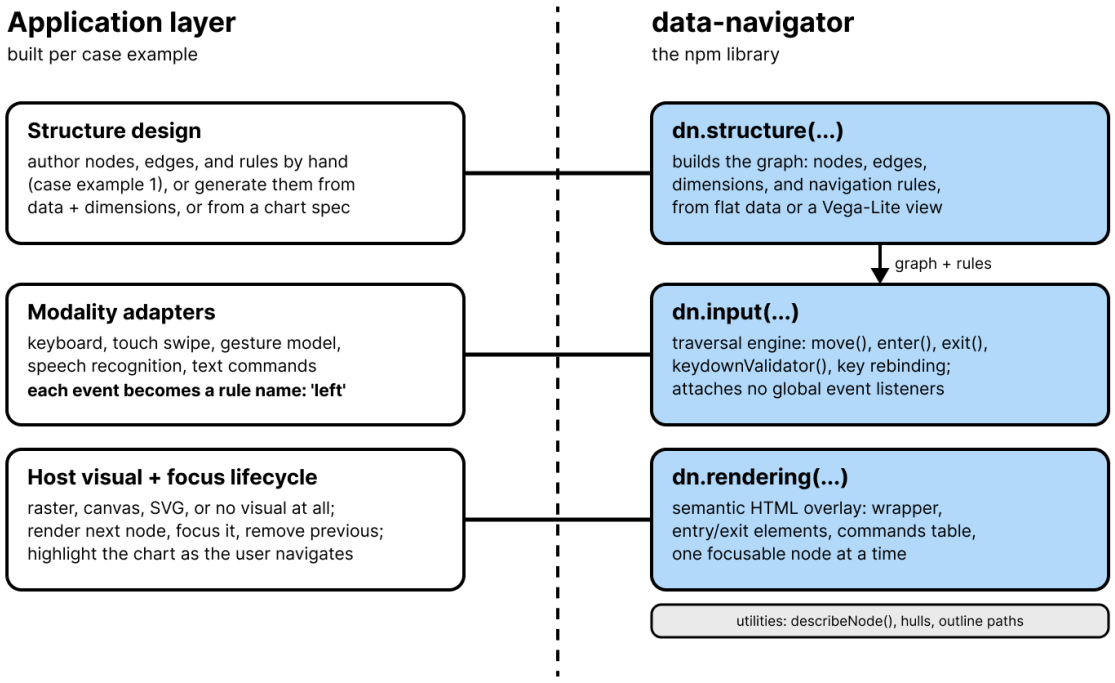


Figure 5.14: The library boundary in Data Navigator. The npm package (right) provides the graph builders, the traversal engine, the rendering layer, and supporting utilities. Each case example (left) supplies the structure design, the modality adapters that translate input events into navigation rule names, the focus lifecycle, and any coupling to the host visualization.

5.8.1.1 The structure module in implementation

In the implementation, a structure is a plain object containing `nodes`, `edges`, and `navigationRules`, plus optional `dimensions` and `elementData` (rendering properties keyed separately from the graph, so that structure and rendering remain independently specifiable). Nodes and edges are dictionaries keyed by identifier. A node object holds its `id` and an ordered list of edge *identifiers*, not edge objects; because nodes reference edges by id, a single edge object can appear in many nodes' lists, which is what makes the generic edges described above cheap. An edge object holds `source`, `target`, and `navigationRules`, the list of rule names it participates

in, where either endpoint may be a function of the current datum and current focus. A navigation rule object holds its orientation (`source` or `target`) and, optionally, a keyboard key.

Developers can author these objects entirely by hand, as in our first case example. The module's builders exist to remove that tedium when the data allows it. Passing `dataType: 'vega-lite'` instead dispatches to the *Vega-Lite* ingestion path used in our second case example, which walks a *Vega-Lite* view's scenegraph and emits nodes, edges, spatial properties, and descriptions, subject to developer-supplied inclusion criteria and an optional describer function.

The declarative grammar for dimensional scaffolding, and an authoring interface built on top of it, was not actually part of this chapter's work despite being in the library at the time of writing this postface. These are unpacked in the following chapter for *Skeleton*.

5.8.1.2 The input module in implementation

The input module is deliberately the smallest of the three: roughly a hundred lines that implement traversal and nothing else. `dn.input()` takes a structure, its navigation rules, and an entry and exit point, and returns a handler exposing `enter()`, `exit()`, `move(current, direction)`, `moveTo(id)`, `keydownValidator(event)`, `focus(renderId)`, and `setNavigationKeyBindings()`. The core, `move()`, implements the traversal function described in the structure subsection: it walks the current node's edge list in order, finds the first edge whose rule list contains the requested direction name, resolves the edge's endpoints (invoking them if they are functions), and returns the node at the rule's orientation, provided it is not the node the user is already on.

Two design choices here carry the input-agnosticism argument made above. First, `move()` takes a rule *name*, a plain string like `'left'` or `'drill-in'`, so any code that can produce a string can drive navigation; the swipe, gesture, speech, and text adapters in our first case example each end in exactly such a call. Second, the keyboard itself is handled the same way as every other modality: `keydownValidator()` is a pure mapping from a keyboard event code to a rule name (by default the arrow keys map to the four directions, Enter to drill-in, Escape to drill-out, and comma and period to backward and forward), and the bindings can be replaced wholesale through `setNavigationKeyBindings()`, which is the hook for end-user remapping mentioned above. The library attaches no global event listeners of its own; the host application listens for events where it sees fit, validates them, and forwards a direction. The library decides where a command leads, never when one occurs.

5.8.1.3 The rendering module in implementation

The rendering module maintains the semantic HTML layer. `dn.rendering()` is initialized with a root element and returns a renderer whose `initialize()` builds the scaffolding around the structure: a wrapper with `role="application"`, an accessible name, and a managed `aria-activedescendant` that tracks the focused node; an optional entry button ("Enter navigation area") that gives keyboard and screen reader users a discoverable way in; an optional exit element announced as the end of the data structure; and an optional, automatically generated commands table that lists every available navigation command and its key, derived directly from the structure's rules so that documentation cannot drift from behavior.

The renderer's `render()` then creates exactly one focusable element for a given node: an outer element whose tag, role, and attributes come from the node's parent semantics, an inner element carrying the required accessible label, a `tabindex` for focusability, and absolute positioning from the node's spatial properties. If a node specifies a path, the renderer additionally draws an SVG outline, which is how the bespoke cross-stack focus indication in Figure 5.8 is produced; the geometry utilities shipped with the package (convex hulls, unions of rectangles, and bounding outlines) exist to compute such paths for derived group nodes. Every rendering property is dynamic in the same sense as edge endpoints: each may be a static value or a function of the element's render data and datum, resolved at render time against a developer-supplied set of defaults. The renderer also enforces one hard accessibility rule: if a node reaches `render()` without a label, the library logs an explicit accessibility error rather than silently producing an unnamed focusable element. The complementary `remove()` and `clearStructure()` methods complete the on-demand lifecycle described above; in implementation, the render-then-remove swap is a convention enacted by the host application's focus lifecycle, which keeps the library free of assumptions about when an application wants elements to persist (as in pre-rendering) or disappear.

5.8.1.4 Graph-theoretic foundations

Having introduced nodes, edges, and navigation rules informally, we can now state what a Data Navigator structure formally is, because several of the system's guarantees follow directly from this formulation. A structure is a labeled multigraph $G = (V, E, R)$: a set of nodes V, a set of edges E where each edge carries a source endpoint, a target endpoint, and a non-empty set of labels drawn from R, and a set of navigation rules R where each rule has an orientation, either *source* or *target*. Endpoints may be node identifiers or functions that resolve to node identifiers at traversal time. Each node holds an ordered list of the edges that involve it. Navigation is then a partial function $move : V \times R \rightharpoonup V$: at node v, given rule r, the system scans v's edge list in order, takes the first edge labeled with r whose resolved endpoint in r's orientation differs from v, and moves there. If no such edge exists, the command is simply inert at that node.

The first is *determinism under multiplicity*. Because G is a multigraph, nothing prevents two edges at the same node from sharing a rule label; the ordered edge list resolves this by acting as a priority order, so every navigation command has exactly one outcome at every node. This is what lets designers attach broadly shared generic edges (such as the undo edge described above) to every node without worrying that they will shadow, or be shadowed by, more specific edges in unpredictable ways: precedence is explicit in the list order.

The second is *symmetry, and therefore undo-ability*. By default a single edge object serves a *pair* of rules with opposite orientations: `left` moves toward an edge's source while `right` moves toward its target, and the dimensional scaffolding described below always assigns rules in such pairs (`sibling_sibling` pairs like left and right or up and down, and `parent_child` pairs like drill-in and drill-out). The consequence is that every traversal step has an inverse along the same edge: a user who presses a key can press its partner and return to exactly where they were. Paths through the structure are reversible by construction, which means users can explore without memorizing routes, an essential property for non-visual navigation where there is no persistent overview to glance back at. The library also allows this symmetry to be deliberately

broken: the `useDirectedEdges` option reorients every rule toward its edge's target, producing one-way movement for genuinely directed structures such as flows and processes. Even then, reversibility can be restored globally by the generic return edge, a single shared edge whose endpoints resolve dynamically to the current and previous nodes, giving users an undo across any jump, directed or not.

The third is *reachability as a design property*. A structure is only as usable as the set of nodes reachable from its entry point, and the exit must be reachable from everywhere. The latter is cheap to guarantee in this formulation: one shared generic exit edge attached to every node makes the exit reachable in a single step from anywhere, which is how our case examples implement it. Reachability of the rest of the graph, however, is not statically guaranteed, precisely because endpoints may be functions: the realized graph is partially lazy, materialized edge by edge as users navigate. Expressiveness and verifiability trade off here. A designer hand-authoring edges can strand a node with no incoming edges, and no compiler will object. This is a genuine cost of the approach, and it is part of what motivated the inspection tooling presented in the following chapter, which renders a structure's graph so that designers can see disconnected or one-way regions before users encounter them.

Cycles in this formulation deserve a brief note, because they are the clearest illustration of why a graph substrate is the right choice. In the rendered tree of the bordering-states example (Figure 5.3), the cyclical relationships between neighboring states had to be materialized as repeated subtrees, producing redundancy that grows with the data. In a graph, a cycle is just a set of edges. Later in our *Dimensions API* explanation for *Skeleton* (next chapter), Data Navigator treats cycles as a controlled design dimension: each dimension's boundary behavior is declared through its `extents`, which determine whether sibling traversal wraps in a cycle (`circular`), terminates at the ends (`terminal`), or bridges out of the current division and into a neighboring one, skipping empty divisions and wrapping if needed (`bridgedCousins`), or into an explicitly chosen node (`bridgedCustom`). Whether a user's repeated keypress orbits a category, stops at its edge, or carries them into the next group is a one-word declaration.

Finally, the multi-dimensional structures shown earlier (Figure 5.9) have a precise reading in this formulation. Each declared dimension contributes a small rooted hierarchy over the *same* underlying data nodes: an optional shared root (level 0), the dimension's own node (level 1), its divisions (level 2), and the data nodes themselves (level 3). One tree over a shared leaf set is just a tree; the union of k of them is a graph in which each leaf has k parents, which is exactly the "two intersecting trees" of our first case example. Because every dimension is assigned its own pair of sibling rules, movement decomposes into independent axes: one pair of keys traverses siblings within one dimension while another pair traverses siblings within a second, all from the same node. The cost model is also worth stating: a move inspects only the current node's edge list, so each navigation step is proportional to that node's degree and the rules per edge, independent of $|V|$; storage is linear in nodes plus edges, with rule objects and generic edges shared rather than duplicated. The graph does not merely represent more structures than a tree, as argued above; it represents the same multi-dimensional structures more compactly and traverses them in constant local work.

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

6.1 Preface

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 re-purposing 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.

6.1.1 Context

Data Navigator was successful enough to help us acquire practitioner-centered funding to apply it in real-world contexts. We worked with three different communities as a researcher-designer-engineer alongside the teams attempting to apply *Data Navigator* to their work. These projects, while ultimately responsible for *Skeleton* (spoiler alert), also quickly motivated a new hypothesis: that making data navigation experiences isn't just *hard* for some generic reasons but that it may

be hard because it involves perceptual accessibility barriers for sighted practitioners (or in other words: we conjectured that sighted practitioners face barriers when working with non-visual design materials).

Data Navigator evolved and grew and found a place in real-world work after its release. However, *Data Navigator* eventually also became a vehicle to open up new research directions. Notably, we made improvements to the system from a technical standpoint and also identified new barriers that practitioners faced when applying *Data Navigator*, both of which we discuss further in this chapter.

The following work was proposed as the culmination of this thesis, and chronologically was the final work we completed for this body of work.

This chapter has also, notably, been modified from the submitted version to IEEE VIS, for the sake of this thesis' overall narrative cohesion as well as the fact that this thesis can afford more space to expand in several areas beyond the page limits that VIS allows.

It is worth stating plainly, before the chapter proceeds, how we intend each artifact to be read. The two pieces of infrastructure described in later subsections, the *Inspector* and the *Dimensions API*, are engineering contributions in the ordinary sense: they are released, reusable, and already used in production-facing collaborations beyond this research. The integrated authoring environment we call *Skeleton* is something different. We built and studied it as a design probe [114, 142], an instrument whose purpose is to make a specific thing observable rather than to ship a finished workflow. What the probe is designed to make visible, up front, is twofold: first, the navigation structure itself, rendered as a manipulable visual object so that sighted practitioners can perceive and react to it; and second, the practitioners' own relationship to that structure, what they notice, question, and revise once it stops being invisible. The first is a property of the artifact; the second is what the study is built to surface.

This framing has consequences for what we claim and how we evaluate. A production authoring tool would be judged on throughput, robustness, export fidelity, and task completion time. A probe is judged on what it reveals. Accordingly, our study (Section 6.4 onward) reports qualitative shifts in how practitioners engage with accessible design, not task performance against a baseline, and the prototype deliberately leaves production concerns such as deployable export out of scope (see the Limitations and Future Work section). Treating *Skeleton* as a probe is also why we study it first with sighted practitioners alone: the probe is positioned to test whether making structure visible changes sighted authors' reasoning, which is a precondition for, not a substitute for, evaluation with the disabled users those structures ultimately serve. We return to the boundary between what the probe makes *designable* and what only end-user evaluation can establish as *good* in the Limitations and Future Work section.

This chapter was adapted from our paper, currently conditionally accepted 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.

6.2 Overview

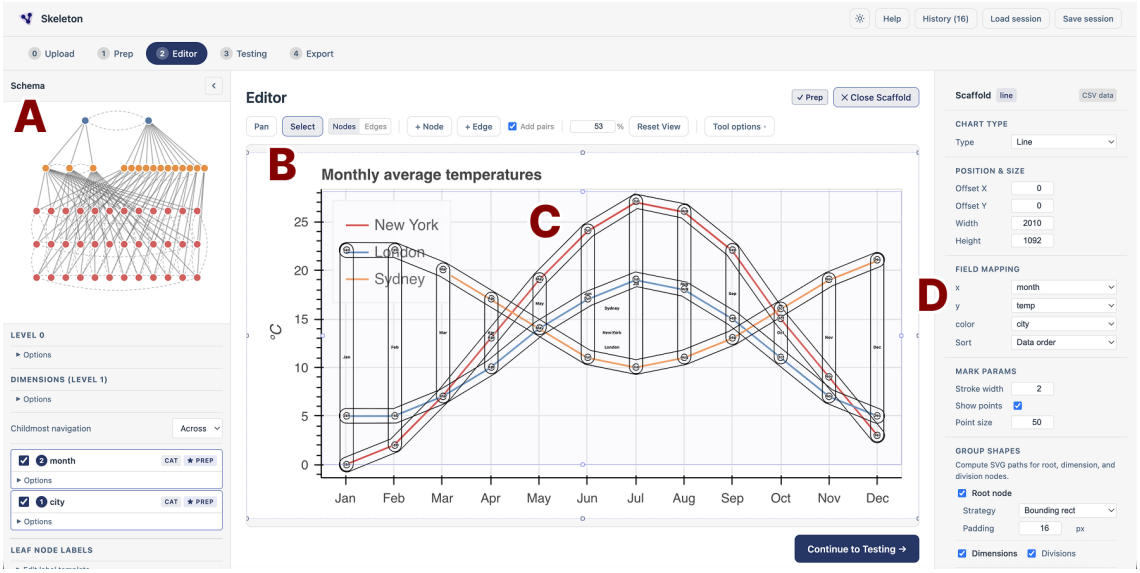


Figure 6.1: *Skeleton* being used to build a navigation structure over a line chart. **A**: a dual tree visualization of nodes and edges built for a line chart with 3 lines, spanning over 12 months. **B**: a direct-manipulation canvas that resides over a static png image of a line chart. **C**: a skeleton-like structure of nodes, edges, and group outlines that correspond to the schema shown in (A), overlaid on the visual space of the chart in (B). **D**: controls for rapidly scaffolding the nodes, edges, and their shapes shown in (C), using Vega's visualization engine for automatically generating spatial properties.

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 [197, 241, 310, 313, 333], direct-interaction interfaces [78, 141, 264], and iterative feedback loops between representation and understanding [85, 122, 173] 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.

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 iterate. 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 [153, 258]. 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 [121] 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 [77], 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 [114, 142] 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 Visualizing Non-visual Structure

Long before accessibility, computing established a practice of taking structures that are not inherently visual and giving them a visual form that can be seen and acted upon directly. Sutherland's Sketchpad turned the constraints and topology of a drawing into a manipulable graphical object rather than coordinates in memory [282]; visual programming environments rendered computation as arrangements of on-screen objects [270], syntax-directed editors treated a program as a hierarchical structure edited through its visible derivation tree [286], and structured flowcharts gave control flow a fixed spatial form [208]. Myers later named the distinction this lineage had been drawing, separating *visual programming* from *program visualization*, depicting otherwise non-visual programs graphically [207]. A consequence is easy to overlook: many structures we now treat as inherently visual, including trees, graphs, flowcharts, and the document object model, are abstractions whose visual rendering has become so dominant that the representation now stands in for the structure.

Direct manipulation interfaces, in turn, provide a continuous representation of the objects of interest and give rapid, incremental feedback, letting people work with a structure they would otherwise hold in their heads or reach only through code [141, 262]; grammars of interactive graphics extend the same move to authoring [27, 241]. Yet across this tradition, one structure has remained conspicuously without a visual form to manipulate: the UI structure that a screen-reader traverses. This lack of visual representation is especially surprising because screen readers evolved in parallel with the emergence of graphical interfaces [288].

6.3.2 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 [161, 191, 256]. 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 [154, 158, 162, 177]. 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 [75, 83, 160]. 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 [33, 125, 187], with declarative grammars emerging for authoring these experiences [159]. Haptic and tactile representations offer another channel through refreshable displays, 3D-printed graphics, and multimodal touchscreen interactions [38, 135, 192, 227]. Recent systems integrate multiple non-visual representations of the same data [50, 136, 253].

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 [77, 271, 302, 332]. 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 [289, 332]. Giving users control over the textual tokens surfaced during navigation improves comprehension and agency [151]. And more recent work has found that perceptually congruent navigation structures for charts and diagrams can improve goal-driven exploration [199].

6.3.3 Authoring Non-visual Data Experiences

Accessible visualization has historically centered the experiences of disabled users, but a parallel body of work examines the experiences of visualization practitioners working on accessibility.

Practitioner challenges. Research consistently finds that sighted visualization practitioners struggle with accessibility [82, 153, 258]. Most visualizations in the wild are inaccessible and designers themselves report lacking guidance, especially for complex and interactive graphics [153, 258]. 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 [82, 256]. 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 [153, 178, 258, 334]. 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 [162]. Of those that do, most rely on code-based approaches [25, 77, 255, 257]. *Umwelt* [334] 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* [25]. 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 [318]. 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" [332], *ChartReader* [289], *Benthic* [199], and *Data Navigator* [77] 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 [205].

Yet, most other non-visual modalities are not only communicated but *authored* in visual and non-visual modalities. Sonification is designed in editors that are both visual and auditory [159]. Tactile graphics are composed by transcribers on a graphical canvas before being physically embossed or printed [26].

Inspecting accessibility structure. 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 thus understood and communicated visually by sighted researchers and designers, yet built entirely without visual feedback by developers; this is the gap we address.

6.4 Co-design Foundations

Our research follows an *action research* orientation [121], 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 [77] 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, Nadolskis. CD1's contributions are noted in Section 6.4.1, CD2's are noted in Section 6.5.3. Nadolskis 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 design partners [57, 59, 224, 289, 331, 332], 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.

These themes did not arrive as abstractions; each project below surfaced them as concrete friction, and together the three engagements produced five design goals that *Skeleton* was built to satisfy. We state them here and, in the subsections that follow, mark where each emerged:

DG1 Make structure visible and inspectable. Render the navigation structure as a manipulable object during authoring, not only as code, so that practitioners can perceive and react to it.
DG2 Specify navigation at the level of data. Provide a higher-level, data-driven abstraction so practitioners describe traversal in terms of data dimensions rather than hand-wiring every node and edge.
DG3 Shorten the iteration loop. Let practitioners verify a design without a full code build or a manual screen-reader pass.
DG4 Support image-only and bespoke visualizations. Do not require a single underlying dataset, format, or recognizable chart type, since the cases that most need accessible tooling often have neither.
DG5 Reach beyond keyboard input. Accommodate input modalities other than assuming keyboard navigation, including touch and text.

6.4.1 Geologic Map

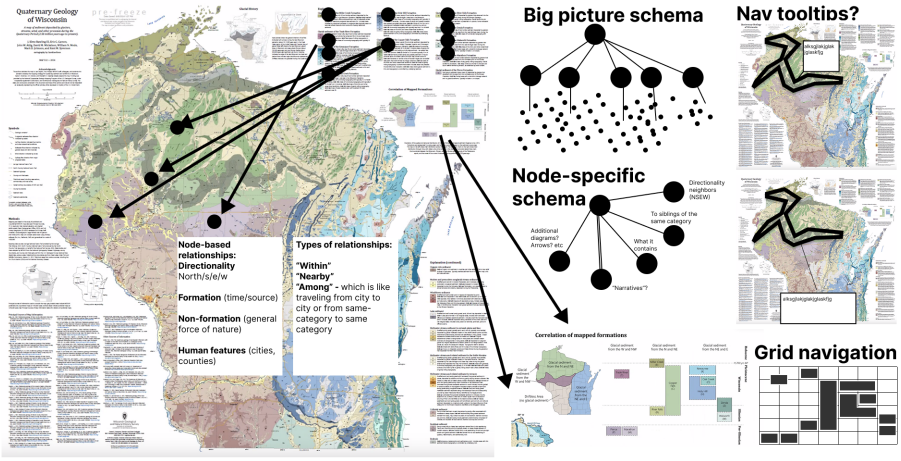


Figure 6.2: Our visual design work in Figma over a static geologic infographic map of Wisconsin. We use visual forms and illustrations over and beside the map to communicate flows, structure, navigation styles, and interaction patterns.

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 [225]. 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 [22] 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 (**DG1**). And the map's lack of a single tabular dataset made clear that a useful navigation tool should not assume structured data or a grammar of graphics (**DG4**).

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. Each of these frictions pointed somewhere: the dimension concept toward a higher-level abstraction (**DG2**), the cost of verifying each change toward a faster loop that does not require a full build or screen reader (**DG3**), and the mobile-gesture gap beyond keyboard input (**DG5**).

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 [80] based on Chartability [76] 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. A library with no native chart types required the abstraction to be data-driven and chart-type-agnostic rather than tied to fixed templates, sharpening **DG2** into a concrete constraint.

6.4.4 Infrastructure from Practice

The three collaborations converged on the design goals above. Two of them demanded new infrastructure before *Skeleton* could exist: a way to visually render and inspect navigation structures (**DG1**), and a higher-level, data-driven abstraction for specifying navigation without hand-wiring every node and edge (**DG2**). We built two pieces of infrastructure to address these, described next; the remaining goals (**DG3**, **DG4**, **DG5**) are met by *Skeleton*'s authoring environment (Section 6.5). These infrastructural improvements as well as *Skeleton* are open source on our GitHub repo.

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. Our *Dimensions API* addresses this.

Our new vocabulary mirrors how practitioners already think about their data, much as a grammar of graphics lets a designer write a visualization at the level of data fields rather than primitive

marks. Rather than specifying a graph directly, a practitioner describes the *dimensions* of their data, the meaningful axes along which a user might want to navigate, and a single build step constructs the full node-link structure automatically. Notably, this enables systems to programmatically construct navigable structures as long as dimensions are configurable. Each dimension declares a `type` (`categorical` or `numerical`) and a `dimensionKey` naming the field it traverses, together with a `behavior` block that governs traversal (how navigation behaves at group boundaries, and whether leaf nodes are reachable laterally across divisions) and an `operations` block that governs how the data is shaped into the hierarchy (sorting, filtering, and binning numerical ranges). Section 6.4.4.3 gives a deeper look into the API's parameters.

This data-dimensions framing both aligns with and departs from *Olli* [25], which similarly lets authors express a navigable structure over a chart from a higher-level specification rather than hand-built nodes. Both share the premise that fields, not graph primitives, are the right authoring unit. The difference is what the specification produces and where it lives: *Olli* compiles a chart specification into a traversable tree that an author does not subsequently manipulate as a graph, whereas the *Dimensions API* generates an explicit Data Navigator node-link structure whose every node, edge, and rule remains addressable, so that the declarative output is itself the object a practitioner can then inspect in the *Inspector* and edit directly in *Skeleton*.

6.4.4.3 Dimensions API Reference

Each dimension in the *Dimensions API* (Section 6.4.4) declares a `type` (`categorical` or `numerical`, inferred from the data when omitted) and a `dimensionKey` naming the data field it traverses. Two further groups of properties govern behavior. A `behavior` block controls traversal: `extents` sets boundary behavior (`terminal` stops at the edges of a group, `circular` wraps around, and `bridgedCousins` carries focus across to the nearest element of an adjacent group rather than stopping or wrapping), and `childmostNavigation` determines whether leaf-level nodes are reachable laterally across a dimension's divisions (`across`) or only within a single division before returning to a parent (`within`, the default). An `operations` block controls how the data is shaped into the hierarchy: a `sortFunction` orders divisions and leaves, a `filterFunction` selects which data participate, and `createNumericalSubdivisions` bins a numerical dimension into a chosen number of ranges.

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.

To make the input and output concrete, the examples below use this small four-row dataset of average temperatures, with an explicit `id` on each row:

```
const data = [
  { id: '_0', month: 'Jan', city: 'NYC', temp: 0 },
  { id: '_1', month: 'Jul', city: 'NYC', temp: 25 },
  { id: '_2', month: 'Jan', city: 'Sydney', temp: 22 },
  { id: '_3', month: 'Jul', city: 'Sydney', temp: 8 }
];
```

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' }
    }
  ]
}
```

From this declaration, the build step does three things automatically. It assembles the nodes and the parent-child and sibling edges that connect them, including, when multiple dimensions are declared, the cross-dimension edges that let a user move between hierarchies over shared leaves. It applies the boundary and childmost rules to decide what happens at each edge of the structure. And it assigns keyboard navigation rules to each dimension, drawing from a pool of directional key pairs; because that pool is finite, the API recommends keeping a structure to six dimensions or fewer and asks for explicit navigation rules beyond that point.

This bounds the structures the abstraction expresses by default: it generates the regular, dimensionally-organized hierarchies that recur across common chart types, and structures that do not decompose into a small set of data dimensions (or that exceed the directional-key budget) fall back to explicit graph construction or manual editing in the editor.

The abstraction is otherwise 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, where no native chart types exist to anchor a pattern: the API mirrors data fields and encoding choices as a set of dimensions, so a thin wrapper can map an arbitrary glyph assembly onto a handful of recurring dimensional shapes (Section 6.4.3).

The vocabulary composes to richer topologies simply by adding dimensions. Declaring two categorical dimensions over the same dataset, for instance a temperature series recorded by month and by city, produces a dual hierarchy in which left and right traverse one axis and up and down traverse the other, with both sharing the same leaf nodes; here the month dimension wraps at its boundaries while the city dimension stops:

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

```
    {
      dimensionKey: 'city',
      type: 'categorical',
      behavior: {
        extents: 'terminal',
        childmostNavigation: 'across'
      }
    }
  ]
}
```

This two-dimension declaration assigns a keyboard control to each axis. Table 6.1 lists the controls Data Navigator generates for it; the arrow keys move between siblings along each dimension, `Enter` drills toward the data, and each dimension receives its own drill-out key from the default key pool (here `W` and `J`).

Table 6.1: Keyboard controls generated for the two-dimension (`month`, `city`) structure over the sample dataset. Movement along `month` wraps (`circular`); movement along `city` stops at the ends (`terminal`).

Key	Action
← / →	Move to previous / next `month` (wraps Dec–Jan)
↑ / ↓	Move to previous / next `city` (stops at ends)
`Enter`	Drill in (dimension → division → data)
`W`	Drill out along the `month` grouping
`J`	Drill out along the `city` grouping
`Esc`	Exit the structure

Finally, to show what the build step emits, the two-dimension declaration above, applied to the four-row dataset, expands into the nodes and edges below. Each dimension produces a dimension node carrying its `dimensionKey`, a set of division nodes (one per distinct value, each tagged with the field it was `derived` from), and the shared leaf nodes that carry the original rows. A node's `edges` list the edges it participates in; each edge names a `source`, a `target`, and the `navigationRules` that traverse it. Crucially, a single edge is bidirectional: the `left`/`right` pair on one edge means pressing `left` moves toward its `source` and `right` toward its `target`, so the `month` and `city` axes at the leaf level each collapse to one edge with two rules (abbreviated here for space):

```
// nodes (excerpt: one division + one leaf per axis)
{
  month: {
    id: 'month',
    data: { dimensionKey: 'month' },
    edges: ['month-Jan', 'any-exit']
  },
```

```
  city: {
    id: 'city',
    data: { dimensionKey: 'city' },
    edges: ['city-NYC', 'any-exit']
  },
  'Jan': {
    id: 'Jan',
    derivedNode: 'month',
    data: { month: 'Jan' },
    edges: [
      'Jan-Jul',
      'month-Jan',
      'Jan-_0'
    ]
  },
  'NYC':  {
    id: 'NYC',
    derivedNode: 'city',
    data: { city: 'NYC' },
    edges: [
      'NYC-Sydney',
      'city-NYC',
      'NYC-_0'
    ]
  },
  _0: {
    id: '_0',
    data: { month: 'Jan', city: 'NYC', temp: 0 },
    edges: [
      'leaf-_0-_1',    // month axis (l/r)
      'leaf-_0-_2',    // city axis  (u/d)
      'Jan-_0', 'NYC-_0',
      'any-exit'
    ]
  }
  // Jul, Sydney, and _1/_2/_3
  // follow the same pattern
}

// edges
{
  'month-Jan': {
    source: 'month',
    target: 'Jan',
```

```
    navigationRules: ['drill-out_month', 'drill-in']
  },
  'city-NYC': {
    source: 'city',
    target: 'NYC',
    navigationRules: ['drill-out_city', 'drill-in']
  },
  'Jan-Jul': {
    source: 'Jan',
    target: 'Jul',
    navigationRules: ['left','right']
  },
  'NYC-Sydney': {
    source: 'NYC',
    target: 'Sydney',
    navigationRules: ['up','down'] },
  'leaf-_0-_1': {
    source: '_0',
    target: '_1',
    navigationRules: ['left','right']
  },
  'leaf-_0-_2': {
    source: '_0',
    target: '_2',
    navigationRules: ['up','down']
  },
  // ...etc
}

// navigationRules
{
  left: { key: 'ArrowLeft',  direction: 'source' },
  right: { key: 'ArrowRight', direction: 'target' },
  up: { key: 'ArrowUp',    direction: 'source' },
  down: { key: 'ArrowDown',  direction: 'target' },
  'drill-in': { key: 'Enter', direction: 'target' },
  'drill-out_month': {
    key: 'KeyW',
    direction: 'source'
  },
  'drill-out_city': {
    key: 'KeyJ',
    direction: 'source'
  }
```

```
}
```

Every node, edge, and rule in this output remains individually addressable: this is the structure the *Inspector* renders and that practitioners select and edit directly in *Skeleton*, rather than a compiled artifact they cannot reopen.

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, test, and iterate on it with immediate feedback. *Skeleton* is that environment. It integrates both and extends them with a guided preparation phase, a spatial rendering canvas, and a live testing mode, reversing the *Inspector*'s code-first direction: practitioners design a navigation structure visually and inspect the code representation as a consequence of that design. Each of its 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.

A principle runs through the environment: everything *Skeleton* generates is an editable default, not a fixed output. The *Dimensions API* proposes a topology, the scaffold proposes spatial positions and group outlines, and the preparation wizard proposes labels and rules, but each is materialized as concrete nodes, edges, positions, and text that a practitioner can then select and override by direct manipulation, or bypass entirely by constructing nodes and edges manually over an image. The declarative and automated layers lower the cost of a reasonable starting point; they do not constrain what the final structure can be.

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: 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. Because dimensions are derived from whatever fields a dataset actually carries rather than from a fixed, pre-registered schema, the approach does not require the data to be known in advance: a different dataset simply yields a different set of inferred

dimensions, and a visualization with no tabular dataset at all is handled through the image-only path. 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 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 hierarchy 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

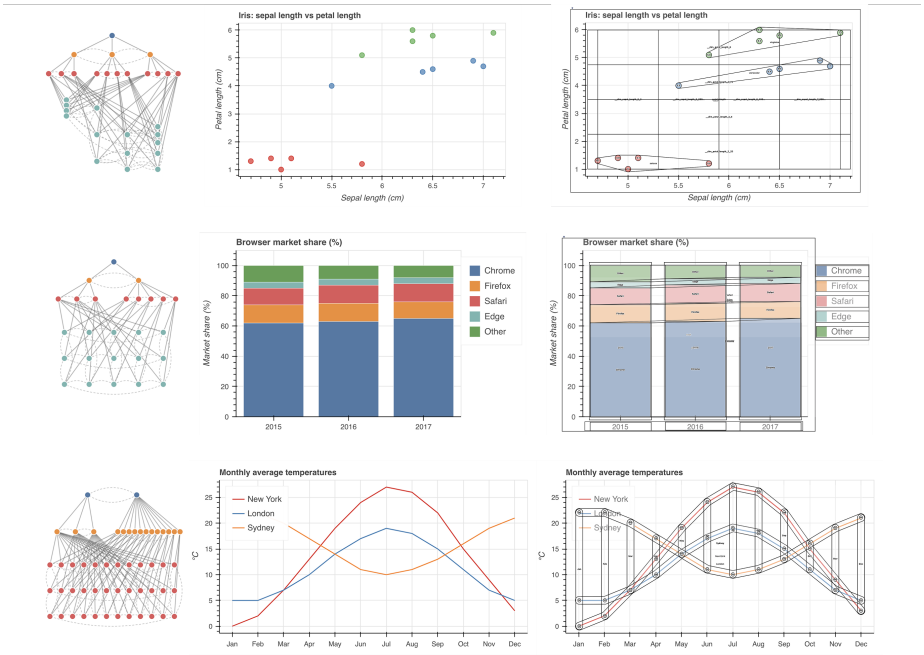


Figure 6.3: Input data transformed into a navigable structure using the *Dimensions API* and visualized with our *Inspector* gadget (left). The input chart (middle). The navigable structure is transformed and drawn over the chart using the *Scaffolding Engine* (right).

Seeing the system, seeing the experience. The editor is *Skeleton*'s primary authoring environment (Figure 6.1) 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 [241] as a layout engine.

The scaffold renders a Vega chart specification to a hidden, off-screen container and extracts 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 [151]. Editing labels can apply to all nodes of the same type or to a single instance.

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: redun-

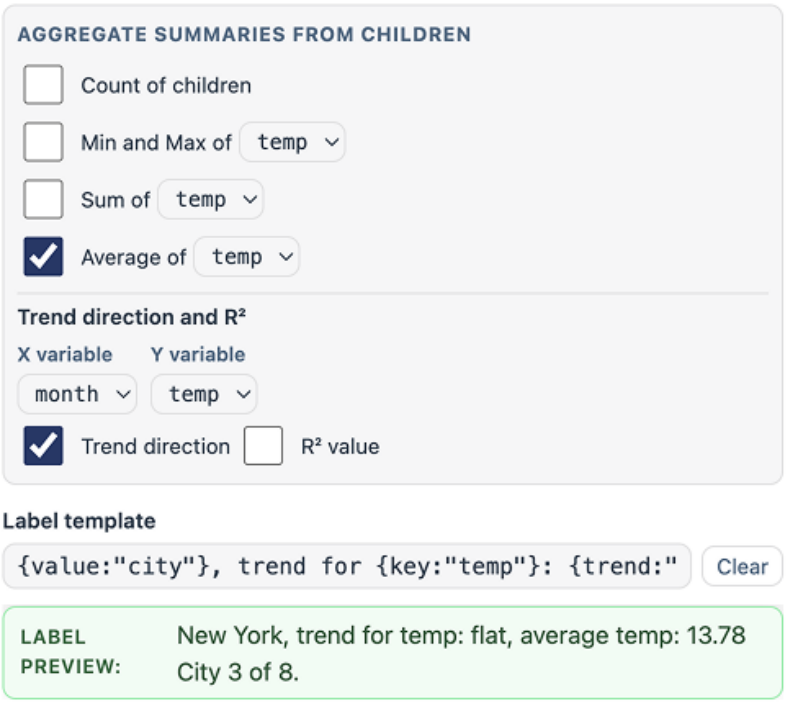


Figure 6.4: Group label pattern builder, including an array of aggregate summary options, template formatter field, and preview.

dancies 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). Practitioners can verify at a glance both where focus is and what role it occupies. A mobile-friendly text-chat

mode is also available, in which practitioners navigate by typing natural language commands, motivated by the Adobe collaboration (Section 6.4.2).

Practice-based validation. The testing stage also served, during development, as the primary debugging interface for *Skeleton*'s own data pipeline. Additionally, 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 use list-based navigation while in the editor (not in testing) in case users build themselves into a keyboard trap.

6.5.4 System Implementation: Technical Details

This section details the three technical components behind *Skeleton*'s spatial authoring: how the scaffold repurposes a visualization rendering engine to approximate mark positions, how group outlines are computed from those positions, and how a lightweight, non-ML computer vision pass aligns the scaffold to an uploaded chart image. We intentionally chose deterministic math approaches at any opportunity, for speed and inspectability benefits.

6.5.4.1 Vega as a Deterministic Layout Engine

The scaffold (Section 6.5, "Leveraging visualization as a scaffolding engine") treats a visualization grammar as a coordinate oracle rather than a rendering target. From the active configuration, which carries a chart type, field-to-channel mappings (`xField`, `yField`, and an optional `colorField`), mark parameters, and an outer plot box, *Skeleton* assembles a *Vega-Lite* [241] specification and renders it through `vega-embed` into a hidden, off-screen container using the SVG renderer. The container is never attached to the visible document; it exists only so that *Vega* will compile a full scenegraph and, with it, the scale functions that map data values to pixels.

Positions are then read directly from the compiled view rather than parsed from the rendered output. For each leaf node, *Skeleton* evaluates the view's `x` and `y` scales on the corresponding data values: a band scale supplies a bar's horizontal position and, through its `bandwidth`, its width, while a linear scale maps a quantitative value to a pixel height with the baseline taken at the scale's zero. Stacked bars accumulate per-segment offsets in data space before scaling; clustered bars add the sub-band offset; scatter and line marks evaluate both scales at each point. Because the scale functions are pure, this path needs no DOM measurement and is fully deterministic: identical configurations yield identical coordinates. A secondary path, used when scale evaluation does not apply, parses the hidden SVG and reads mark bounding boxes instead.

The configuration distinguishes the inner mark area from the outer box. *Vega-Lite*'s `width` and `height` describe the inner area in which marks are drawn, while a `padding` object reserves space for axes and labels; *Skeleton* treats `plotWidth` and `plotHeight` as the outer, border-box dimensions and derives the inner area by subtracting padding. A mark at normalized position f within the inner area therefore lands at image coordinate $o + p + f\, w_{\text{in}}$, where o is the plot

offset, p the leading padding, and w_{in} the inner extent. This relation is what later makes padding recoverable.

A practical subtlety motivates the rest of this section. The engine that produced the uploaded image is generally unknown and is rarely *Vega*; our co-designers authored charts in other tools entirely (Section 6.4). What transfers across engines is the *relative* geometry of the marks, the spacing of bars or the placement of points within the data rectangle, not the absolute pixel offsets, which depend on how much room each particular chart reserved for its axes, ticks, labels, and title. The scaffold reconstructs the relative geometry faithfully; reconciling it with the absolute placement in the image is the task taken up by the padding estimation described below.

6.5.4.2 Computing Group Outlines

Once leaf positions exist, the group nodes above them (divisions, dimensions, and the root) receive a visible outline computed purely from the geometry of their descendants. Each leaf contributes a rectangle (for bars) or a circle (for points), and a chosen strategy turns the collection into a single SVG path string. These functions live in the Data Navigator core (`geometry.ts`) and are side-effect free, taking arrays of rectangles or circles and an optional padding term and returning a path `d` attribute, so they can be reused outside *Skeleton* and packaged for export.

Four strategies cover the cases that recurred during co-design (Section 6.4). A *bounding rectangle* encloses all descendants in the single axis-aligned box given by their combined extent. A *union* emits one rectangular subpath per mark and relies on the SVG fill rule to render disconnected regions, which suits groups whose marks are spatially separated. A *convex hull* collects the corner points of every rectangle (or sixteen sampled points around every circle) and runs a Graham scan [105], optionally pushing each hull vertex radially outward from the centroid to add padding; this gives a tight, convex boundary appropriate to scatter clusters and bar groups. For line series, an *offset path* traces the polyline of points at a fixed perpendicular distance on each side, using per-vertex averaged normals for miter joins and semicircular arc caps at the ends, producing a closed band that follows the line. Numerical dimensions additionally support a grid drawn over the dimension's value bounds.

Because every outline derives from the same leaf coordinates the scaffold computed, changing a mark position, a chart type, or the padding propagates automatically to the group shapes, and authors edit outline strategy and padding as ordinary node properties rather than drawing shapes by hand.

6.5.4.3 Estimating Plot Padding from the Image

The scaffold's layout is internally consistent but free floating: it knows where marks sit relative to one another, not how the uploaded image split its canvas between the data rectangle and the axis space, which authors originally recovered by dragging the four padding values until the scaffold marks settled onto the real marks. Our most recent work automates it with a small, fast, dependency-free computer vision pass that runs entirely in the browser, in the spirit of chart reverse-engineering that recovers marks from bitmap images [221, 242] but reduced to coarse mark localization, since the scaffold already supplies the data.

Detection proceeds on a downscaled copy of the plot region. The image is drawn into an off-screen canvas with its longer side capped near one thousand pixels, and a binary foreground mask is built by thresholding the pixel data [217]. The key observation is that data marks are almost always rendered in color while chart chrome (axis lines, gridlines, tick labels, titles, and legend text) is grayscale. A saturation threshold therefore isolates the marks and prevents a dark axis line from bridging adjacent bars into a single blob; for the rare monochrome chart the pass falls back to a dark-ink mask. Connected-component labeling [231] with eight-connectivity groups the mask into candidate marks, each carrying a bounding box, centroid, area, and mean color. Two filters clean the result: components far smaller than the median are discarded as legend swatches, and for line and area charts, where the mark is one long stroke rather than discrete glyphs, the endpoints of the dominant component are used in place of separate marks. The pass returns the extreme detected marks along each axis (leftmost and rightmost, topmost and bottommost) as points in image space.

Padding then follows in closed form. Recall that a scaffold mark at normalized inner position f sits at $o + p + f\, w_{\text{in}}$, and that f is invariant to padding because it is fixed by the data and the scale. Taking the two extreme scaffold marks on an axis, with inner positions f_{lo} and f_{hi}, and the two detected image positions d_{lo} and d_{hi} they should align to, the inner extent and the two padding values follow while the outer box (o, W) is held fixed:

$$w'_{\text{in}} = \frac{d_{\text{hi}} - d_{\text{lo}}}{f_{\text{hi}} - f_{\text{lo}}}, \quad p'_{\text{lo}} = d_{\text{lo}} - o - f_{\text{lo}}\, w'_{\text{in}}, \quad p'_{\text{hi}} = W - w'_{\text{in}} - p'_{\text{lo}}.$$

The two axes are solved independently. Several guards keep the result trustworthy: an axis whose two marks are nearly coincident in f is ill-conditioned and is skipped, negative padding is clamped, and a solved inner extent larger than the outer box signals that the box itself is too small and is reported rather than forced. The pass runs once automatically when an image, a chart type, field mappings, and at least two scaffold marks are all present, and on demand thereafter from a control in the scaffold panel; after applying the padding it re-measures the residual between detected and scaffold marks as a one-shot verification.

In practice this recovers padding consistent with the original chart across the four canonical types we tested (bar, stacked bar, line, and scatter), shown in Figure 6.5. On the bar example, for instance, the solver recovers a left padding close to the width of the chart's y-axis label gutter and a small right padding, with no manual adjustment. The approach is deliberately heuristic and assumes colored marks over light chrome; its single entry point makes a heavier detector (for example an `OpenCV` build) substitutable without changing the padding solver. We regard this as promising but early, and a fuller evaluation across more diverse chart images remains future work.

6.6 User Study

Our co-design work was motivated by the problems of the communities the research team was embedded in, and our co-designers were deeply familiar with the problem space, having spent months or years working on accessible navigation. We still needed to understand what happens when practitioners who are *not* embedded in this process design, author, and debug navigation

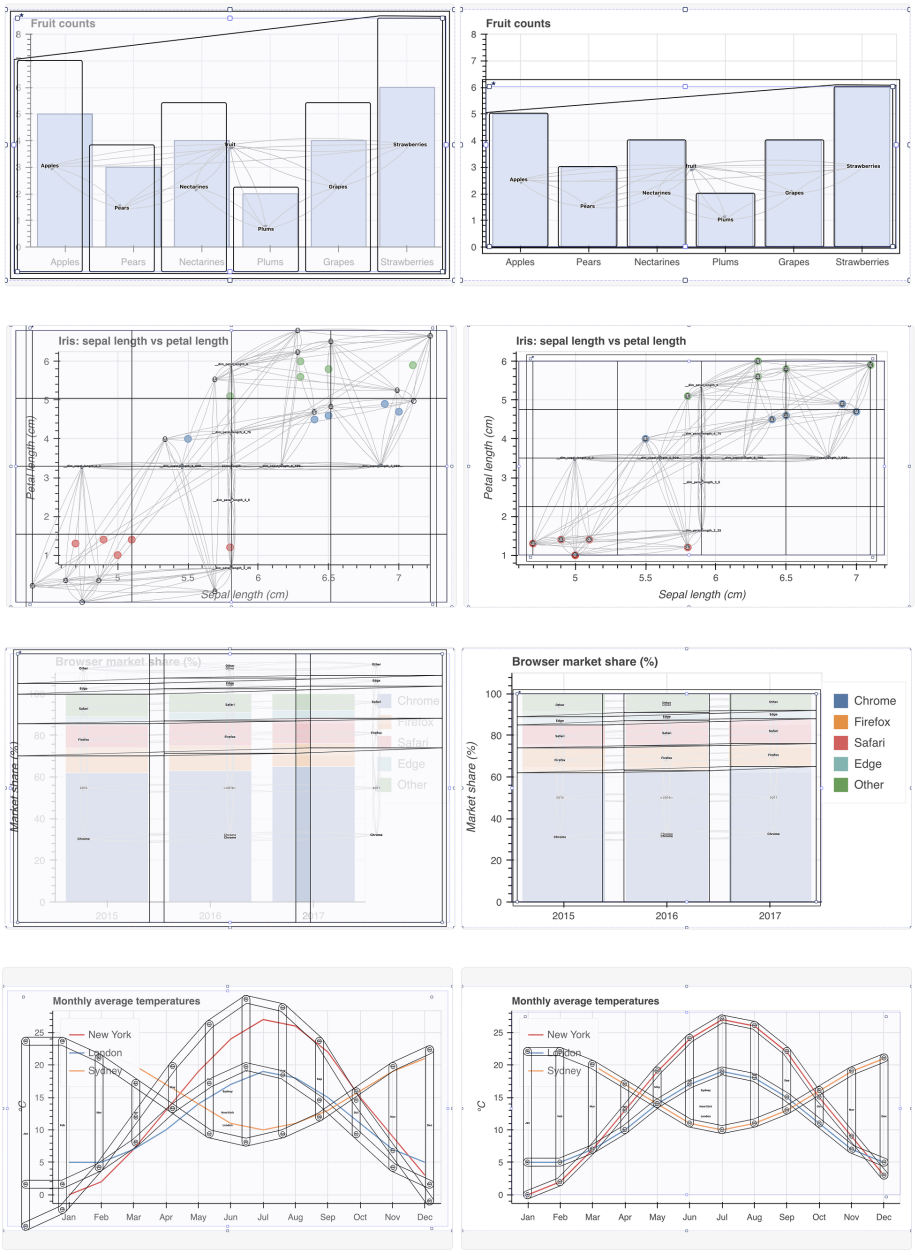


Figure 6.5: Computed layout for marks via Vega's visualization engine plus group outlines (left) and with padding estimation using our non-ML computer vision approach (right) for various types of common visualizations.

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 [7] (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 and adds 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 [31] and affinity diagramming [119], 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?"

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. 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."

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.

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.

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 6.6) 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. P8 concluded that well-written alternative text was sufficient for the screen reader experience and chose not to build any 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 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,

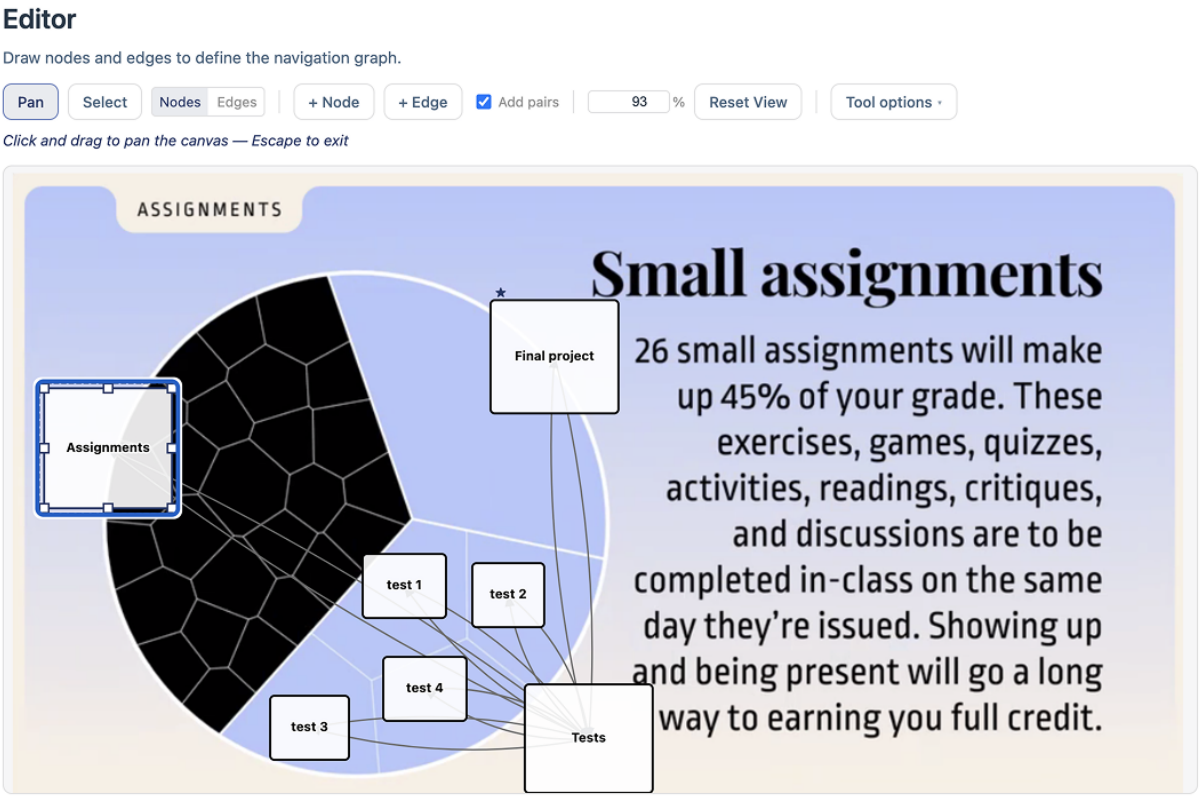


Figure 6.6: Re-creation of P8's moment of realization, placing nodes manually: not every element in their voronoi pie chart *should* be navigable.

"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."

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 a 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 enabled continuous design improvements.

This mechanism is consistent with Schön's account of reflective practice [251]: 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 (Section 6.4) provides complementary evidence: in each collaboration, sighted practitioners 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. The practical implication is broad: 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 insti-

tutional obligation to meet [76, 153, 258]. 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 (Section 6.7.2). But alongside that desire for guidance, they began doing something compliance framing does not typically produce: they encountered complexity and then *iterated* (Section 6.7.3). 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. 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. It is worth being precise about where generality holds and where it stops. Because the *Dimensions API* derives structure from whatever fields a dataset carries rather than from a fixed schema, the data side of the approach already extends to datasets not known in advance; what does not yet generalize is the prior, harder question of which dimensions, if any, a bespoke visual should expose when its visual form was never organized around a tabular structure to begin with. Generalizing the abstraction is therefore not primarily a matter of supporting more schemas but of supporting the design judgment that precedes choosing a schema at all.

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 accessible 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 [334], 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 surfaces design questions but does not answer them: several participants asked what constitutes good navigation, and the tool had nothing authoritative to offer. As Section 6.8 argues, our study evaluated whether making structure visible stimulated design consideration, not whether the resulting designs were good for the people who will use them, which is a distinct question that 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.

Our approach using *Skeleton* as a design probe only with sighted participants is 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 multimodal data experiences.

A further boundary concerns the kind of visualization *Skeleton* targets. Highly interactive or dynamic visualizations, where selecting an element reloads the data, where marks animate, or where the navigable structure itself changes in response to user interaction, are not yet addressed.

We see this as an important future direction rather than a fundamental obstacle: the same argument for making structure visible applies with more force when there are several structures to coordinate, and an inspectable representation may be precisely what makes a dynamic navigation experience tractable to author.

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. 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, prompting them to ask whether the design of their structure was *good* rather than merely whether it passed.

The implication generalizes beyond our tool: wherever a non-visual structure or user experience is authored without a visible representation, building one the author can react to and manipulate invites them to design. 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 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.

6.11 Postface

While this chapter did not take the liberty of making a strong position statement, it is clear to us now that when viewed through the social model of disability (where disability is produced between human interaction with systems and environments) practitioners who are not traditionally considered "people with disabilities" in the medical sense may still face barriers to access when using an interface or system. *Systems can disable any user.*

As another note worth making, we also attempted to "vibe-code" the prototype for this work as well, and encountered (ironically) many barriers while developing, both that we personally faced and that were unfortunately also produced for end users. Remediating our prototype has taken a non-trivial amount of manual and agentic effort.

Large-language models are not necessarily skilled enough out of the box, at the time of this writing, to make highly complex research prototypes that focus on cutting-edge accessibility functionality. The overall work spent vibe-coding now vastly outpaces whatever work we would have invested if we hadn't vibe-coded at all, because we will need to virtually rebuild this entire prototype from scratch before deploying it live for real-world use. It is, in this sense, a throwaway research prototype in its current condition. But at the same time, we did learn that vibe-coding in contexts with significant system complexity also takes from its user an opportunity to solve the engineering problems themselves, which is a challenge that we enjoy and believe helps us to better understand our own work.

Perhaps in the future, we will submit an experience report or write a workshop paper on the *self* and *social* barriers that vibe-coding introduced into our accessibility-centered prototyping work.

Part IV

Interaction: Exploring New Possibilities for Blind Data Science

Chapter 7
Cross-perception: Rethinking Input Design Towards Blind Analytical Interaction

7.1 Preface

Interactive visualization research has produced a rich repertoire of interaction techniques (brushing, linking, and cross-filtering chief among them) that accelerate exploratory data analysis for sighted users. And accessible visualization research has made substantial progress on making data perceivable and navigable for blind users.

Yet what remains largely unaddressed is the *analytical* tier of blind interaction: the capacity to actively filter, conjecture, and test hypotheses through direct data manipulation. In existing systems that accomplish interactive, analytical tasks, users predominantly query text or sequentially navigate across discrete units of information, paradigms which focus the user on one piece of information at a time. These experiences limit a user's ability to rapidly explore and interrogate multiple dimensions within a dataset simultaneously.

To target this analytical gap, we introduce *cross-perception*, an interaction technique that adapts cross-filtering for non-visual, tactile interaction. *Cross-perception* enables coordinated tactile brushing over one data space to produce perceptible computational output in a separate, linked space, without requiring serial navigation between them.

We instantiate *cross-perception* in the *cross-feelter*, a prototype hardware device with dual motorized faders, and evaluate it against screen reader methods in a within-subjects study with 15 blind participants, 7 of whom have professional data expertise. Cross-feelter substantially improves exploration speed (+90%), query generation (+188% computational, +54% spoken), enjoyment, and reduces anxiety (particularly for participants without prior data expertise). We discuss implications for interaction design in accessible visualization and propose *cross-perception* as a framework for a broader class of non-visual, analytically-oriented data interaction techniques.

7.1.1 Context

This project has been a pet project, tandem to the main thread of the others in this thesis. It stands out a bit: centered exclusively on *blind* practitioners and people working with data, as opposed to sighted practitioners who are making things for people with disabilities.

We unpack this in the chapter, but our prior co-design work with a blind colleague eventually led to this project. And if anything, the greatest takeaway from this chapter is that, in true scientific fashion, simply observing something with a fresh perspective and conjecturing on possibilities can lead to fantastic places.

This project was also borne out of a tiny resistance to the then-rapid rise of large-language models. All of a sudden, certain problems that were very hard in accessible visualization (such as generating descriptions for visualizations with dynamic and changing data) were orders of

magnitude less difficult. While we argue (outside of the scope of this thesis) that these new technologies exacerbate existing issues of bias, trust, and validation for people who are blind [75], generally speaking they are undeniably revolutionary.

And yet, I (Frank) had often spoken about my core motivation for leaving a well-paying, highly-supported, staff-level engineering role to start a PhD in HCI: because to me, the greatest challenge was figuring out how to motivate people to think more deeply about their work. At Visa, we automated much of our accessibility by default. And when we realized designers and developers rarely deviated from the defaults, we added validation and checks in the console that would log or throw errors to motivate them to stop and think about how they were using our charts. This ultimately made me want to study human behavior and how tools can shape practitioners towards better outcomes.

But to me, I had always assumed that *practitioners would do the work*. Automation takes away the thinking! And my mission was to elicit deeper consideration and better design. I had assumed that humans *want* knowledge and expertise. But agentic tooling and LLMs began to challenge those assumptions. Perhaps *convenience* is what people really want. People seem to now call on LLMs for whatever labor they value the least. And accessibility is now a common project for LLM-based intervention.

So *cross-perception* was motivated by looking where language models have little influence: in novel hardware development. As of writing, it is still required for a human to solder boards, design circuitry, model, 3D print, and assemble hardware prototypes. So this project was a way to look in a direction that few others are paying any attention. While there seems to be a gold rush to move to higher and higher levels of software abstraction, this project is a return to lower levels of hardware components, tactors, and interaction design.

This project was also immensely difficult to recruit for. On the one hand, I had several blind data practitioners, researchers, and scientists who were colleagues, friends, and mentors supporting me. But few of them had the time or bandwidth to participate in our study. Recruiting locally was quite hard, so we decided to recruit non-data practitioners locally and, if we managed to get lucky, pull off running our studies "in the location of our participant's choosing, should they consent to assume risk" (roughly paraphrased, as our IRB allowed). This verbiage meant that our participants could opt not to participate in our lab space on campus but rather choose their own location (with certain conditions). This ultimately enabled us to recruit blind data practitioners at conferences (ASSETS and CSUN, in particular), which made the study for this project stretch nearly a year and a half (but helped us to see the finish line).

But it was worth it. Our participants were fantastic. And not to spoil the chapter's findings, but we discovered that hardware and input design might be the key to a whole new spectrum of studies where data interaction is framed as a fun and generative activity, and not just something dry, boring, or professionalized as data work often is, in practice.

This chapter was adapted from our paper, soon to be submitted to TVCG:

F. Elavsky, Y. Li, J. Jang, L. Nadolskis, P. Carrington, and D. Moritz, '*Cross-Perception*: Rethinking Input Design Towards Blind Analytical Interaction', *IEEE Transactions on Visualization and Computer Graphics*, 2027.

7.2 Overview

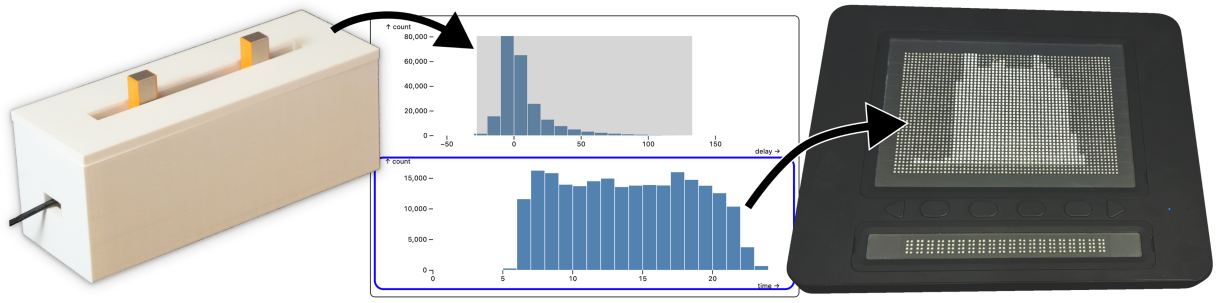


Figure 7.1: Interaction and perception in one space while being able to perceive output in a separate space is the cornerstone of cross-filtering. Our prototype *cross-feelter* (left) can manipulate a visual cross-filter on one visualization (middle) in order to produce output in a separate visualization, as a tactile graphic (right).

The visualization community has spent decades formalizing interaction techniques for sighted users: brushing scatterplots [17], linking coordinated views [36], and cross-filtering aggregated data to rapidly surface relationships across dimensions [1, 306]. These techniques are not only conveniences. Empirical work has shown that fast, direct manipulation of data fundamentally changes the speed and character of hypothesis generation, and that even small increases in interaction latency measurably reduce the quality of exploratory analysis [85, 173].

Accessible visualization research has made substantial and genuine progress in recent years. Blind users can access data through rich text descriptions [154, 177], sonification [136, 159, 334], and tactile representations [38, 135, 191, 192, 269, 281]. Increasingly, this work extends beyond rendering to *interaction*: screen readers support structured traversal of chart elements and their relationships at varying levels of granularity [77, 199, 332], and multimodal systems pair conversational agents with tactile displays to support rich data exploration [227].

What existing work addresses is, broadly, the question: *what is here?* A blind user can navigate a data representation to understand its structure, locate specific values, extract summary statistics, ask descriptive questions, and build a mental model of what the data contains. This is *access-oriented* interaction, and current tools support it with increasing sophistication.

What remains largely absent from accessible visualization is a different kind of interaction entirely: the *analysis-oriented* kind, in which a user not only reads a dataset but actively operates on it: filtering subsets, conjecturing relationships, testing hypotheses, and generating new

knowledge through iterative manipulation. These are the fundamental activities of interactive data science, and they have a different character from access-oriented interaction. Questions shift from "*what is here?*" to "*what happens to Y when I constrain X?*" According to taxonomies of analytical tasks, existing work has largely not addressed "characterizing distributions" and "correlating" tasks, which are fundamental to hypothesis-construction [9].

We propose *cross-perception*, an interaction technique that adapts cross-filtering for non-visual contexts while preserving the simultaneity that makes it analytically useful. Cross-perception is defined by two properties. First, *coordinated tactile brushing*: the user manipulates a persistent tactile input in one interaction space, selecting a range across a data dimension. Second, *linked perceptual output*: the computational result of that manipulation is immediately perceivable in a separate space, without requiring serial navigation between them. Together, these properties support not only access to data but active analytical engagement with it: the capacity to ask iterative questions through direct manipulation.

To evaluate cross-perception and explore its design space, we built the *cross-feelter*: a prototype device with two motorized linear faders that maintain persistent, absolute tactile position for brush handles, paired with a refreshable braille display for tactile output. We conducted a within-subjects study with 15 blind participants, comparing the cross-feelter against a screen reader baseline on structured and open-ended data exploration tasks.

In our work, we address four research questions:

R1 (Exploratory): How might we formalize a tactile interaction approach that enables coordinated, non-visual interaction and perception across multiple spaces?

R2 (Quantitative: Objective): In what measurable ways can we observe improvements to blind data interaction using our approach, in terms of speed, accuracy, precision, and quantity of data queries?

R3 (Quantitative: Subjective): What are the self-reported effects on stress, cognitive load, anxiety, and enjoyment when using our approach?

R4 (Qualitative): In what ways do blind data practitioners imagine extending our approach and its instantiating hardware to new contexts?

This paper makes three contributions. First, we formalize *cross-perception* as an interaction technique for non-visual *analytical* data interaction; one that equips blind users to actively filter, conjecture, and test hypotheses through direct tactile manipulation, not only to access and navigate data (Section 7.4). Second, we provide an empirical evaluation demonstrating that cross-perception yields substantial improvements in exploration speed, query generation, and user experience relative to screen reader interaction, while maintaining comparable accuracy and precision (Section 7.5–7.7). Third, we argue that accessible visualization has made significant progress at the access-oriented tier of interaction, and that our field's frontier has shifted: the work now should be to give blind users *analytical agency*: the ability to interrogate data, not only to read it (Section 7.8).

7.3 Related Work

This paper sits at the intersection of two bodies of work: research on cross-filtering and analytical interaction in data visualization, and research on making data accessible to blind users. We review each in turn, then characterize the specific gap that cross-perception addresses.

7.3.1 Cross-Filtering and Interactive Visual Analysis

A note on terminology. *Brushing* is the direct manipulation gesture of selecting a contiguous visual range across a data dimension [17]. *Linking* propagates that selection across coordinated views [36]. *Cross-filtering* applies the linked operation as a filter on shared underlying data, constraining what is visible across views rather than merely annotating selections [1, 306].

What makes cross-filtering analytically valuable is not that users can manipulate the interface, but that the gesture and its computational consequence are perceived simultaneously. Even 500ms of additional latency measurably reduces the rate at which users generate observations and formulate hypotheses [173], and providing partial results incrementally reduces but does not eliminate this cost [85]. Temporal coupling between input and output is the mechanism by which direct manipulation supports hypothesis generation, not an implementation detail [122]. Recent work on scalable cross-filtering has continued to optimize this coupling through improved architectures [123], including pre-computed aggregates [204]. This distinction, between analytical interaction that constructs new information through manipulation and navigational interaction that locates information already present, motivates the core design challenge of cross-perception.

Previous work establishes that cross-filtering supports *analytical* interaction, in which the user constructs and tests hypotheses through rapid, iterative, direct manipulation, rather than navigating toward information that is already present. The value of this analytical interaction is that the user, not a system, gains new insights, questions, and understanding [264].

7.3.2 Non-Visual Data Representations

Accessible visualization has produced many systems for making data perceivable to blind users across text, audio, and tactile modalities [79, 103, 154, 159, 177, 224, 281]. We position cross-perception as complementing this work: many rendering and representation problems have made significant advancements in recent years; but analytical interaction problems, comparatively, have not.

Sonification. Sonification, mapping data variables to auditory properties such as pitch or timbre, is a distinct and sophisticated field with its own tooling, theory, and evaluation methodologies [125]. Recent work has formalized sonification as a first-class rendering approach with grammar-level expressiveness, enabling mappings comparable in richness to visual encodings [159]. Some work has explored data-centric approaches that treat sonification and visualization as parallel co-equal representations [334].

A note on auditory perception is warranted: the human auditory system is capable of parsing concurrent streams and segregating them into distinct perceptual objects [32]. However, the relevant constraint for blind and low vision users working with data is more specific: in task-driven contexts, synthesized speech from a screen reader competes with data-representing audio

in the same channel, increasing cognitive load of the user and providing complex challenges for authors when designing speech audio and sonification interfaces [97, 159, 289]. Dual-task and multi-task operations become difficult or impossible, which is a design challenge for cross-filtering specifically, where the filter state and its output must be perceived simultaneously.

Tactile and haptic output. Research on tactile data representation spans several decades, from foundational work on tactual perception [247] to recent studies formalizing tactile alternatives to common chart types [79, 224]. More recent work has explored simultaneous tactile-and-audio output [38], slide-tone and tilt-tone haptic encodings for shape characteristics of graphs [81], comparative studies on tactile vs audio output [52], and design guidelines for multimodal touchscreen-based graphics that inform both resolution and tactile landmark placement [103].

7.3.3 Access-Oriented Data Interaction

In parallel with rendering advances, accessible visualization has developed increasingly sophisticated interaction techniques for blind users, such as serial and multi-directional navigation across data structures using a screen reader [77, 332]. However, screen reader traversal is sequential and discrete, which impacts a blind user's ability to extract information from visualizations, impacting accuracy by 61% and costing 210% more time doing so [256]. Empirical work has engaged "questions" that everyday blind folks have of data representations [160], and most recently, conversational agents have been paired with refreshable tactile displays to support question-based exploration combining voice input with tactile output [227].

These systems share a common character: they support *access-oriented interaction*, attempting to bridge the gaps that exist between sighted and blind access to information. However, the activities that remain unsupported require constructing information through manipulation rather than locating it through navigation; the work of professional data science. In analysis-oriented interaction, the question is not merely lookup or query tasks such as *what is the value at this position* but along the lines of *what is the joint distribution within this subrange*, an answer that does not exist until a filter is applied (and a question that may not even exist unless within a system that supports direct interaction and iteration with the data).

7.3.4 Tactile and Haptic *Input*

Tactile and haptic input for data interaction has been explored in immersive contexts (not related to accessibility or blind interaction). For example, Cordeil and colleagues' *embodied axes* allow manipulation of data axes as physical objects in augmented reality [55], and the *MADE-Axis* extends this to motorized dual-axis faders for AR/VR data environments, demonstrating a cross-filtering use case [268]. Both leverage tactile input and motorized output, though both presuppose sighted users in visual contexts.

Within accessibility-specific work, a touchpad-based brush interface showed 1D filter selection feasibility but provides no queryable positional state when the user's hand is lifted [259]. And a non-motorized slider within an accessible smartphone interface shows the principle of persistent tactile state [329] but cannot be programmatically positioned, which cross-filtering requires for two-way synchronization.

The gap that remains is specific: persistent physical state and simultaneity of input and output perception are both necessary for analytical interaction, and no existing accessible system combines them.

Cross-perception is our attempt to address this gap. Our design is informed by the principles reviewed above: the temporal coupling of cross-filtering, the persistent-state affordances of motorized tactile input, and the insights from prior work on non-visual data interaction regarding the importance of simultaneity, spatial memory, and multi-channel perception. The following section formalizes these into a set of design goals for the interaction technique.

7.4 Formalizing Cross-Perception

Cross-perception is a tactile interaction technique defined by two properties. First, *coordinated tactile brushing*: the user manipulates a persistent tactile input in one interaction space, selecting a range across a data dimension. Second, *linked perceptual output*: the computational result of that manipulation is immediately perceivable in a separate space, without requiring serial navigation between them. These properties directly instantiate the simultaneity requirement established in Section 7.3: brush state and its consequence are perceived together, in the same moment, supporting the rapid inference cycles of analysis-oriented interaction.

In this section, we formalize cross-perception by deriving six design goals from the principles reviewed in the related work. We begin with a design vignette with our co-designer and co-author Nadolskis, that illustrates how the same cognitive mechanisms that underpin our goals appear in naturalistic blind interaction with physical documents.

7.4.1 A Design Vignette: Fuzzy Search and Dual-Task Comparison in Naturalistic Blind Reading

The following vignette, shared with permission by our blind co-designer and co-author Lucas Nadolskis, illustrates two tactile interaction strategies that correspond directly to our design requirements formalized below. We present this vignette as both grounded in the cognitive science and visualization literature reviewed in Section 7.3 as well as evidence that the principles we derive from that literature describe real, expert-level behavior in the wild.

Nadolskis, a blind neuroscience engineer, was reading an embossed academic paper and explaining a particular equation to the primary author as the equation recurred and evolved across multiple pages. The embossed format provides high-resolution tactile rendering at up to 100 dots per inch, and Nadolskis interacted with it using two strategies we have named *fuzzy tactile search* and *dual-task comparison*.

Fuzzy tactile search is a spatial indexing strategy. Nadolskis picked up the stack of pages and thumbed to an approximate location based on tactile position in the stack, touched near the top of the page, immediately recognized from the local texture pattern that he was one page off, flipped to the next page, and navigated directly to the spatial region where the equation began. He then repeated this process to locate the equation's recurrence later in the paper. The strategy relies on two cognitive functions reviewed in Section 7.3: spatial memory, encoding the

approximate location of content within a document's physical structure, [34, 53, 60] and *object-location memory*, the ability to retain and retrieve the position of objects without continuous visual or tactile contact [95, 171, 174]. Both functions require that objects *have* a stable spatial position to encode: they cannot operate on a system where position is reassigned or meaningless.

Dual-task comparison is a simultaneous spatial reasoning strategy. Having located the same equation in two places, Nadolskis placed one hand on each and skimmed both simultaneously, stopping when his hands detected the point of divergence between them. He explained the evolution of the equation by reference to that divergence. This strategy demonstrates exactly the interaction character that cross-filtering supports for sighted users: two data regions perceived in parallel, with relational structure emerging from simultaneous contact rather than sequential reading. It also demonstrates the additional requirement of *simultaneous multi-location tactile interaction*: a user resting a pinky on a known reference point while exploring relative to it with another finger, a pattern documented in prior empirical work on tactile interactive systems [103].

Together, these strategies illustrate a key observation: skilled blind readers of physical documents are not limited to sequential access. The constraint is architectural: the medium must maintain a persistent, meaningful spatial state so that object-location memory can operate, and it must allow simultaneous multi-point contact so that comparison can occur. Our design goals operationalize these requirements for computational, analytical interaction with tactile interfaces.

7.4.2 Design Goals

We derive six design goals from the principles established in Section 7.3 and illustrated by the vignette above.

G1: Maintain and reflect system state through tactile location. Object-location memory requires objects with stable, queryable positions [95, 174]. Tactile input elements must therefore have absolute, persistent positional state: the knob must remain where the user or system last placed it and be readable by touch without performing an action. This also enables motorized two-way synchronization, where the system sets knob position as output and the user sets it as input.

G2: Allow for multiple simultaneous channels of perception. Synthesized speech competes with data-representing audio under cognitive load [97]. Cross-perception routes data manipulation to a tactile input channel, eliminating this competition. Audio output via ARIA-live announcements remains available for precise positional feedback but is kept serial and non-competing.

G3: Correspond tactile-to-computational functional semantics. Interface elements should have names and behaviors that describe what they do [76, 298]. The visual cross-filtering metaphor involves dragging handles along a spatial dimension; cross-perception preserves this by mapping a physical rail sliding to setting a filter boundary, reducing translation between user action and analytical action.

G4: Provide linked interaction across multiple data dimensions. Cross-filtering requires that a filter on one dimension immediately propagates to constrain coordinated views of others that share the same underlying dataset [1, 36]. The filter state set by tactile input must immediately determine the output displayed in linked views; without this, the technique only accomplishes single-view selection.

G5: Support fast, dynamic, and reversible interaction. The analytical value of cross-filtering depends on rapid hypothesis iteration [85, 173]. Filter adjustments must be continuous and incremental, system response must be low-latency, and any state must be immediately reversible. These motivate motorized faders with 1024 units of precision and a pre-computed data cube backend [123].

G6: Facilitate user-driven exploration. Cross-filtering is a direct manipulation paradigm in which the user, not an algorithm, controls the analytical flow [1, 264]. The user's physical manipulation of the filter handles is the direct and sole determinant of what the tactile output displays, with no intermediary statistical, language model, or non-deterministic interpretation.

7.5 Prototype: The Cross-Feelter

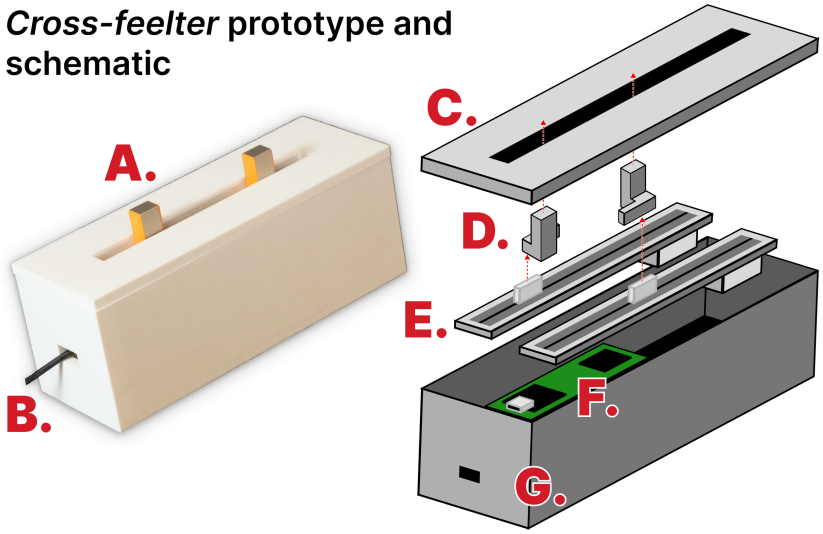


Figure 7.2: A. Our prototype *cross-feelter*. B. USB output (to computer). C. 3D printed rail cover. D. 3D printed knobs that consolidate rails into one. E. 2 motorized linear potentiometers. F. Arduino board. G. 3D printed casing.

We instantiate cross-perception in a hardware prototype we call the *cross-feelter*: a device with two motorized linear faders that provide persistent, absolute tactile input state for brush handles, paired with a refreshable braille display for coordinated tactile output. All hardware schematics, firmware, analysis, and interface code are archived on OSF; 3D-printable STL files for the housing components are available in our supplemental materials on OSF. The total cost of hardware components is $30–50 USD, which we designed intentionally to enable replication by the DIY-AT and maker communities [140, 269].

7.5.1 Interaction Flow

We begin with an idealized walkthrough of what a blind analyst actually does when using the cross-feelter to perform a cross-filtering task. This is the ground truth from which the hardware

design choices derive; we describe the hardware in Section 7.5.2 with reference to the steps below.

Consider an analyst exploring a dataset of local rent prices and transit access across city neighborhoods. Three linked histograms are rendered in sequence on the refreshable braille display. The analyst wants to ask: *do neighborhoods with the highest rents also have better transit access?* At startup, the left knob sits at 0% and the right at 100%, representing an unfiltered view of the full distribution.

Step 1: Feel and filter. The analyst feels the starting state of the tactile data representations on the refreshable braille display, navigating between each chart and hearing supplemental screen reader descriptions alongside the touchable shapes formed from the pins. The analyst moves their hand or hands to the feelter device and slides the left knob rightward, narrowing the filter from below. As they move, an ARIA-live announcement reports the current percentage and knob name via the screen reader, providing precise serial feedback for positioning (G2). The analyst moves the right knob slightly leftward, hearing another announcement. The two knobs now define a range, say 75% to 90% of the rent distribution.

Step 2: Perceive the consequence. During filtering, the analytical environment updates linked histograms immediately on each knob movement. The analyst navigates the braille display to the transit histogram without touching the knobs. Because the knobs are motorized and maintain position, the filter state is preserved and readable by touch at any time (G1). They feel the updated distribution and compare it to the shape they perceived in the unfiltered state.

Step 3: Iterate. The analyst slides the left knob further right, narrowing to the top 10% of rents. The display updates. They feel the new shape. The pattern sharpens or changes, and they iterate again, building toward a hypothesis in the same cycle that sighted analysts perform visually. Each knob adjustment is a computational query: a gesture that constructs a question and immediately exposes its answer.

7.5.2 Hardware Design

The cross-feelter uses two motorized linear potentiometers (commonly called motorized faders in sound design hardware) mounted side by side in a 3D-printed housing (see Figure 7.2). Each fader provides a physical knob on a bounded rail.

Motorized, absolute-position faders (G1, G5). We chose rail-based faders over alternatives (touchpad, rotary encoder, joystick) for two reasons grounded in the G1 requirement. First, the fader is *absolute*: the knob's physical position on the rail directly corresponds to the filter boundary value, with no drift or accumulation. Second, the fader is *motorized*: the system can programmatically set knob position as well as read it. This bidirectionality is essential for cross-filtering: the system must be able to reset filter state on condition changes, counterbalance, and initialization, not only receive input. Prior work on brush-based accessible interfaces [259] used a touchpad, which provides no queryable positional state when the hand is lifted; prior work using non-motorized sliders [329] cannot be programmatically positioned. The motorized fader solves both problems.

Each rail terminates in a tactile boundary, a strong edge inset 4mm from the housing wall per tactile design guidelines [103], so the analyst can feel both the knob position *and* its position relative to each end of the scale (G1). The API reads and writes fader position every 5ms,

providing up to 1024 units of precision, which maps well to histogram bin resolutions in practical cross-filtering datasets (G6).

Merged-rail housing (G3). We designed two housing configurations: one exposing the rails independently, and one using angled knobs to place both rails along a single track (Figure 7.2, part D). The merged-rail design, used for all data tasks in the study, ensures the left knob remains to the left of the right knob at all times (they cannot cross over). This preserves the functional semantics of a spatial filter range: left boundary is always left, right boundary is always right, matching the computational filter and the visual histogram (G3). The independent-rail configuration was used for the video interaction training task (Section 7.6.4), where the two rails served different functions (volume and position).

Our dual-fader design is similar in form to motorized fader-based systems proposed for axis interaction in AR/VR data environments [55, 268]; however, those systems presuppose sighted users operating in 3D visual contexts. Our design differs in three ways: (a) the housing constrains the two faders to a single interaction dimension to support filter semantics; (b) the motorized positioning enables two-way synchronization with the data interface; and (c) the entire system is designed for eyes-free operation as the primary use mode.

ARIA-live auditory feedback (G2). When a knob is moved, the interface emits an ARIA-live announcement to the screen reader reporting the new filter percentage. ARIA-live is set to `polite` mode, meaning announcements queue behind any active screen reader speech rather than interrupting it. This provides precise, serial auditory feedback for filter positioning without competing with other auditory streams in the environment (G2). Tactile feedback (the knob's physical position) and auditory feedback (the ARIA announcement) thus serve complementary roles: tactile provides continuous, queryable approximate state; auditory provides precise on-demand readout triggered by movement.

Electronics and firmware. Figure 7.3 provides the full wiring of the device, sufficient for a reader to replicate it. Each motorized fader (a Behringer MF100T: a 100mm fader with a 10kΩ linear resistive track, a capacitive touch-sense track, and a small DC motor) exposes six terminals: the two fixed ends of the resistive track, the position wiper, the touch line, and the two motor terminals. A Teensy 4.0 microcontroller, running Arduino-style firmware, reads and drives both faders. The track ends connect across the Teensy's 3.3V supply and ground, so each wiper reports absolute knob position as a 10-bit analog value (0 to 1023) on pin A0 (rail 0) or A2 (rail 1). The touch lines connect to pins A1 and A3. The firmware samples them every 5ms and counts readings at the extremes of the analog range (below 50 or above 973) over a rolling ten-sample window; four or more extreme samples within the window register as a touch. This works because a finger on the conductive knob couples body capacitance and ambient electrical noise into the otherwise stable touch track, swinging successive samples toward the rails. Touch state gates the motors: any pending motor target is cancelled the moment a knob is touched, so the device never fights the analyst's hand (G6).

The motors are driven through a DRV8833 dual H-bridge motor driver: Teensy digital pins D2 and D3 control rail 0 (up and down respectively), pins D4 and D5 control rail 1, and the driver's output pairs connect to the two motors. The driver's motor supply takes 5V from the Teensy's USB power pad, so the entire device is powered and controlled over a single USB cable. The firmware runs a 1ms control loop that drives an actively commanded motor toward its target until the wiper reads within a small tolerance band, and a 50ms reporting loop that emits

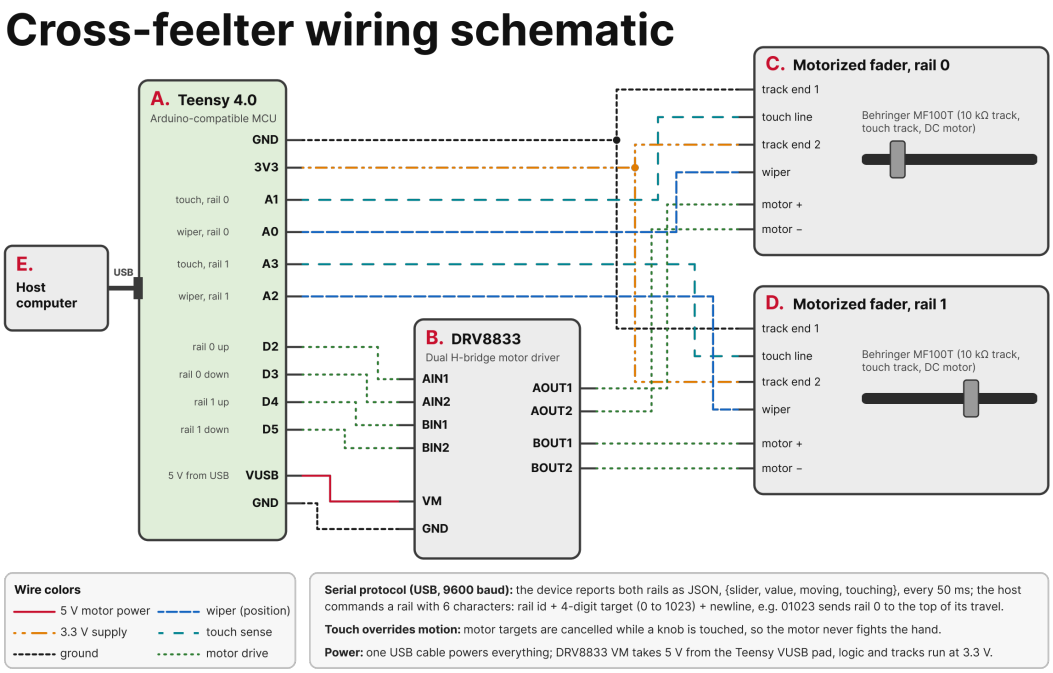


Figure 7.3: Wiring schematic of the cross-feelter. A. Teensy 4.0 microcontroller. B. DRV8833 dual H-bridge motor driver. C. Motorized fader, rail 0. D. Motorized fader, rail 1. E. Host computer, connected over USB. Wire colors distinguish 5V motor power, the 3.3V supply, ground, wiper position lines, touch-sense lines, and motor drive lines.

the state of both rails (position, moving, and touching) as a compact JSON message over a 9600 baud serial connection. Host-to-device commands are six characters: a rail identifier, a four-digit zero-padded target position, and a newline. For example, `01023` sends rail 0 to the top of its travel.

7.5.3 Analytical Environment

The cross-feelter operates within a web-based analytical environment purpose-built for this study (Figure 7.4). The environment presents three linked histograms derived from different variables in a shared dataset. Each histogram is rendered both visually (for debugging and sighted collaborators) and as a 60×40 pin array sent to the Dot Pad refreshable braille display.

Cross-filtering backend (G4, G5). We use Mosaic [123] to manage the cross-filtering logic. Mosaic pre-aggregates filter results across binned input values, enabling fast, low-latency responses to filter changes even over larger datasets (G5). It also provides a straightforward API for specifying a brush-filter interaction layer over a histogram dimension, including programmatic control of filter state and undo actions (G5). When the user moves a fader knob, the new filter boundaries are sent to Mosaic, which immediately updates the linked histograms, which are then re-rendered on the braille display. The render cycle from fader movement to updated tactile output is 500–1200ms on the Dot Pad hardware, fast enough to support iterative exploration,

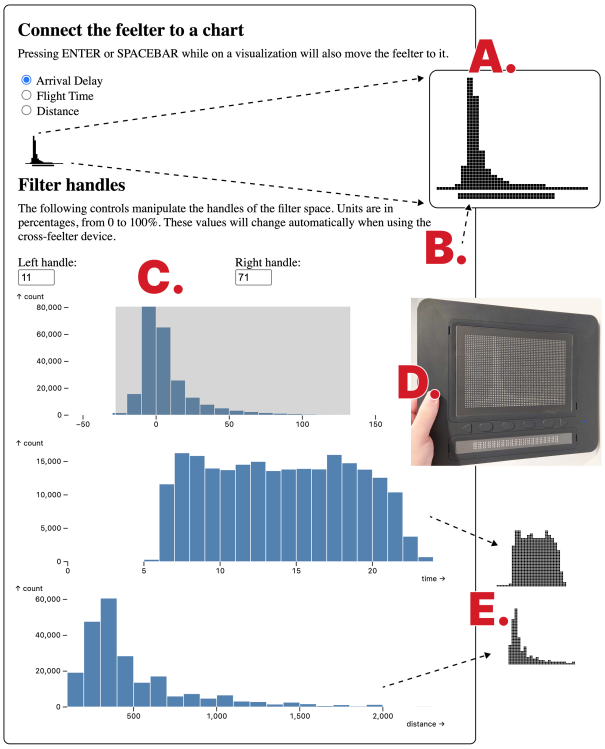


Figure 7.4: Our analytical environment. A. a 1 to 1 translation between visual and tactile that represents a 60x40 pixel-to-pin array. B. A tactile version of the filter location (if on the chart being focused). C. Cross-filtering controls, including text inputs for screen-reader only manipulation. D. Refreshable tactile display output. E. Example renderings of the other 2 charts.

though slower than visual cross-filtering systems and a recognized limitation (Section 7.10).

Software architecture. The analytical environment runs entirely in the browser. It connects to the cross-feelter through the Web Serial API, filtering on the microcontroller's USB vendor identifier so that no driver installation is required, and polls the device's serial reports every 25ms. When a report shows a knob that is both moving and touched, the interface maps the raw fader value through a clamped linear scale onto the pixel range of the corresponding Mosaic brush handle and applies it by synthesizing native pointer events on the handle itself. The fader therefore drives exactly the same interaction pathway as a mouse drag on the visual brush, which keeps the tactile and visual input methods behaviorally identical by construction. Mosaic propagates the updated selection as a cross-filter, DuckDB-WASM re-aggregates the data in the browser, and all three linked histograms re-render. On each update, the focused histogram is redrawn onto an internal 60×40 canvas, including a low horizontal bar marking the current brush extent, packed into the Dot Pad's cell-based packet format, and sent over Web Bluetooth; unchanged rows are diffed out so that only modified lines of pins are refreshed. Programmatic motor control flows in the other direction: when the analyst attaches the feelter to a different chart, the interface writes positioning commands over the same serial connection to drive both knobs to the ends of their

rails, resetting the physical filter state to match the new chart's unfiltered view.

Braille display output (G2). The Dot Pad provides a 60×40 binary pin array (pins raise or lower on command). We render each histogram as a bitmap at this resolution, trading fidelity for speed relative to a high-resolution braille embosser (which takes 3–4 minutes per page). At any time, the braille display shows one chart; the analyst navigates between charts using the screen reader. The visual representation of the target pin array is visible to sighted observers and collaborators in the environment (Figure 7.4, part A), illustrating the resolution gap that future advances in refreshable display technology could close.

Screen reader compatibility. The analytical environment is a standard web interface, accessible via any JAWS, NVDA, or VoiceOver-compatible screen reader. Cross-filter input controls are exposed as labeled range inputs so they can be operated by screen reader alone (providing the baseline condition for our study, Section 7.6). The ARIA-live region for fader announcements is separate from these controls and does not interfere with standard screen reader navigation of the interface.

7.6 Data Exploration Study

We conducted a within-subjects study comparing cross-feelter-based cross-filtering (CF condition) against screen-reader-based cross-filtering (SR condition) across structured and open-ended data exploration tasks. Sessions were 90 minutes, conducted in person, and approved by our IRB.

7.6.1 Study Design

Overview. Our study employed a 2×2 Latin square design, fully counterbalanced across two factors: (1) condition order (CF-first vs. SR-first) and (2) dataset order (flight data vs. neighborhood data), producing four participant groups (Table 7.1).

Table 7.1: 2×2 Latin square study design. CF = cross-feelter condition; SR = screen reader condition.

Group	Condition order	Dataset order
A (n=4)	CF → SR	Flight → Neighborhood
B (n=4)	SR → CF	Flight → Neighborhood
C (n=4)	CF → SR	Neighborhood → Flight
D (n=3)	SR → CF	Neighborhood → Flight

Within each condition, participants performed two tasks on the same dataset: a 3-minute structured data-relationship task followed by a 10-minute open-ended data exploration. Counterbalancing both orders controlled for dataset familiarity, learning effects, and dataset-specific task difficulty.

Conditions. In the **CF condition**, participants used the cross-feelter (merged-rail housing; Section 7.5.2) to manipulate cross-filter bounds, with the Dot Pad displaying linked histograms. In the **SR condition**, participants used their screen reader to operate labeled text input fields

exposing the same filter controls, with the same Dot Pad display. The analytical environment was identical across conditions; only the input method differed.

Session procedure. Each session followed this order regardless of group:

1. Tech setup, baseline interview, introduction to tactile histograms and cross-filtering, baseline anxiety Likert question *(20 min)*
2. Device familiarization: video interaction task and brief semi-structured interview *(15 min)*
3. First data condition: structured task, open-ended exploration, Likert-scale interview *(20 min)*
4. Second data condition: structured task, open-ended exploration, Likert-scale interview *(20 min)*
5. Closing semi-structured interview *(15 min)*

7.6.2 Participants

We conducted our study with 15 blind participants: 8 without prior professional data expertise and 7 with it (data analysts, scientists, or engineers currently or recently employed in data-related roles). 7 participants identified as male, 8 as female.

Participants without data expertise, plus 2 with it, were recruited locally through community outreach. The remaining 5 participants with data expertise were recruited through professional networks and coordinated at mutual academic conferences, due to difficulty recruiting blind data scientists locally. All participants were screened for eligibility (blind, screen reader proficient, 18 or older, US citizen or permanent resident), informed of risks and voluntary participation, and compensated with a $50 gift card.

7.6.3 Datasets

Study tasks used two datasets selected through a community data-preference workshop. Pilot sessions revealed that participants were not engaged by our initial generic dataset, so we held a workshop at a local braille and talking book library (n=21 community members). Participants voted on datasets from topics including local politics, housing, and community life. The two highest-voted were: (1) rent prices across city neighborhoods, and (2) fish fry locations across city neighborhoods (a dataset reflecting a regional Catholic Lent food tradition observed broadly across denominations). Both are publicly available. A generic flight dataset was included as the second counterbalanced dataset.

Each dataset was configured with three histogram variables from a shared source, with cross-filter tasks designed to target a 75%–100% filter range producing a clear, unambiguous relationship between variables.

7.6.4 Device Familiarization: Video Interaction Task

Rationale. Before data tasks, participants completed a device familiarization activity using the cross-feelter to control a video feed rather than a data chart.

The goal was to build fluency with the motorized fader mechanism (absolute-position input, motorized output, the relationship between physical knob position and system state) without pre-exposing participants to the cross-filtering metaphor, data tasks, or the study datasets. A practice task using data histograms would have constituted condition-specific training, introducing learning effects that could confound the within-subjects comparison. Video scrubbing and volume control are familiar and intuitive interaction goals for all of our users, which served as a way for all of them to understand how to operate the device.

Task. Participants were asked to find the name of the person in the video recording, announced early in the clip. This simple information-retrieval goal required genuine device use while keeping cognitive demand low before the main study.

7.6.5 Introduction to Tactile Histograms and Cross-Filtering

At session start, after consent and the baseline interview, we provided a scripted introduction to tactile histograms and cross-filtering. We explained histogram structure, introduced the concept of linked histograms sharing a source dataset, and demonstrated cross-filtering with three histograms from a fruit example dataset, using two sheets of paper held over one histogram to physically simulate a spatial filter.

We framed filtering explicitly as an analytical activity: "filtering is a way to ask questions more than to find answers, to look for possible relationships that might exist in the data." Two follow-up questions gauged comprehension and interest before proceeding.

7.6.6 Data Tasks

7.6.6.1 Structured data-relationship task

Participants cross-filtered the dataset and answered a three-option multiple-choice question within three minutes. The question and options were available via screen reader in the interface; the facilitator restated them only if asked and did not otherwise intervene.

For the flight dataset: *"What is the relationship between filtering for longer flight distances and the arrival time of flights?"* (Correct: longer flights tend to arrive earlier compared to all flights.)

For the neighborhood dataset: *"What is the relationship between filtering for a higher number of fish fry locations and the rent price of a neighborhood?"* (Correct: neighborhoods with more fish fry locations tend to have higher rent prices.)

We measured task completion rate (binary), time to completion (seconds), filter placement precision (distance of handles from target positions of 75% and 100%, scored in $\pm 4\%$ increments awarding 0.5/0.3/0.1 points per handle tier), and answer accuracy (binary).

7.6.6.2 Open-ended data exploration

Participants then performed a 10-minute open-ended exploration of the same dataset using a prompted, concurrent think-aloud method [7]. Before beginning, we framed the activity: "the

purpose is not to *answer* questions but to *come up with* questions. This is how many data scientists explore unfamiliar datasets, by looking for possible relationships."

We measured: (1) *computational queries*: any filter manipulation followed by braille display inspection of one or more linked charts (filter changes without output inspection did not count); and (2) *spoken queries*: any data-related question voiced aloud, regardless of formality or answerability via cross-filtering.

7.6.7 Measures and Analysis

Quantitative analysis. Objective measures (completion rate, time, precision, accuracy, query counts) and subjective measures (5-point Likert scales for stress, cognitive load, future-task anxiety, and enjoyment) were collected per condition. A pre-study baseline anxiety item provided a pre-condition reference.

Analysis was conducted in Python (code and de-identified data in supplemental materials). Objective continuous measures were compared using paired t-tests. Likert-scale differences were evaluated using Wilcoxon Signed-Rank tests given the ordinal nature of the data. We report means, standard deviations, test statistics, and p-values; group-level results (data experts vs. non-experts) are reported separately where they diverge.

Qualitative analysis. Qualitative data comprised session audio/video recordings, facilitator notes, and closing semi-structured interview transcripts. The interview covered three topics: overall experience and condition comparison, interface and environment feedback, and future application ideas.

Analysis proceeded in two stages. In the first stage, two members of the research team independently reviewed all recordings, notes, and transcripts, generating initial codes at the utterance level. Codes captured themes across four domains: device usability (physical interaction, feedback quality, learning curve); condition comparison (speed, cognitive load, perceived control); analytical engagement (question generation, hypothesis articulation, confidence with data); and future use speculation. The two coders then resolved all disagreements through discussion, reaching consensus before proceeding.

In the second stage, codes were organized using affinity diagramming [119], conducted collaboratively by members of the research team working from the reconciled code set. Resulting themes inform the qualitative findings in Section 7.7 as well as our future use cases in Section 7.9.

7.7 Results

Our analysis compared 15 blind participants performing cross-filtering tasks under two conditions: using the cross-feelter (**CF**) versus using a screen reader alone (**SR**). We report objective and subjective measures in turn. Full analysis code and de-identified data are available in our supplemental materials.

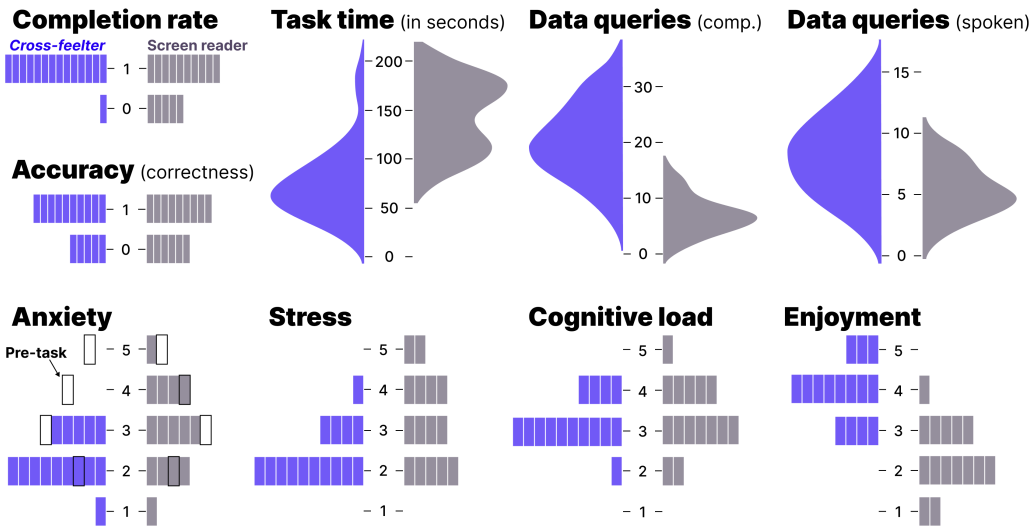


Figure 7.5: Results. Completion rate and accuracy are counts. Task time, data queries (computational), and data queries (spoken) are objective, observational measures. Anxiety, stress, cognitive load, and enjoyment are Likert-scale results from 1 ("very little") to 5 ("quite a lot").

7.7.1 Objective Results

7.7.1.1 Substantially improved task completion rate and speed

For the structured data-relationship task, participants achieved a significantly higher completion rate with CF (mean = 0.93, SD = 0.26) compared to SR (mean = 0.60, SD = 0.51; paired t-test: $t = 2.646$, $p = 0.019$). Task completion time was significantly lower in the CF condition (mean = 75.53s, SD = 36.14) than in the SR condition (mean = 143.93s, SD = 34.65; $t = -6.876$, $p < 0.001$), a 90% improvement in speed.

7.7.1.2 Comparable accuracy and precision

Accuracy (correct multiple-choice answer) was higher in the CF condition (mean = 0.67, SD = 0.49) than SR (mean = 0.53, SD = 0.52), but this difference was not statistically significant ($t = 1.468$, $p = 0.164$). Notably, when filtering for only completed tasks, SR users showed marginally higher accuracy (SR: mean = 0.78; CF: mean = 0.71), though again non-significant ($t = 0.555$, $p = 0.594$). We interpret this in Section 7.8.2.

Filter placement precision (scored by proximity to target handle positions of 75% and 100%) was marginally higher in CF (mean = 0.66, SD = 0.37) than SR (mean = 0.59, SD = 0.34), with no significant difference ($t = 0.617$, $p = 0.547$).

7.7.1.3 Substantially increased quantity of data queries

During the open-ended exploration task, CF elicited substantially more computational queries (mean = 20.13, SD = 6.45) than SR (mean = 7.00, SD = 3.18; $t = 9.567$, $p < 0.001$), a 188%

increase. Spoken queries were also significantly higher in CF (mean = 8.20, SD = 3.14) than SR (mean = 5.33, SD = 1.99; $t = 3.765$, $p = 0.002$), a 54% increase.

7.7.2 Subjective Results

Subjective measures varied between participants with and without professional data expertise; we report group-level differences where they diverge meaningfully.

7.7.2.1 Reduced stress

Stress ratings were significantly lower in the CF condition (mean = 2.40, median = 2.0) than SR (mean = 3.20, median = 3.0; Wilcoxon $W = 4.000$, $p = 0.013$). The effect was driven primarily by participants without data expertise (CF: mean = 2.62, SR: mean = 3.75; $W = 2.500$, $p = 0.047$). Data experts reported similarly low stress across both conditions (CF: mean = 2.14, SR: mean = 2.57; $W = 0.000$, $p = 0.083$).

7.7.2.2 Comparable cognitive load

Cognitive load showed an observed difference in means (CF: mean = 3.13, SR: mean = 3.47) but was just outside statistical significance ($W = 4.000$, $p = 0.059$). We interpret both conditions as producing comparable cognitive load during task performance.

7.7.2.3 Lower future-task anxiety

Participants in the CF condition reported significantly lower anxiety about performing a future timed data task using the same tools (CF: mean = 2.27, SR: mean = 3.00; $W = 10.000$, $p = 0.017$). CF anxiety was also significantly lower than participants' baseline anxiety measured before any tasks ($W = 5.000$, $p = 0.003$), while SR anxiety was not ($W = 8.000$, $p = 0.132$), suggesting the cross-feelter reduced anticipatory anxiety relative to participants' pre-study state, while the screen reader did not.

7.7.2.4 Substantially improved enjoyment

Enjoyment was substantially higher in CF (mean = 3.93, median = 4.0) than SR (mean = 2.33, median = 2.0; $W = 0.000$, $p = 0.001$). This effect was strongest for participants without data expertise (CF: mean = 3.875, SR: mean = 2.000; $W = 0.000$, $p = 0.015$). One participant's unprompted remark captures the qualitative texture of this finding: *"Sighted people get to have this much fun when they work with data?"*

7.8 Discussion

Our findings suggest that cross-perception shows meaningful promise as an approach to accessible analytical data interaction. We discuss our results in relation to our four research questions, then develop the broader implications for the field.

7.8.1 Research Question Responses

R1 (Exploratory): How might we envision and formalize a tactile interaction approach that enables coordinated, non-visual interaction across multiple spaces? Cross-perception, as formalized in Section 7.4, provides one answer: an interaction technique grounded in principles from visual cross-filtering, spatial cognition, and blind information interaction, instantiated in a device with persistent-state motorized input and a linked tactile display.

The six design goals (G1–G6) constitute the formalization, and the design vignette demonstrates that the underlying cognitive mechanisms are ecologically grounded in expert blind reading practice. We intend this formalization to be extensible and generative; we demonstrate this extensibility in our future use cases in Section 7.9.

R2 (Quantitative: Objective): In what measurable ways can we observe improvements to blind data interaction using our tactile interaction approach? Cross-perception produced significant improvements in task completion rate (+55% relative), task completion speed (90% faster), and quantity of data queries during open exploration (+188% computational, +54% spoken). Accuracy and precision were statistically comparable between conditions, though with a nuance discussed in Section 7.8.2 below.

R3 (Quantitative: Subjective): What are the self-reported benefits, opportunities, and challenges that blind people experience with our tactile interaction approach? Participants using the cross-feelter reported significantly lower stress, lower future-task anxiety, and substantially higher enjoyment than in the screen reader condition. These improvements were particularly pronounced for participants without data expertise, suggesting cross-perception may lower affective barriers to data work for blind users who are new to or learning data analysis.

R4 (Qualitative): In what ways do blind data scientists imagine extending our approach? Participants, especially those with professional data expertise, proposed substantive extensions including: a pre-rendered "interaction cube" for environments without a refreshable display; feelter-only navigation of data arrays and sequences; sonification paired with rail brushing; and multi-dimensional panning, zooming, and scatterplot brushing using multiple feelter units. These proposals are detailed in Section 7.9.

7.8.2 Accuracy and Precision Considerations

The comparable accuracy and precision results merit careful interpretation. Participants maintained similar correctness across conditions while performing substantially faster in CF. However, when filtering for completed tasks only, SR users showed marginally higher accuracy. One interpretation is that the additional time the serial screen reader interface imposed allowed for more deliberate verification before committing to an answer, while the cross-feelter's speed encouraged more rapid, and occasionally less careful, decision-making.

This is not necessarily a weakness of cross-perception. In genuine exploratory analysis, speed of iteration is often more valuable than precision on any individual query, since the point is to generate and test many hypotheses rather than verify a single one carefully. Some precision can be a little "fuzzy" if the gained effect has analytical benefit overall [149]. This difference in interaction patterns warrants further investigation and suggests the two approaches may support different analytical strategies rather than being simply faster and slower versions of the same

strategy.

Future iterations should explore mechanisms that preserve the speed benefits of direct tactile manipulation while supporting careful verification when the task demands it (for example, a confirmation gesture or a "hold" mode that locks the filter and prompts output inspection before advancing).

7.8.3 The Case for Analytical Interaction as a Primary Research Site

The result most significant for the field is not the speed improvement but the query count. 20 computational queries in 10 minutes compared to 7 is not merely a faster version of the same activity. This increase in speed also reflects a qualitative shift in what participants were doing. In the CF condition, participants iterated hypotheses through direct manipulation; in the SR condition, they navigated toward output values that were structurally separate from their space of interaction. This maps directly to the distinction between analysis-oriented and access-oriented interaction: the screen reader, however capable at navigation, kept participants in a mode that structurally separated interaction and information; a descriptive mode. The cross-feelter enabled a *generative* mode where interaction and information are coupled.

The enjoyment gap is legible through the same lens. One participant's unsolicited remark, *"Sighted people get to have this much fun when they work with data?"* describes the experience of analytical agency. This is what visualization researchers have understood for decades to be the core value of interactive exploratory analysis: an experience of power over and with data interaction. This agency has not been consistently available to blind data practitioners, and the affective signature of its absence is measurable in our stress and anxiety results.

The implication is that analytical interaction accessibility constitutes a distinct research problem, not an extension of the accessible rendering/description problem. The accessible visualization community has made substantial progress on helping blind users read and navigate data. What our results suggest is that reading data and analyzing data are sufficiently different activities that the techniques developed for one cannot simply be adapted for the other: they require different interaction architectures, different evaluation approaches, different design primitives, and sometimes even different hardware. We encourage researchers working at this intersection to treat analytical interaction as a primary site of inquiry: studying the existing practices of blind data analysts during open-ended exploration, identifying what analytical operations (filtering, outlier detection, hypothesis iteration, assumption testing) current tools support or foreclose, whether existing tools support direct interaction with data representations in these tasks, and designing interaction techniques grounded in these blind-centered findings, rather than in adaptations of what sighted-user interfaces already provide.

7.8.4 Toward Hardware and Physicalization in Accessible Data Interaction

Building on that previous point, a recurring practical constraint in accessible data interaction research is the emphasis on solutions that work within existing consumer device ecosystems: screen readers, keyboards, web browsers, and computationally synthesized audio. This is rea-

sonable: it maximizes immediate reach and avoids asking blind users to acquire specialized equipment if the capabilities are already realized on existing computer and mobile hardware. But it also implicitly constrains the design space to what those platforms can support, and some of the interaction properties that matter most for analytical work (such as persistent physical state, absolute spatial position, simultaneous bilateral manipulation) are not well-approximated by software alone on current consumer hardware. Existing work (mentioned in Section 7.3) hasn't ignored this problem, but this existing work is a small minority. Priorities in research have predominantly focused on leveraging *existing* software, hardware, and computational models towards problems and barriers blind people face, rather than innovation towards *novel* hardware and physicalization systems.

The cross-feelter's component cost ($30–50 USD) and the availability of all schematics under an open license suggest that the hardware barrier is lower than it might appear. The more substantive barrier is that the visualization and accessibility communities have not yet developed strong shared practices around hardware prototyping and physical interaction design. Work from the tangible and physicalization communities is directly relevant here, including the insight that physical materials afford interaction properties that screens do not, among them permanence, resistance, and the kind of spatial memory that our design vignette illustrates. Bringing these ideas into contact with accessible data interaction research is, we believe, a productive direction.

The future use cases our participants proposed, discussed in detail in Section 7.9, illustrate the range of what becomes thinkable when the design space is extended beyond the screen: pre-rendered interaction outputs for environments without powered displays, motor-driven data array navigation, multi-axis spatial exploration using multiple feelter units, and sonification navigated through physical rail position. These proposals emerged from a structured speculative futuring exercise [70] with participants who are themselves blind data practitioners. They are worth treating as design starting points, not only as participant feedback.

The field has successfully applied careful empirical and design methods to the challenge of making non-visual data representations perceivable. Extending that rigor to the challenge of making data manipulable through the hands, with all the hardware and prototyping work that entails, is the direction we are advocating.

7.9 Future Use Cases

The following use cases were developed through the speculative futuring exercise [70] conducted as part of our closing interview, with contributions from participants and the research team. We present them as design starting points grounded in the cross-perception framework, each addressing a limitation of the current prototype or extending the approach to new contexts. Figure 7.6 illustrates all four cases.

7.9.1 Cross-Perception Without a Powered Display: The Pre-Rendered Interaction Cube

Refreshable braille displays remain expensive ($8k–$20k USD) and are not universally available. The current prototype depends on the Dot Pad to render cross-filter output in real time, which

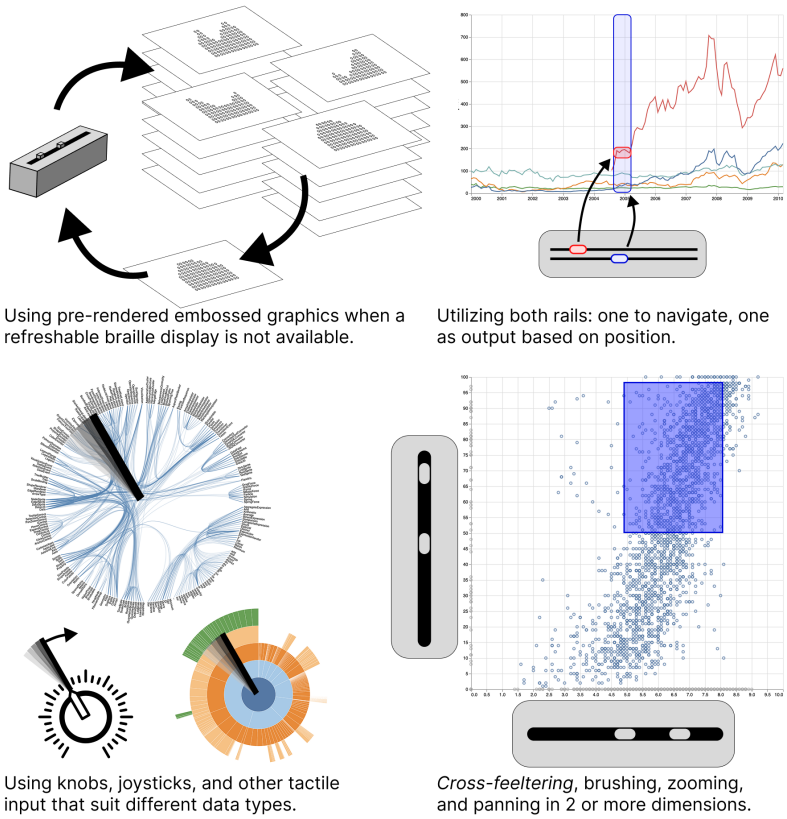


Figure 7.6: Speculative future use cases for our *cross-perception* approach, contributed by our participants.

means the full cross-perception loop is unavailable to users without access to one.

One solution is to pre-render the interaction outputs for a given dataset before a session begins. The cross-feelter's filter operates over a bounded, discretizable range; it is feasible to pre-compute a set of interaction results at binned filter positions and emboss them onto paper as an "interaction cube" of tactile states, analogous to the data cube that Mosaic pre-computes for fast cross-filtering over large datasets [123]. After each filter manipulation, the user turns to the corresponding page in a booklet of embossed tactile charts, reducing the interaction loop from 3–4 minutes (a full embosser render) to roughly 10 seconds. This approach trades the continuous, real-time update of the powered display for a discrete but physically permanent output, which has its own advantages for users who prefer to annotate or revisit prior states or even compare multiple outputs simultaneously, like in the dual-task comparison vignette in Section 7.4.1.

7.9.2 Sequential Data Navigation with Continuous Positional Feedback: Feelter-Only Chart Interaction

Serial navigation of data arrays using a screen reader is slow because the user must step through observations one at a time without any persistent sense of their position within the full range [77].

A single cross-feelter could replace this with continuous physical scrubbing. Most charts and

datasets have finite, bounded observations in any given dimension; these bounds map naturally onto the full range of a fader rail. One rail would scrub through the ordered observations as the user moves the knob, with the screen reader announcing the current value. A second rail, motorized, would move its knob to a position encoding the corresponding value in a second data dimension, like a rising and falling bar in a bar chart.

More broadly, different input devices could be matched to different data topologies. A rotary dial would suit circular or periodic data (seasonal patterns, directional data); a joystick would suit data with an infinite or continuous range such as a parametric function or model output.

7.9.3 Sonification Navigation via Physical Rail Position

Navigating within a sonification (seeking to a specific region, comparing two positions, returning to a reference point) is currently difficult even on touch-enabled devices, because scrubbing through audio provides no queryable positional state when the hand is not actively in contact with the surface [50, 51, 52]. This is the same object-permanence problem that motivated the cross-feelter's design for cross-filtering.

A cross-feelter rail could serve as the navigation input for a sonification: physical position on the rail maps to a position within the audio timeline or data domain, and the motorized mechanism maintains that position when the hand is lifted. The user can navigate to a region of interest, remove their hand, let the sonification play, return to the rail, and feel exactly where they left off.

7.9.4 Multi-Dimensional Spatial Interaction: Rails as Input and Output

Two cross-feelter units, one arranged as an x-axis controller and one as a y-axis controller, could support spatial panning, zooming, and brushing over tactile maps or scatterplots, which are interactions that currently lack a satisfactory non-visual implementation because pinch-and-zoom gestures provide no queryable positional state for the zoom extent or pan position. Each feelter unit would maintain the current viewport bounds in its two rails, readable by touch at any time.

More substantially: the motorized rails could serve as *system output* for spatial queries. When a user asks "where is this object?" of a map or image, the system could drive the rails to the corresponding x and y positions and bounds. This would let the user feel the location directly rather than receiving a verbal coordinate and build a spatial mental model of a spatial layout.

7.10 Limitations

Our study's controlled task design (three-minute timed tasks with multiple-choice answers) prioritized speed and query quantity, which are the right measures for a proof of concept but do not capture the full texture of blind data analysts' real exploratory workflows. Activities common in open-ended analysis, such as wrangling messy data, identifying outliers, and questioning assumptions, were not represented. An ethnographic or longitudinal study would be better suited to those questions, and we encourage that work.

The Dot Pad's 500–1200ms render latency falls outside the max 500ms threshold at which Liu and Heer identify measurable degradation in exploratory analysis quality [173]. Our query

count results are therefore a conservative lower bound on what cross-perception could support with faster display hardware.

Novelty effects may have inflated subjective measures. The enjoyment and anxiety improvements may attenuate with extended exposure, though the effect may persist in contexts such as some classroom, work, or onboarding settings, where engagement is itself valuable.

Finally, we compared against a screen reader baseline because screen readers are the dominant interaction tool for blind users working with data and all of our participants were proficient in screen reader use. More granular comparative work against other emerging tactile and voice-based input strategies would further characterize where cross-perception sits in the broader interaction space.

7.11 Conclusion

We introduced cross-perception, a tactile interaction technique that adapts cross-filtering for blind data analysts by preserving the simultaneity of input and output perception that makes the technique analytically valuable. We formalized it through six design goals grounded in principles from visual analytics, spatial cognition, and blind information interaction; instantiated it in the *cross-feelter*, a low-cost motorized fader device; and evaluated it against a screen reader baseline with 15 blind participants.

In our study, cross-perception substantially improves task completion speed, data query generation, and subjective experience, with particularly strong effects for participants without prior data expertise. The result that matters most is not the speed improvement but the query count: participants using the cross-feelter generated nearly three times as many analytical queries during open exploration, reflecting both a quantitative improvement and a qualitative shift away from navigating data toward interrogating it.

We argue that analytical interaction accessibility, the capacity for blind users to filter, conjecture, and test hypotheses through direct manipulation, is an open research problem that deserves sustained attention as a primary site of inquiry, distinct from accessible rendering. The future use cases our participants proposed suggest that extending the design space to include hardware and physicalization opens directions that software-only approaches cannot reach.

7.12 Postface

This project has been framed and reframed several times: should we focus on the prototype? the study? or the interaction design?

Ultimately, the story we want to tell with this research isn't about our *cross-feelter* prototype (despite how much we love its name) or the study (despite how much we love some of our findings). But ultimately, we hope to see more work that takes the core idea here (that *cross-perception* enables new kinds of interaction with information), which is a generative space for new research [16]. We ran speculative sessions with users precisely because we wanted to explore whether the simplicity of our core idea could lead to new directions. And it did.

Often, research papers are defensive about intellectual territory. Papers frame "future work" as a zone of intended control, a domain that the primary author hopes to protect. Some papers are generous in their future work sections, giving new ideas and directions away freely, but most are situated on the style of narrative that the main author intends to explore certain paths forward, perhaps even to the discouragement of others.

Yet research ideas should be generous. If researchers didn't have to worry about competitive grants or the pressures of tenure requirements, the spirit of academia might shine brightest in a paper's future work section. To us, this is where new pathways should be freely given, new directions should be inspired, and unanswered questions are earnestly laid bare. We hope that this is the case with this project, in particular. We want to see more hardware in accessible data interaction. Visualization as a field is situated uniquely to answer this call, too. There are already innovators in hardware and input modalities in our research community. But many haven't turned their gaze towards accessibility. We hope to capture some of their imaginations, and see more work that builds on *cross-perception* in the future.

Part V

Personalization: System-building for User Agency

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

8.1 Preface

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 participants' 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.

8.1.1 Context

As mentioned in the opening of this thesis, our work in practical contexts applying *Chartability* became what motivated the main categories of work that organize this thesis. *Personalization* is motivated by *Chartability*'s heuristics that center on the *Flexible* principle: the tests that can be done to determine how much an end user is able to adjust text, colors, sizing, zoom, and presentation according to their needs. This was easily the principle with the highest rates and severity of failure out of our tests applying *Chartability* in practice. It is barely handled by any visualization application, tool, or library at even a minimum level.

However, *Softerware* is relatively unique compared to other projects. Unlike *Data Navigator*, *Skeleton*, and *Cross-perception*, we, in addition to our soon-to-be collaborators at Highsoft and Elsevier, realized that a singular study or prototype would likely not be sufficient to enact change at the scale that real personalization requires. We wanted, first and foremost, not to bring new knowledge into the world but to ultimately demonstrate to our peers that our field needs significant attention and change in the area of personalization.

Of course, there is plenty we don't know about how people with disabilities might want to personalize visualizations. So there *is* a reasonable gap in knowledge that a proper human studies approach would address, even without focusing on system-level concerns and prototype building. But Highsoft's existing visualization library *Highcharts* is already, according to comparative analysis that builds on *Chartability*, the most-accessible library out there [162]. And Highsoft was interested in continuing their tradition of pushing the boundaries of accessibility in hopes that peers would follow suit. And they were also interested in how exactly we might imagine

not only their ecosystem evolving to better support a *softerware* approach, but ways that libraries and tools could also share an infrastructure of user preferences as well.

So we set out to engage this complex space of priorities together, which is held in tension with the traditional aims of a lower-risk, smaller-in-scope sort of research project: clearly define a singular gap and attempt to fill it. We wanted to contribute research that fills a gap, demonstrate how one might apply that knowledge in their own practical, real-world industry environment, and then use this work as a call to action for fellow visualization system builders and tool-makers to follow suit. Thankfully, Computer Graphics and Applications had a call for a special issue on *Inclusive Data Experiences*. CG&A has a philosophy that invites both academic and non-academic readership, which was a perfect venue for us.

This chapter was adapted from our 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*).

8.2 Overview

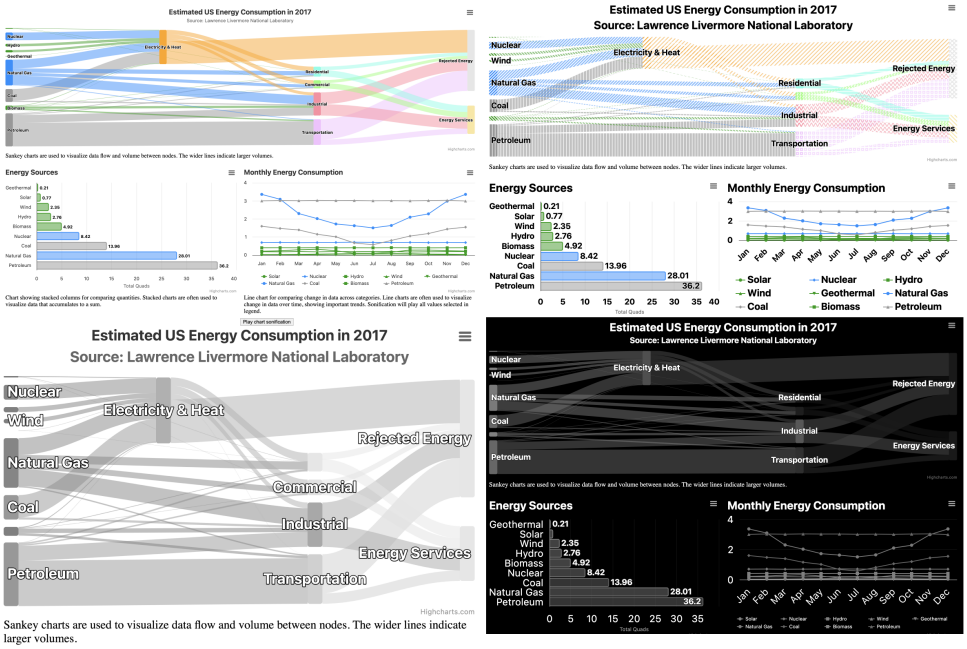


Figure 8.1: Sometimes one design is not enough. Our design (upper left) and three different designs by low vision users. All low vision users chose larger text, but then diverged: redundant-encoding enabled (upper right), high zoom and greyscale on white (bottom left), and then dark mode (enabled externally)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

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 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 [128].

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 adjust, 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 [115, 138]. 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 [191, 317], there are growing resources on areas such as cognitive/intellectual disabilities [322, 324], neurodivergence [291], and both research and systems exploring epilepsy and vestibular/motion inaccessibility in visualization [266, 272].

Yet despite these resources, making data visualizations more accessible remains a difficult task for practitioners [153, 258]. 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 [244] while another stresses that redundant encoding strategies are necessary [76]. 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* [320], 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 instead 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 [326]. 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 [218]. 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 [151]. 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 [180], 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 [323] 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 [39, 164].

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 [127], where *fit* refers to a system that is perfectly complementary 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).

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 [323]. 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 8.2) 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 viewing 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

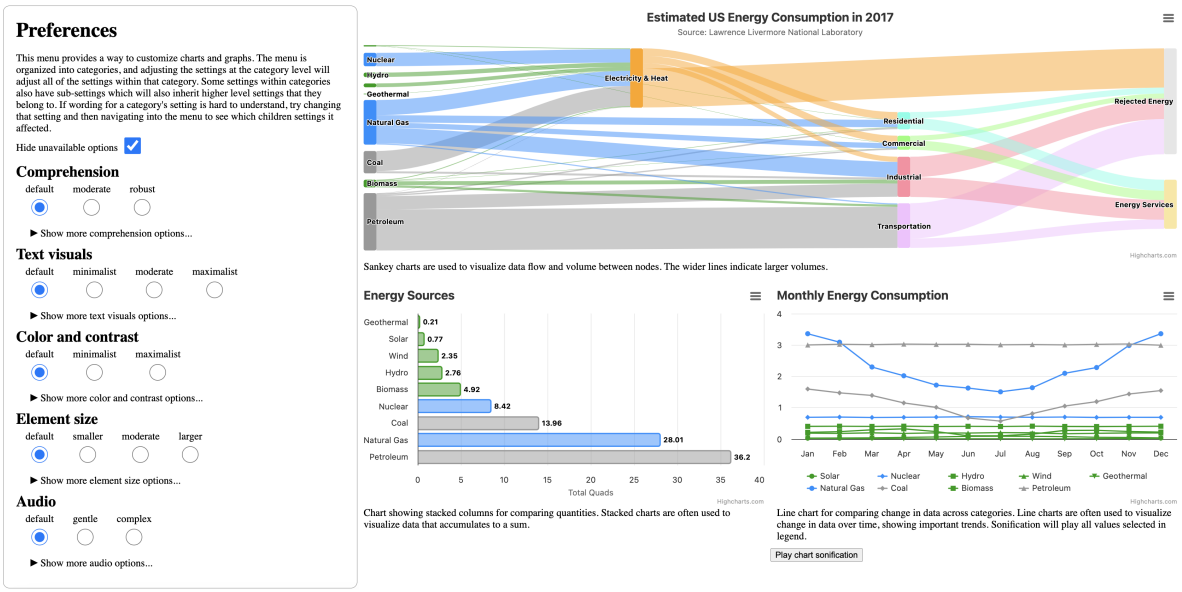


Figure 8.2: Our dashboard design on US Energy Consumption in 2017 with a sankey, bar chart, line chart, and preferences menu on the left.

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 [76, 296]. 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 [162].

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 8.2), 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 [11] and instead focus on higher-level categories 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.

The study subset is documented in full in Table 8.1, and the complete working set of 195 option categories and 774 options is reproduced in Appendix A. Both are also viewable in our live, interactive demo online. Across both sets, every category that offers choices presents them as a closed, binned list of no more than 7 options that begins with "default." The choices span ordinal intensity bins (such as "small" to "large"), nominal alternatives (such as font families), toggles (such as "enabled" and "disabled"), numeric bins (such as font weights of 100, 400, and 700, or bar spacing from 0 to 50 percent), and "custom" escape hatches in categories where a free value, such as a custom color or a custom keypress, would be required.

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 ma-

Table 8.1: The study subset of our option taxonomy: 33 option categories and 137 total options, exactly as enabled in the prototype's preferences menu. Indentation reflects the menu hierarchy; setting a parent category cascades to its children until a child is set directly. N is the number of choices in each category.

Option category	Choices	N
Comprehension	default, moderate, robust	3
Alt text visual appearance	default, show high level, show all	3
Description verbosity	default, disable, minimal, verbose	4
Chart	default, disable, minimal, verbose	4
Text visuals	default, minimalist, moderate, maximalist	4
Font Size	default, small, medium, large	4
Title	default, small, small+, medium, medium+, large	6
Subtitle	default, small, small+, medium, medium+, large	6
Series Labels	default, small, small+, medium, medium+, large	6
Tooltip	default, small, small+, medium, medium+, large	6
Legend Labels	default, small, small+, medium, medium+, large	6
Axis Labels	default, small, small+, medium, medium+, large	6
Font Weight	default, 100, 400, 700	4
Title	default, 100, 400, 700	4
Subtitle	default, 100, 400, 700	4
Series Labels	default, 100, 400, 700	4
Tooltip	default, 100, 400, 700	4
Legend Labels	default, 100, 400, 700	4
Axis Labels	default, 100, 400, 700	4
Color and contrast	default, minimalist, maximalist	3
Text color	default, black, grey, white, custom	5
Mark color	default, limited dark, limited white	3
Distinguish without color	default, enabled	2
Fill patterns	default, low contrast, high contrast, custom	4
Element size	default, smaller, moderate, larger	4
Lines	default, light, moderate, thick	4
Audio	default, gentle, complex	3
Sonification duration	default, slow, moderate, fast	4
Sonification order	default, sequential, simultaneous	3
Sonification volume	default, low, medium, high	4
Sonification pitch range	default, small, moderate, full	4
Pitch min.	default, c4, c3, c2	4
Pitch max.	default, d5, d6, d7	4
33 option categories		**137**

nipulation 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 8.2). 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.

8.5.4 What the Prototype Actually Is

Concretely, our prototype is a single-page web application: three Highcharts charts arranged as a dashboard (a sankey diagram of estimated 2017 US energy flows, a bar chart of totals by energy source, and a line chart of monthly consumption with on-demand sonification), with the preferences menu rendered in a column beside them. The charts are built directly on Highcharts and its accessibility, sonification, and pattern-fill modules; the menu itself is dependency-free HTML, CSS, and JavaScript. The entire option dataset ships inside the prototype as a plain-text nested list, which is parsed at load time into a tree of option categories, and the menu interface is generated from that tree: each category becomes a disclosure group containing a radio group of its choices. All 195 categories are parsed and rendered, the 33 study categories are flagged as available, and the remainder render as disabled controls behind a "Hide unavailable options" toggle that is on by default. Participants therefore interacted with the 33 enabled categories, while the structure of the larger set remained present in the page.

The hierarchical inheritance described above is implemented as a cascade. When a choice is made at a category level, each descendant first looks for a choice of the same name, then consults an explicit override mapping (for example, "maximalist" under "Color and contrast" maps to "enabled" for "Distinguish without color"), and otherwise falls back to the choice at the matching position in its own list, which is how "gentle" under "Audio" resolves to "sequential" for "Sonification order." Any category a user sets directly is marked as an override and stops inheriting from its parents. The current value of every category is held in a single state object, and choosing "default" restores values captured from the charts when the page loaded.

Each enabled category is bound to a handler that applies the selection to the live charts in one of three ways. Most handlers go through Highcharts' update interface: text size, weight, and color per text element, data line thickness, and the sonification module's duration, order, volume, and pitch range. A second group manipulates the document and styles directly: the "limited white" and "limited dark" mark color modes are implemented as greyscale and inverted greyscale filters applied to the chart graphics, and the fill pattern options swap the sankey's marks to textured pattern fills. A third group exposes accessibility structure that is normally non-visual: the "Alt text visual appearance" options reveal the screen reader region on screen and can overlay each mark's screen reader label as a visible label on the chart itself.

We are also explicit about what is *not* implemented. "Motion" and "Interactivity" appear

in the menu but are not wired to the charts in this prototype, and the top level of each enabled category acts only by cascading to its children rather than applying any direct change of its own. The menu-hidden phase of our study used this same page, with the menu suppressed through a URL parameter, so both study conditions ran against an identical dashboard.

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

Table 8.2: 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

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 [114].

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 [222].

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 8.2), 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, 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 8.2), 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 [4]. 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 [119].

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 [175] and aspect ratios [42] 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) is an important design consideration. 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" [180].

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 [297]. 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 [86]. 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 8.1).

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 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 [58], we observed an *effort-to-outcome* ratio among our participants, in line with previous results from existing work [323]. 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.

8.9 Postface

Since publication, Highsoft has now run more studies and matured their approach into a proper contribution to their production environment. Our prototype is now directly responsible for work that is integrated into the *Highcharts* ecosystem. They dramatically trimmed the customization options down, after settling on the best balance between option-overload and highest-impact accessibility choices. Given that *Highcharts* is a library with downloads on the magnitude of millions per month, we believe that this project in particular has the greatest potential to impact the lives of people with disabilities who interact with visualizations.

And beyond this work, personalization and the broader dream of *softerware* might yet be realized in the coming years. We look forward to work that begins to blur the lines between designer/developer/owner and user, and work that reduces the effort required to make the interfaces we use feel as if they are our own, made for us. While we are wary of the environmental and ethical impacts that large language models and modern agentic technologies have had and continue to have on our world, the flexibility and power that they afford someone to scaffold and manipulate software using code may signal a new avenue to enable *softerware*.

But even with stochastic models and agent-based design taking over the imagination of HCI researchers and innovators at this present moment, the prototype we built for *softerware* was designed with *deterministic agency* in mind: giving users discrete controls with predictable cause and effect. Additionally, *softerware*'s greatest challenges aren't actually at the individual interface level. Manipulating an interface isn't the hard problem. The hard problems are persistence, reducing individual labor, socializing labor, and preservation of privacy all at scale. In the conclusion of this thesis, we further unpack why the *substrate* has been the key space of intervention that we implore fellow technologists to investigate.

Part VI

Conclusion

Chapter 9
Findings

The five projects in this thesis were presented as five chapters, each with its own results and its own conclusion. Read separately, they contribute a heuristic resource for evaluation, a structural toolkit for navigation, an authoring environment for non-visual design, an interaction technique and device for blind analysis, and a system and study of end-user personalization. This chapter states what they show *together*. Chapter 2 closed on a lopsided field: a breadth of empirical work and exemplary prototypes on one side, and a near absence of practitioner tools in production environments and practitioner action on the other, with the hardest remaining challenges concentrated in navigation, interaction, and personalization. The findings below are what this thesis learned by working inside that gap.

Each finding is a synthesis across chapters, and each is traceable to results reported in the body of this thesis. We have written them for two audiences at once: practitioners who build, evaluate, and maintain data experiences, and researchers who study how that work happens. Where a promising idea outran the evidence, it is not here; it is in the next chapter, with the rest of the discussion and future work.

9.1 Access Barriers Are Produced for Practitioners, Both With and Without Disabilities

The social model holds that disability is not located in a body but produced in the interaction between a person and a human-made system [190]. This thesis began by applying that lens to end users: visualizations, like curbs, produce barriers. The cross-cutting finding of this thesis is that the same lens applies, with the same force, to the practitioners doing accessibility work, and that the materials of that work are among the systems producing the barriers.

Every project surfaced a version of this. In *Chartability*, the guideline landscape itself was a barrier: standards and research were scattered, often high-level, and difficult to apply to complex interactive data interfaces, and practitioners had little way to judge the quality of what they gathered [153, 258]. Guidelines written to remove barriers for end users limited what practitioners could reasonably understand and act on, and an expert visualization practitioner in our study admitted he did not know where to start. In *Skeleton*'s co-design foundations, *Data Navigator* made better end-user experiences possible, but because its structures were entirely non-visual and code-bound, it produced severe iteration barriers for sighted practitioners: they could not perceive what they were building, and prototypes were misinterpreted or abandoned. And in *Cross-perception*, the screen reader, the most important enabling technology for blind computer use, also limited what blind practitioners could reasonably do analytically: its serial, discrete traversal kept participants in an access-oriented mode that structurally separated interaction from information, with measurable costs in time, completion, and the number of questions they could ask of their data [256].

The operationalized version of this finding: when accessibility work stalls, look for barriers

in the practitioner's tools and materials before attributing the failure to motivation or knowledge. None of the practitioners in these studies lacked care. What they lacked, in each case, was a working relationship with materials that had been shaped without them in mind, and this is true whether the practitioner is sighted, blind, disabled, or not. The same observational stance the field has learned to take toward end users belongs in the study of the workers, too.

9.2 Tools Can Reduce the Work Required to Make Things Accessible

Chapter 2 argued that guidance cannot substitute for means. The constructive half of that argument is that supplying means measurably changes what practitioners can do, and three chapters provide the evidence.

Chartability reduced the perceived difficulty of evaluation and increased the confidence of practitioners new to accessibility, whose audit results were reasonably comparable to the authors' own; beyond the study, it has been applied by policy and governmental organizations and more than 100 companies, across toolchains from Tableau and Excel to D3 and Figma. *Data Navigator* gave previously inaccessible rendering formats a navigable structure: a static raster image gained full multi-input navigation with no automatically detectable conformance failures, and a deterministic ingestion pipeline recreated, and then improved on, an existing toolkit's navigation scheme for rendering modes that previously had none. *Skeleton*'s scaffolding collapsed the spatial authoring of a navigation structure to one or two minutes of refinement, and within single sessions every participant iterated substantively on designs that, in a code-only workflow, our co-design collaborators had struggled to iterate on at all.

None of this is evidence that tools solve access; the boundaries of that claim get their own finding below. It is evidence of something narrower and more useful: the labor of access work is sensitive to tooling, and reductions in that labor are achievable, measurable, and repeatable across very different kinds of tools, from a workbook of heuristics to a rendering-independent toolkit to an authoring interface.

9.3 Making the Invisible Structurally Visible Changes Practice

The strongest through-line in this thesis is what happened when an invisible structure underlying practitioners' work was given a perceivable, inspectable form. Three chapters did this to three different structures. *Chartability* made the evaluative space visible: a scattered landscape of standards and research became a single workbook a practitioner could hold in view, and participants described it as helping them focus, stay consistent between sessions, and aim above compliance. *Skeleton* made navigation structure visible: nodes, edges, and announcements rendered over the practitioner's own chart, and its study found that participants who began by asking compliance questions (does this pass? what does the guideline say?) shifted to asking design questions (is this good? is this too many steps? what is this *for*?). *Softerware* made the dimensions of prefer-

ence visible: an explicit menu of option categories gave end users something concrete to react to and gave practitioners a representation through which access friction stopped being abstract.

The mechanism is one visualization researchers already trust: reflective practice runs on externalized representations that a designer can perceive and respond to [251]. What this thesis adds is evidence that the loop works on structures the field had left invisible, and that restoring the loop moves practitioners from verification toward judgment. As a guideline, this is directly actionable for toolmakers: when practitioners treat some aspect of accessibility as a checkbox, ask what representation of that aspect they can actually perceive, because a structure no one can see is a structure no one can design.

9.4 Friction From a Tool Can Be Productive

Perhaps in tension with the previous finding that tooling can reduce work, tooling may also *increase* some kinds of work in exchange for quality. The assumption for tooling, in a classic or naïve sense, is that tools exist to make work easier by reducing the labor and friction of certain tasks. But ease of work is only a goal if a baseline exists. If some work isn't being done or new design possibilities open up, then *more*, not less, work might be the goal of an idealized tool. Friction can be useful, even productive, and sometimes necessary. While outside of the scope of this thesis, we would urge fellow researchers to explore the tensions that arise in design when removing friction is the primary, if only, goal of a tool (as it appears to be the case with modern large language models and agentic tooling).

But it *is* a reasonable assumption about practitioner tooling that its job is to make work smoother and easier. The evidence in this thesis complicates that assumption: in several places, the value of a tool came precisely from the resistance it introduced. *Chartability* forced engagement with conflicts between guidelines (such as redundant encodings that help some users while overwhelming others) rather than smoothing those conflicts away; audits remained slow, and participants reported that the slowness helped them focus and prioritize. Evaluating a visualization for accessibility actually *adds* time to the overall time it takes to make a visualization. *Chartability* made it clear that if work isn't being done at all in the first place, introducing new heuristics to a practitioner's workflow ultimately generates friction that wasn't there before. We view this as necessary and good.

Data Navigator arguably introduced so much additional work and friction that it necessitated additional tools to make it easier to work with (in the *Inspector*, *Dimensions API*, and ultimately *Skeleton* of Chapter 6). It became apparent that giving practitioners a nearly-infinite new space of design without clear scoping and bounds for that design space was burdensome and resulted in many additional barriers and design iterations. *Data Navigator* could create *any* discrete navigation experience from relationships in an information space. This is fantastic. But it appears that *scope* in design is, in its own way, a source of friction *and* a relief on the work required of that design: it limits and holds back what can be done. But this scoping friction in turn reduces the labor of exploration required. Infinite, frictionless exploration of a design space appears to have put friction elsewhere in the work: during deliberation, handoffs, testing, and ultimately in producing scope.

Skeleton's sessions opened on an intentionally problematic default structure, and its testing

stage made designs that looked adequate feel inadequate in traversal; participants responded to that resistance not by abandoning the work but by iterating, restructuring dimensions, and in one case restarting from the wizard. The tool introduced *more*, not fewer design iterations. Interestingly, *Skeleton* was intended to reduce friction and make work easier, yet by surfacing and making perceivable and manipulable a space of design elements, practitioners didn't just neatly solve problems in less time or solve more problems, in some cases they problematized their own work and returned to the figurative (and literal) drawing board.

Softerware also produced friction at the level of professional identity: it challenges the traditional separation between the people who design a system and the people who use it, and our practitioners immediately raised the ethical concern that personalization could excuse poor defaults. That concern surfacing early, before any deployment, is the friction doing its work. Practitioners suddenly had to worry about ways that users could self-design into misleading, harmful, or newly inaccessible designs on their own. And yet, practitioners then had to wrestle with the question, "if we give people power over their interfaces, isn't it then their responsibility in the end for what they do with that power?" Our prototype and study didn't neatly resolve these frictions or close out this conversation, but rather these frictions simply served to open up new directions and outline problems presently unsolved.

We would argue that the *cross-feelter* device from *Cross-perception* was a "classic" tool in this sense, unlike the rest: it reduced friction *and* reduced work, which led to more productivity. It is good to hold this project in tension with all of the others in this thesis, which introduced new, productive frictions into accessibility work in some way or another and may or may not have made work easier in some ways in the process. *Cross-perception*, by comparison, primarily served to open up a space of design while we demonstrated one design within that space through our hardware device.

Chapter 2 argued that tools direct the work and the workers who use them. Friction is one of the ways they do it. The finding here is that a tool which removes all resistance from accessibility work can also remove the engagement that the work requires. This is, in some cases, not an ideal outcome. Engagement with particular tensions might be the *best* thing that a particular tool can accomplish. This means that toolmakers can place friction deliberately: at conflicts that deserve a decision, at defaults that deserve suspicion, at moments where compliance would otherwise substitute for judgment, and in places where designerly, considerate work isn't being done but *should* be.

9.5 Tool-use and Tool-making Are Generative

Across the studies in this thesis, tools did not merely execute their users' existing intentions; they expanded what practitioners and users could imagine. In *Skeleton*'s study, five of eight participants reconsidered the design of the visualizations they had brought, not just the navigation built over them: some simplified, some questioned their chart type, and one concluded that the right non-visual design was no navigation at all, a genuine design decision rather than an omission. In *Cross-perception*, formalizing the technique and putting a working device in participants' hands opened an input design space: blind data practitioners proposed pre-rendered tactile interaction cubes, feelter-driven array navigation, sonification scrubbing, and multi-axis spatial interaction,

extensions we present as design starting points. In *Softerware*, the visible option space provoked interface ideas we had not designed for, including in-context, voice-driven adjustment while navigating, and led practitioners to imagine authoring their own systems through a softerware interface. Even *Data Navigator*, the lowest-level contribution here, showed this property (and perhaps the strongest): its vocabulary of nodes, edges, and rules gave co-design sessions a shared language in which novel navigable prototypes could be drafted in days.

The generative freedom afforded by tools isn't just *expressiveness*, as defined by existing work on visualization grammars [241, 313], which tends to enable fluency within the bounds of a structured language, but also generation outside of any existing bounds as well [16].

For researchers, this finding doubles as a methodological one, and it is the stance Chapter 2 named: studying practitioners through their tools works because a tool placed in real work is simultaneously an intervention and an instrument. The generative responses it provokes, the reconsiderations, extensions, and refusals, are data about why the work is hard and where it could go next.

And tool-making isn't just generative because tool-users appropriate, re-use, and innovate once they have a new tool, but viewing gaps and problems that practitioners face is also generative for new research directions. This was the cornerstone motivation behind our *action research* approach in Chapter 6: that collaboration with practitioners as an embedded designer, engineer, and researcher in order to join them as they solve problems provides a rich space where novel intellectual gaps can be made both legible and actionable. Tool-making alongside tool-makers and tool-users without having an *a priori* intellectual agenda in mind leaves room for developing new conjectures, evaluating multiple collaborations in aggregate, and ultimately generating research agendas.

9.6 The Hardest Barriers Required Intervention at the Substrate

Chapter 2 characterized navigation, interaction, and personalization as a missing structure, a missing technique, and a missing infrastructure. Having now worked in all three domains, we can state what they share: in each, the barrier sat one level below where practitioners are ordinarily able to act, in the substrate their work is built on, and in each, outcomes changed when the intervention went to that level rather than adding features above it. This finding is held heavily in tension with existing research that tends towards higher levels of software and interaction abstraction, not lower. A massive wave of new research has emerged that essentially asks, "what new layers or wrappers can we create around existing interfaces and user experience problems, using large language models and agents?" While our thesis doesn't engage the effectiveness or intellectual value of those projects, instead we hold this thesis in fundamental contrast to their approach: that some of our most difficult gaps are best engaged below our present levels of software abstraction, not above.

The evidence is consistent across the three parts of this thesis. No amount of feature work on top of canvas-rendered pixels can recover semantics that the rendering substrate never carried, and forcing complex, non-hierarchical data relationships into HTML's hierarchical substrate

limited what was possible to build. This is precisely what a large and passionate resistance to "overlay" technologies engages within both the practical and academic accessibility communities [72, 109, 120, 184]. There may simply be a limit to how much good data, if at all, can be generated from bad or missing data [24, 220].

Data Navigator changed outcomes by supplying a structural substrate of its own, a graph independent of any renderer. This requires underlying data that is supplied with sufficient quality, what Doug Schepers calls an "underlay" rather than an overlay [246].

No software feature on existing consumer hardware approximates persistent physical state and simultaneous perception of input and output, the properties analytical interaction depends on [173]; the *cross-feelter* changed the hardware substrate, and the character of blind analysis changed with it, from accessing data toward interrogating it, at nearly three times the query rate.

And per-chart personalization features fail the effort-to-outcome test that end users themselves applied; what every participant and practitioner in the *Softerware* study named instead were system-class needs [86]: persistence, profile sharing, interoperability, and standards beneath any individual chart.

This is why the substrate matters as a space of intervention: in all three domains, the ceiling on what a motivated, well-informed practitioner could make accessible was set by material beneath their reach, so guidance, effort, and even exemplary prototypes hit that ceiling without moving it. The work in this thesis moved outcomes exactly where it changed the material itself, and the same evidence shows the stages of this work leaning on one another: structure without authoring support went unused, authoring raised evaluative questions practitioners could not answer alone, and personalization without accessible defaults and persistent infrastructure shifts burden onto the people it is meant to serve. Researchers and toolmakers deciding where to spend effort in these three domains have, in this thesis, a consistent answer about where the leverage was found.

9.7 What Tool-making Did Not Do

The findings above would be misread without their boundaries. Three are most apparent in this thesis, which we unpack below.

9.7.1 State of the Art does not Necessitate Adoption

First, a tool's existence, even if it is the "best" system by some argument or measure, did not cause its adoption. This should seem obvious, and yet technical research appears to operate with an unspoken assumption that *something* practical should happen at scale after a clever or novel prototype is produced. Research ideas are brought into the world and largely abandoned almost immediately [139]. *Data Navigator* was (and perhaps still is) the most flexible and expressive system made for data navigation, and yet translating designs into it remained too complex to simply hand off, which is what motivated our industry collaborations and ultimately *Skeleton*; *Softerware*'s participants made clear that personalization that must be redone per chart is not worth their effort; and the proportion of the web's most popular pages with critical accessibility failures is essentially unchanged across the six years spanned by this research [308]. Chapter 2

observed that a practitioner cannot simply install a paper; this thesis adds that practitioners will not, and often cannot, adopt even a working tool absent authoring support, incentives, infrastructure, and policy around it. Existence is the beginning of an argument for adoption, not the conclusion or even motivation for one.

9.7.2 Tooling does not *Solve* Access Friction

Second, tools did not solve access friction, and our evidence suggests they may not be able to. Access friction may simply be inevitable, as others already demonstrate and argue [117, 138, 200]. In the *Softerware* study, no two participants chose the same configuration, participants with the same diagnosed condition needed opposite design treatments, and one participant wanted strong defaults rather than any customization at all. Personalization *relocates* friction and gives people agency within it; but it does not engineer the friction away. Strong accessible defaults, despite being a source of friction in the first place, remain a prerequisite of personalization.

9.7.3 Authorship by People with Disabilities Remains Unresolved

Third, the tools in this thesis did not redistribute authorship. They mostly serve practitioners building *for* disabled users: *Skeleton*'s study evaluated sighted practitioners' engagement, not the experiences their structures produce for blind users, and while *Cross-perception* and *Softerware* begin to shift who holds the interaction, people with disabilities remain largely excluded as authors, analysts, and people with power over data interfaces. This marks what should be expected from tooling interventions and what should be expected from policy, practice, and the sharing of power [275]. Some existing work has used tooling as a space to explore what decentering visual authoring might look like [334], but this largely remains an area with significant gaps in practice. The next chapter takes up exactly these questions, of what tools are downstream from, who is responsible for repair, and what this work asks of the field beyond its tools. We view these gaps as generative spaces for future work.

Chapter 10
Discussion & Future Work

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

In the introduction of this dissertation, we 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 our 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. Our research 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 [120] and subsequent lawsuits and grass-roots resistance.

For our 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 our 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, Frank, 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 [275]. Without enforcement and threat of litigation, very little accessibility work has been accomplished in the past by technology companies alone.

Featured in these chapters (in their pre and postfaces) is the policy and outreach work involved in seeing that work like *Chartability* and *Data Navigator* were 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.

10.2 "Applied" accessibility work and *low research*

I came into an academic research environment already as an engineer. I had questions to big problems, especially about how tools might shape human behavior and how we can use that towards ethically good outcomes. But *Chartability* was a project I made first for myself, to engage the problems I, personally, was facing.

And *Data Navigator* was motivated by my existing experience making visualizations navigable. I knew we had to build better tools, because continually trying to make a hierarchical substrate like HTML work for complex, non-hierarchical relationships simply limited what was possible and easy to do.

These two projects then led to many opportunities to test them in applied contexts. Our CZI grant was motivated towards the application of both, Adobe was interested in *Data Navigator*, and the University of Wisconsin's GIS folks wanted to figure out how to make a complex map navigable, *Data Navigator* or not. Yet, as a researcher I wasn't incentivized to pursue these projects. In fact, they all proved to have no immediately apparent novelty. It was a risk to spend my time in these spaces.

Applied work is structurally and socially devalued in academic research environments, seen often as a lower form of research and design compared to more "pure" forms of knowledge production. (We conjecture that in part this is due to the scope of knowledge that practitioners produce: they work to take broad knowledge and apply it to specific, small problem spaces. Foundational research tends to assume that broad knowledge does not then bubble up from this work, that it is a one-way flow.)

In my first year as a PhD student I was given the advice by a senior researcher not to hold my experience as a practitioner too highly because, "practitioners often don't know what they are doing." But I did, of course, to some degree. Yet, this attitude is pervasive in some spaces and is a problem because it limits what kinds of knowledge we care to attend to.

Initially, I felt that with *Chartability* I had lucked out. But after four more collaborations which resulted in funding and two research publications, we can now argue that this method has been reliable. In *Softerware* and *Skeleton*, pursuing practical problems in applied environments *did* serve as a fruitful source of generalizable knowledge and advancements in the state of the art. But these projects, in addition to *Chartability*, were all motivated by immediate, non-novel problems: people knew their charts were inaccessible and wanted some design guidance or engineering solutions.

So perhaps "low research" works.

Yet, some research communities, especially *ASSETS*, have yet to reckon with "applied" work like this. If you trace the lineage of *action research* from Hayes [121] (which is how we frame our applied work in *Skeleton*), it leads into fantastic areas of participatory and community-based work, often with or by marginalized people and people with disabilities [18, 138, 178, 186, 275, 279, 280, 290, 316]. We would even argue that this lineage of work, which prioritizes and centers the lived experience of a person, is also why we have seen so many successful auto-biographical and auto-innovation styles of research as well recently [22, 96, 100, 133, 253, 314, 315].

And yet, action research largely does not touch on how practitioners (particularly those without disabilities) do applied work in accessibility [153, 258]. We believe that significant gaps exist in the space between traditional "high" knowledge production and applied, "lowly" spaces. Researchers tend to pursue novelty and the lifecycle of research production is highly risk averse. And yet, relatively little work has been done within the accessibility research community to intellectually address the actual social and cultural risks involved in producing decades of useless, lost, and forgotten prototypes and design guidelines [139, 145]. We would conjecture that research-producing academics are overly concerned with academic readership relative to the myriad of ways that their knowledge production actually is used by the rest of the world [201]. We would challenge *ASSETS* and the AT/accessibility communities of *CHI* in particular to take seriously the lack of applied (industry, government, and non-academic) work it produces as a community.

10.3 Who is responsible for repair?

Lastly, we want to revisit one of this thesis' opening points, where we argue that the *tool-makers* are first responsible for repair. This is true. However, the most pressing issue we 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 [104] or de-centralized agency [41, 157, 202], 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 in-

terfaces into a better state. But back when we 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 six years later, that proportion is unchanged [308].

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.

We 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 [234], and yet localities treat sidewalk maintenance in different ways: some, like where I, Frank, 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 [321]. 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 our 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 we look to questions of *democratic* and *radical* access to accessibility repair. The barriers we hope to tackle in the future are political and infrastructural. Perhaps tool-making will participate in this work, but it seems clear now from our 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. We 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 [21] separate from the web? Above it, looking down into it, like shared annotation tools but capable of sharing the manipulation of websites [233]? A space with ambient co-repair, modeled after projects that bring people together [254] or that allow community "fixing" of misinformation [146]? Perhaps feminist thought on the ethics of care can help us [124, 183]? 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 we 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.

Chapter 11
Biographical Sketch

Frank Elavsky was born in Seattle, Washington and grew up in Anacortes, to the north on Fidalgo Island. He was raised in poverty by a single mother on SSDI who lives with multiple disabilities; his father is multiply disabled and homeless. Frank himself has chronic pain and disability, as well as living with diagnosed and self-diagnosed neurodivergence. In his early years in school, he had alternative classes and behavioral interventions. In middle school, he began working on a fantasy world and tabletop game called *Braven*. He was not a notable student in high school and did not receive honors. He was in and out of surgeries and the emergency room, missing over 180 days of school combined across his junior and senior years. At the time, he loved traditional art and was accepted into Cornish School of the Arts in Seattle. He could not afford to attend.

Upon turning 18 but before graduating high school, he was kicked out of his house by his mother, who had just sold his car for opiates. After several surgeries by his 19th birthday, his health significantly improved. He then worked several odd jobs (furniture mover, barista, carpenter, painter) until getting fired from a pizza shop at age 21. He crashed out and applied to a nearby liberal arts college, Trinity Lutheran, in Everett, WA. He was awarded a substantial scholarship. 5 years later he graduated Magna Cum Laude with two degrees, a B.S. in Computer Information Systems, and a B.A. in Theology and Philosophy.

While attending, he was committed to serving his peers as Student Body President and also did substantial work in his spare time as a community organizer, fighting for student, housing, and carceral justice at Trinity and in the Everett area. He married Shelby while attending Trinity, who was a close friend from home in Anacortes. Trinity closed as a college and became a foundation the year he graduated (2016), because students, staff, and faculty organized against corruption. (On rare occasions, Frank has been known to spill his insider-exclusive tea on all of this over drinks, with close friends.)

He then worked in industry, specifically in data visualization. His visualization work won several awards, even appearing (uncredited) in the 2017 Nobel Lecture on Physics by Barry Barish (see Fig 9 in [15]). Despite this, he was always terrified by impostor syndrome.

At 31, he left a well-paying, comfortable career with excellent health insurance and started his PhD at Carnegie Mellon University. During his PhD, his work has been recognized for its significant societal contributions, shaping data visualization and accessibility work in: 15+ government and policy orgs internationally (World Health Organization, European Commission), 80+ businesses and corporations (including three of the Fortune 5), 20+ news and journalism groups (BBC, NYT), 50+ community organizations and non-profits (Special Olympics, Data Viz Society), and 24+ higher-ed classrooms.

Frank credits his success and happiness to many second chances, unearned opportunities, and genuine kindness given to him in his life. He did not succeed "despite adversity" (as is a common framing with disability and poverty) but simply because many people supported him in his life and still do. He continues to play and work on his "paracosm" [6, 229] *Braven* and it is his longest-running, sustained passion in life.

Appendix A
The *Softerware* Option Taxonomy

This appendix reproduces the full working set of data visualization personalization options assembled for the *softerware* project (Chapter 8): 195 option categories containing 774 total options. The set is reproduced verbatim from the option dataset published in our open source repository, in the order the categories appear in the prototype's preferences menu. Indentation reflects the menu hierarchy. "Input mapping" is a grouping category that holds remappable input bindings and offers no direct choices of its own. The 33 categories enabled for our study are listed in Table 8.1.

Table A.1: The full *softerware* option taxonomy: 195 option categories and 774 total options.

Option category	Choices	N
Comprehension	default, moderate, robust	3
Alt text visual appearance	default, show high level, show all	3
Description verbosity	default, disable, minimal, verbose	4
Chart	default, disable, minimal, verbose	4
Region alt text	default, disable, minimal, verbose	4
Mark alt text	default, disable, minimal, verbose	4
Interactions	default, disable, minimal, verbose	4
Animations	default, disable, minimal, verbose	4
Sonification	default, disable, minimal, verbose	4
Explanation verbosity	default, disable, minimal, verbose	4
Chart	default, disable, minimal, verbose	4
Data	default, disable, minimal, verbose	4
Interactions	default, disable, minimal, verbose	4
Animations	default, disable, minimal, verbose	4
Sonification	default, disable, minimal, verbose	4
Cues and labels	default, pattern, show	3
Annotations	default, pattern, show	3
Mark Labels	default, pattern, show	3
Series Labels	default, pattern, show	3
Legend Title	default, pattern, show	3
Legend Label	default, pattern, show	3
Axis Title	default, pattern, show	3
Axis Labels	default, pattern, show	3
Gridlines	default, pattern, show	3
Interactions	default, pattern, show	3
Text visuals	default, minimalist, moderate, maximalist	4
Font Size	default, small, medium, large	4
Title	default, small, small+, medium, medium+, large	6
Subtitle	default, small, small+, medium, medium+, large	6
Mark Labels	default, small, small+, medium, medium+, large	6
Series Labels	default, small, small+, medium, medium+, large	6
Tooltip	default, small, small+, medium, medium+, large	6
Legend Title	default, small, small+, medium, medium+, large	6

(continued on next page)

(continued from previous page)

Option category	Choices	N
Legend Labels	default, small, small+, medium, medium+, large	6
Axis Titles	default, small, small+, medium, medium+, large	6
Axis Labels	default, small, small+, medium, medium+, large	6
Caption	default, small, small+, medium, medium+, large	6
Annotations	default, small, small+, medium, medium+, large	6
Misc.	default, small, small+, medium, medium+, large	6
Font Weight	default, 100, 400, 700	4
Title	default, 100, 400, 700	4
Subtitle	default, 100, 400, 700	4
Mark Labels	default, 100, 400, 700	4
Series Labels	default, 100, 400, 700	4
Tooltip	default, 100, 400, 700	4
Legend Title	default, 100, 400, 700	4
Legend Labels	default, 100, 400, 700	4
Axis Titles	default, 100, 400, 700	4
Axis Labels	default, 100, 400, 700	4
Caption	default, 100, 400, 700	4
Annotations	default, 100, 400, 700	4
Misc.	default, 100, 400, 700	4
Font Family	default, sans serif, serif, slab serif	4
Title	default, sans serif, serif, slab serif	4
Subtitle	default, sans serif, serif, slab serif	4
Mark Labels	default, sans serif, serif, slab serif	4
Series Labels	default, sans serif, serif, slab serif	4
Legend Title	default, sans serif, serif, slab serif	4
Legend Labels	default, sans serif, serif, slab serif	4
Axis Titles	default, sans serif, serif, slab serif	4
Axis Labels	default, sans serif, serif, slab serif	4
Caption	default, sans serif, serif, slab serif	4
Annotations	default, sans serif, serif, slab serif	4
Misc.	default, sans serif, serif, slab serif	4
Text overflow	default, truncate, ellipses, allow overflow	4
Title	default, truncate, ellipses, allow overflow	4
Subtitle	default, truncate, ellipses, allow overflow	4
Mark Labels	default, truncate, ellipses, allow overflow	4
Series Labels	default, truncate, ellipses, allow overflow	4
Legend Title	default, truncate, ellipses, allow overflow	4
Legend Labels	default, truncate, ellipses, allow overflow	4
Axis Titles	default, truncate, ellipses, allow overflow	4
Axis Labels	default, truncate, ellipses, allow overflow	4
Caption	default, truncate, ellipses, allow overflow	4
Annotations	default, truncate, ellipses, allow overflow	4
Misc.	default, truncate, ellipses, allow overflow	4
Horizontal Spacing	default, tight, moderate, wide	4
Title	default, tight, moderate, wide	4
Subtitle	default, tight, moderate, wide	4
Mark Labels	default, tight, moderate, wide	4
Series Labels	default, tight, moderate, wide	4
Legend Title	default, tight, moderate, wide	4

(continued on next page)

(continued from previous page)

Option category	Choices	N
Legend Labels	default, tight, moderate, wide	4
Axis Titles	default, tight, moderate, wide	4
Axis Labels	default, tight, moderate, wide	4
Caption	default, tight, moderate, wide	4
Annotations	default, tight, moderate, wide	4
Misc.	default, tight, moderate, wide	4
Vertical Spacing	default, tight, moderate, wide	4
Title	default, tight, moderate, wide	4
Subtitle	default, tight, moderate, wide	4
Mark Labels	default, tight, moderate, wide	4
Series Labels	default, tight, moderate, wide	4
Legend Title	default, tight, moderate, wide	4
Legend Labels	default, tight, moderate, wide	4
Axis Titles	default, tight, moderate, wide	4
Axis Labels	default, tight, moderate, wide	4
Caption	default, tight, moderate, wide	4
Annotations	default, tight, moderate, wide	4
Misc.	default, tight, moderate, wide	4
Italics	default, allow, disabled	3
Alignment	default, left, centered, right	4
Title	default, left, centered, right	4
Subtitle	default, left, centered, right	4
Mark Labels	default, left, centered, right	4
Series Labels	default, left, centered, right	4
Legend Title	default, left, centered, right	4
Legend Labels	default, left, centered, right	4
Axis Titles	default, left, centered, right	4
Axis Labels	default, left, centered, right	4
Caption	default, left, centered, right	4
Annotations	default, left, centered, right	4
Misc.	default, left, centered, right	4
Color and contrast	default, minimalist, maximalist	3
Text color	default, black, grey, white, custom	5
Mark color	default, limited dark, limited white	3
Categorical data types	default, limited dark, limited white, custom	4
Ordinal data types	default, limited dark, limited white, custom	4
Quantitative data types	default, limited dark, limited white, custom	4
Non data marks	default, limited dark, limited white, custom	4
Distinguish without color	default, enabled	2
Shapes	default, 1 varied type, different types, custom	4
Fill patterns	default, low contrast, high contrast, custom	4
Line patterns	default, enabled, custom	3
Interaction states	default, patterns, marks, patterns + marks	4
Text contrast	default, low contrast, high contrast	3
Title	default, low contrast, high contrast, custom	4
Subtitle	default, low contrast, high contrast, custom	4
Mark Labels	default, low contrast, high contrast, custom	4
Series Labels	default, low contrast, high contrast, custom	4

(continued on next page)

(continued from previous page)

Option category	Choices	N
Legend Title	default, low contrast, high contrast, custom	4
Legend Labels	default, low contrast, high contrast, custom	4
Axis Titles	default, low contrast, high contrast, custom	4
Axis Labels	default, low contrast, high contrast, custom	4
Caption	default, low contrast, high contrast, custom	4
Annotations	default, low contrast, high contrast, custom	4
Misc.	default, low contrast, high contrast, custom	4
Mark contrast	default, low contrast, high contrast	3
Categorical data types	default, low contrast, high contrast	3
Ordinal data types	default, low contrast, high contrast	3
Quantitative data types	default, low contrast, high contrast	3
Non data marks	default, low contrast, high contrast	3
Mark opacity	default, low, moderate, full	4
Axis line contrast	default, low contrast, high contrast, custom	4
Gridline contrast	default, low contrast, high contrast, custom	4
Mark outline contrast	default, low contrast, high contrast, custom	4
Text outline contrast	default, low contrast, high contrast, custom	4
Interaction state contrast	default, low contrast, high contrast, custom	4
Misc. element contrast	default, low contrast, high contrast, custom	4
Element size	default, smaller, moderate, larger	4
Bar spacing	default, 0%, 10%, 20%, 30%, 40%, 50%	7
Lines	default, light, moderate, thick	4
Data lines	default, thin, light, moderate, thick, heavy	6
Axis lines	default, thin, light, moderate, thick, heavy	6
Gridlines	default, thin, light, moderate, thick, heavy	6
Outlines	default, thin, light, moderate, thick, heavy	6
Mark min. size	default, 0, 3px, 10px	4
Scaled mark min.	default, 0px to 44px	2
Non scaled mark min.	default, 1px to 44px	2
Interaction target	default, minimal, moderate, large	4
Audio	default, gentle, complex	3
Sonification duration	default, slow, moderate, fast	4
Sonification order	default, sequential, simultaneous	3
Sonification volume	default, low, medium, high	4
Sonification equalizer	default, low, medium, high	4
High pitch volume	default, quiet, low, medium, high, loud	6
Middle pitch volume	default, quiet, low, medium, high, loud	6
Low pitch volume	default, quiet, low, medium, high, loud	6
Sonification pitch range	default, small, moderate, full	4
Pitch min.	default, c4, c3, c2	4
Pitch max.	default, d5, d6, d7	4
Instruments	default, all sine, all piano, custom	4
Tempo ticker	default, disabled, enabled	3
Tempo ticker volume	default, low, medium, high	4
Tempo ticker timing	default, 1/2, 1/3, 1/4	4
Motion	default, disabled, slower, faster	4
Interactivity	default, minimal feedback, high feedback	3

(continued on next page)

(continued from previous page)

Option category	Choices	N
Pointer focus	default, disabled, on hover, on click	4
Interaction console	default, show as log, always show	3
Tooltips	default, show on focus, toggle only	3
Element selection	default, no confirmation, use confirmation	3
Filtering	default, no confirmation, use confirmation	3
Input mapping	*(grouping only)*	
Help	default, F1, custom keypress	3
Preferences menu	default, Shift+F1, custom keypress	3
Activate Hover	default, on focus, custom keypress	3
Remove Hover	default, Backspace/on loss of focus, custom keypress	3
Select	default, Spacebar/Enter, custom keypress	3
Exit and skip chart	default, Escape, custom keypress	3
Next element	default, AT right, down, custom keypress	4
Previous element	default, AT left, up, custom keypress	4
Next across element	default, period, custom keypress	3
Previous across element	default, comma, custom keypress	3
Parent element	default, AT up, page up, custom keypress	4
First child element	default, AT down, page down, custom keypress	4
195 option categories		**774**

References

[1] Christopher Ahlberg and Ben Shneiderman. Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI94, page 313–317, New York, NY, April 1994. ACM. doi: 10.1145/191666.191775. URL https://doi.org/10.1145/191666.191775. 7.2, 7.3.1, **G4:**, **G6:**

[2] Sara Ahmed. *Queer Phenomenology: Orientations, Objects, Others*. Duke University Press, Durham, NC, 2006. ISBN 9780822339144. ISBN: 978-0-8223-3914-4. 2.1.2

[3] Sara Ahmed. *What's the Use? On the Uses of Use*. Duke University Press, Durham, NC, 2019. ISBN 9781478006503. ISBN: 978-1478006503. 2.1.2

[4] 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

[5] Frances Aldrich. Talk to the Hand: An Agenda for Further Research on Tactile Graphics. In Gem Stapleton, John Howse, and John Lee, editors, *Diagrammatic Representation and Inference*, Lecture Notes in Computer Science, page 344–346, Berlin, Heidelberg, 2008. Springer, Springer Berlin Heidelberg. ISBN 9783540877301. doi: 10.1007/978-3-540-87730-1_31. URL https://doi.org/10.1007/978-3-540-87730-1_31. 4.3.1

[6] Christine Alexander. Playing the author: Children's creative writing, paracosms and the construction of family magazines. In *Children, Childhood and Cultural Heritage*, pages 85–103. Routledge, 2012. 11

[7] Obead Alhadreti and Pam Mayhew. Rethinking thinking aloud: A comparison of three think-aloud protocols. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–12. ACM, April 2018. doi: 10.1145/3173574.3173618. URL https://doi.org/10.1145/3173574.3173618. 6.6.2, 7.6.6.2

[8] Ofunne Amaka and Amber Thomas. The Naked Truth. https://pudding.cool/2021/03/foundation-names, 2021. Accessed: 2021-08-20. 4.6.3

[9] 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.*, page 111–117. IEEE, November 2005. doi: 10.1109/infvis.2005.1532136. URL https://doi.org/10.1109/infvis.2005.1532136. 3, 7.2

[10] Apple Inc. Audio graphs. https://developer.apple.com/documentation/

accessibility/audio-graphs, 2021. Apple Developer Documentation. Feature introduced at WWDC 2021. Accessed: 2026-06-12. 2.2.1

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

[12] Catherine M. Baker, Lauren R. Milne, Ryan Drapeau, Jeffrey Scofield, Cynthia L. Bennett, and Richard E. Ladner. Tactile graphics with a voice. *ACM Transactions on Accessible Computing*, 8(1):1–22, January 2016. ISSN 1936-7236. doi: 10.1145/2854005. URL https://doi.org/10.1145/2854005. 2.2.1, 4.3.1

[13] Abhijit Balaji, Thuvaarakkesh Ramanathan, and Venkateshwarlu Sonathi. Chart-Text: A Fully Automated Chart Image Descriptor. *arXiv Preprint*, December 2018. doi: 10.48550/ARXIV.1812.10636. Accessed: 2021-09-06. 4.3.1

[14] BANA. Guidelines and Standards for Tactile Graphics. Braille standard, Braille Authority of North America, 2010. Accessed: 2021-09-03. 2.2.1, 4.3.1, 4.3.3

[15] Barry C. Barish. Nobel lecture: Ligo and gravitational waves ii. *Reviews of Modern Physics*, 90(4), December 2018. ISSN 1539-0756. doi: 10.1103/revmodphys.90.040502. URL http://dx.doi.org/10.1103/RevModPhys.90.040502. 11

[16] Michel Beaudouin-Lafon, Susanne Bødker, and Wendy E. Mackay. Generative theories of interaction. *ACM Transactions on Computer-Human Interaction*, 28(6):1–54, November 2021. ISSN 1557-7325. doi: 10.1145/3468505. URL https://doi.org/10.1145/3468505. 2.1.2, 7.12, 9.5

[17] 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. 7.2, 7.3.1

[18] Cynthia L. Bennett, Erin Brady, and Stacy M. Branham. Interdependence as a Frame for Assistive Technology Research and Design. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '18, page 161–173, New York, NY, USA, October 2018. Association for Computing Machinery, ACM. doi: 10.1145/3234695.3236348. URL https://doi.org/10.1145/3234695.3236348. Accessed: 2021-09-03. 1.1.1, 2.3, 4.4.3, 10.2

[19] Cynthia L. Bennett, Burren Peil, and Daniela K. Rosner. Biographical prototypes. In *Proceedings of the 2019 on Designing Interactive Systems Conference*, pages 35–47. ACM, 2019. doi: 10.1145/3322276.3322376. 2.3

[20] 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, page 1–7. ACM, May 2021. doi: 10.1145/3411763.3441321. URL https://doi.org/10.1145/3411763.3441321. 2.1.2

[21] Hakim Bey. *TAZ: The temporary autonomous zone, ontological anarchy, poetic terrorism*.

Autonomedia, 1991. 10.3

[22] 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. 2.2.1, 6.4.1, 10.2

[23] Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, and Tom Yeh. Vizwiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, page 333–342, New York, NY, USA, October 2010. ACM. ISBN 978-1-4503-0271-5. doi: 10.1145/1866029.1866080. URL https://doi.org/10.1145/1866029.1866080. Accessed: 2021-09-06. 4.3.3

[24] Abeba Birhane, Vishnu Boddeti, Sanghyun Han, Sasha Luccioni, and Vinay Prabhu. Into the laion's den: Investigating hate in multimodal datasets. In *Advances in Neural Information Processing Systems 36*, NeurIPS 2023, page 21268–21284. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2023. doi: 10.52202/075280-0930. URL http://dx.doi.org/10.52202/075280-0930. 9.6

[25] 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. 2.2.1, 2.4.1, 5.4.3.1, 5.4.3.2, 5.4.3.3, 6.3.3, 6.4.4.2

[26] Jens Bornschein, Denise Prescher, and Gerhard Weber. Collaborative Creation of Digital Tactile Graphics. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility - ASSETS '15*, ASSETS '15, page 117–126, New York, NY, USA, October 2015. Association for Computing Machinery, ACM Press. doi: 10.1145/2700648.2809869. URL https://doi.org/10.1145/2700648.2809869. 4.3.1, 6.3.3

[27] 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. 1.1.2, 2.2.3, 5.5.1.1, 6.3.1

[28] Denis Boudreau. Unlocking Accessibility for UX/UI Designers. https://www.24a11y.com/2018/unlocking-accessibility-for-ux-ui-designers/, December 2018. Accessed: 2021-09-07. 4.3.4

[29] Geoffrey C. Bowker and Susan Leigh Star. *Sorting Things Out: Classification and Its Consequences*. MIT Press, Cambridge, MA, 1999. ISBN 9780262522953. ISBN: 978-0-262-52295-3. 2.1.2

[30] 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. ISSN 1559-1476. doi: 10.1177/0145482x9008401002. URL https://doi.org/10.1177/0145482x9008401002. 5.3.2.3

[31] 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. 3, 6.6.3

[32] Albert S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. The MIT Press, Cambridge, MA, May 1990. ISBN 9780262269209. doi: 10.7551/mitpress/1486.001.0001. URL https://doi.org/10.7551/mitpress/1486.001.0001. 7.3.2

[33] 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. 2.2.1, 4.3.1, 6.3.2

[34] Nicola J. Broadbent, Larry R. Squire, and Robert E. Clark. Spatial memory, recognition memory, and the hippocampus. *Proceedings of the National Academy of Sciences*, 101 (40):14515–14520, October 2004. ISSN 1091-6490. doi: 10.1073/pnas.0406344101. URL https://doi.org/10.1073/pnas.0406344101. 7.4.1

[35] Craig Brown and Amy Hurst. Viztouch: automatically generated tactile visualizations of coordinate spaces. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*, TEI '12, page 131–138, New York, NY, USA, February 2012. ACM. ISBN 978-1-4503-1174-8. doi: 10.1145/2148131.2148160. URL https://doi.org/10.1145/2148131.2148160. Accessed: 2021-09-03. 4.3.1

[36] A. Buja, J.A. McDonald, J. Michalak, and W. Stuetzle. Interactive data visualization using focusing and linking. In *Proceeding Visualization '91*, pages 156–163,, Los Alamitos, CA, December 2002. IEEE Comput. Soc. Press. doi: 10.1109/visual.1991.175794. URL https://doi.org/10.1109/visual.1991.175794. 7.2, 7.3.1, **G4:**

[37] 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: a meta-analysis. In *Proceedings of the 8th international conference on Multimodal interfaces*, ICMI06, page 108–117. ACM, November 2006. doi: 10.1145/1180995.1181017. URL https://doi.org/10.1145/1180995.1181017. 1.1.1

[38] 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, page 1–15. ACM, May 2021. doi: 10.1145/3411764.3445207. URL https://doi.org/10.1145/3411764.3445207. 4.3.1, 6.3.2, 7.2, 7.3.2

[39] 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

[40] Silvia Canelón and Liz Hare. Revealing Room for Improvement in Accessibility within a Social Media Data Visualization Learning Community. https://silvia.rbind.io/talk/2021-05-04-data-viz-accessibility/, 2021. Accessed: 2021-09-06. 4.3.2

[41] 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. 10.3

[42] C. R. Ceja, C. M. McColeman, C. Xiong, and S. L. Franconeri. Truth or square: Aspect ratio biases recall of position encodings. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1054–1062, February 2021. 8.7.1.2

[43] Alex Chaparro and Maria Chaparro. Applications of Color in Design for Color-Deficient Users. *Ergonomics in Design: The Quarterly of Human Factors Applications*, 25(1): 23–30, November 2016. ISSN 1064-8046. doi: 10.1177/1064804616635382. URL https://doi.org/10.1177/1064804616635382. Accessed: 2021-09-06. 4.3.1

[44] James I. Charlton. *Nothing About Us Without Us: Disability Oppression and Empowerment*. University of California Press, Berkeley, CA, March 1998. ISBN 9780520925441. doi: 10.1525/9780520925441. URL https://doi.org/10.1525/9780520925441. 2.3

[45] Charles Chen, Ruiyi Zhang, Sungchul Kim, Scott Cohen, Tong Yu, Ryan Rossi, and Razvan Bunescu. Neural caption generation over figures. In *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*, UbiComp/ISWC '19 Adjunct, pages 482–485, New York, NY, USA, September 2019. Association for Computing Machinery, ACM. doi: 10.1145/3341162.3345601. Accessed: 2021-09-06. 4.3.1

[46] Charles Chen, Ruiyi Zhang, Eunyee Koh, Sungchul Kim, Scott Cohen, and Ryan Rossi. Figure Captioning with Relation Maps for Reasoning. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, page 1526–1534. IEEE, March 2020. doi: 10.1109/WACV45572.2020.9093592. URL https://doi.org/10.1109/wacv45572.2020.9093592. ISSN: 2642-9381. 4.3.1

[47] Jinho Choi, Sanghun Jung, Deok Gun Park, Jaegul Choo, and Niklas Elmqvist. Visualizing for the Non-Visual: Enabling the Visually Impaired to Use Visualization. *Computer Graphics Forum*, 38(3):249–260, 2019. doi: 10.1111/cgf.13686. Accessed: 2021-09-03. 4.3.1

[48] Ngip Khean Chuan, Ashok Sivaji, and Wan Fatimah Wan Ahmad. Usability Heuristics for Heuristic Evaluation of Gestural Interaction in HCI. In Aaron Marcus, editor, *Design, User Experience, and Usability: Design Discourse*, Lecture Notes in Computer Science, page 138–148, Cham, 2015. Springer International Publishing, Springer International Publishing. ISBN 9783319208862. doi: 10.1007/978-3-319-20886-2_14. URL https://doi.org/10.1007/978-3-319-20886-2_14. 4.3.4

[49] Pramod Chundury. Building multimodal chart accessibility for blind and low-vision individuals in the prompt-driven era. *XRDS: Crossroads, The ACM Magazine for Students*, 32(1):34–37, September 2025. ISSN 1528-4980. doi: 10.1145/3778055. URL http://dx.doi.org/10.1145/3778055. 4.11

[50] 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. ISSN 2160-9306. doi: 10.1109/tvcg.2021.3114829. URL https://doi.org/10.1109/tvcg.2021.3114829. 2.2.1, 4.3.1, 5.3.1.2, 6.3.2, 7.9.3

[51] 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, page 1–5. ACM, October 2024. doi: 10.1145/3663548.3688496. URL https://doi.org/10.1145/3663548.3688496. 2.1.1, 7.9.3

[52] Pramod Chundury, J. Bern Jordan, Yasmin Reyazuddin, Niklas Elmqvist, and Jonathan Lazar. Sound, touch, or the full monty? a comparative study of accessible data exploration systems for blind users. *ACM Transactions on Accessible Computing*, 19(1):1–41, March 2026. ISSN 1936-7236. doi: 10.1145/3798100. URL https://doi.org/10.1145/3798100. 7.3.2, 7.9.3

[53] Andy Cockburn and Bruce McKenzie. Evaluating the effectiveness of spatial memory in 2d and 3d physical and virtual environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI02, page 203–210. ACM, April 2002. doi: 10.1145/503376.503413. URL https://doi.org/10.1145/503376.503413. 7.4.1

[54] RStudio Community. Making Shiny apps accessible for all humans. https://community.rstudio.com/t/making-shiny-apps-accessible-for-all-humans/8458, May 2018. Accessed: 2021-09-06. 4.3.2

[55] Maxime Cordeil, Benjamin Bach, Andrew Cunningham, Bastian Montoya, Ross T. Smith, Bruce H. Thomas, and Tim Dwyer. Embodied axes: Tangible, actuated interaction for 3d augmented reality data spaces. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–12. ACM, April 2020. doi: 10.1145/3313831.3376613. URL https://doi.org/10.1145/3313831.3376613. 7.3.4, 7.5.2

[56] B. Craft and P. Cairns. Beyond guidelines: What can we learn from the visual information seeking mantra? In *Ninth International Conference on Information Visualisation (IV'05)*, page 110–118. IEEE, July 2005. doi: 10.1109/IV.2005.28. URL https://doi.org/10.1109/iv.2005.28. ISSN: 2375-0138. 4.3.4

[57] 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. 2.2.1, 4.3.1, 6.4

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

[59] 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*, ASSETS '21, page 1–6. ACM, October 2021. doi: 10.1145/3441852.3476468. URL https://doi.org/10.1145/3441852.3476468. 2.2.1, 5.5.3.1, 6.4

[60] Sergio Della Sala, Colin Gray, Alan Baddeley, Nadia Allamano, and Lindsey Wilson. Pattern span: a tool for unwelding visuo–spatial memory. *Neuropsychologia*, 37(10): 1189–1199, September 1999. ISSN 0028-3932. doi: 10.1016/s0028-3932(98)00159-6. URL https://doi.org/10.1016/s0028-3932(98)00159-6. 7.4.1

[61] Chris DeMartini. Coin flip game. https://public.tableau.com/views/CoinFlipGame/CoinFlipGame, 2019. Accessed: 2021-08-20. 4.6.3

[62] Chris DeMartini. A Tableau Accessibility Journey - Part IV - Keyboard Accessibility. https://www.datablick.com/blog/2021/8/10/a-tableau-accessibility-journey-part-iv-keyboard-accessibility, 2021. Accessed: 2021-09-03. 4.3.2

[63] Deque. Automated Testing Identifies 57 Percent of Digital Accessibility Issues. Technical report, Deque, March 2021. Accessed: 2021-09-06. 3, 4.2

[64] Deque. Axe-core, August 2021. Accessed: 2021-08-20. 4.7.5

[65] 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

[66] 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

[67] Paul Dourish. The appropriation of interactive technologies: Some lessons from placeless documents. *Computer Supported Cooperative Work (CSCW)*, 12(4):465–490, December 2003. ISSN 1573-7551. doi: 10.1023/a:1026149119426. URL https://doi.org/10.1023/a:1026149119426. 5.6

[68] Sebastian Draxler, Gunnar Stevens, Martin Stein, Alexander Boden, and David Randall. Supporting the social context of technology appropriation: on a synthesis of sharing tools and tool knowledge. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, page 2835–2844. ACM, May 2012. doi: 10.1145/2207676.2208687. URL https://doi.org/10.1145/2207676.2208687. 2.1.2

[69] Johanna Drucker. *Graphesis*. metaLABprojects. Harvard University Press, London, England, July 2014. ISBN 9780674724938. 4.1.1

[70] Anthony Dunne and Fiona Raby. *Speculative everything*. The MIT Press. MIT Press, London, England, December 2013. ISBN 9780262019842. 7.8.4, 7.9

[71] 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. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1010622. URL https://doi.org/10.1371/journal.pcbi.1010622. 5.3.1.3

[72] Niklas Egger, Gottfried Zimmermann, and Christophe Strobbe. *Overlay Tools as a Support for Accessible Websites – Possibilities and Limitations*, page 6–17. Springer International Publishing, 2022. ISBN 9783031086458. doi: 10.1007/978-3-031-08645-8_2. URL http://dx.doi.org/10.1007/978-3-031-08645-8_2. 9.6

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

[74] Frank Elavsky. Chartability, 2021. URL https://chartability.github.io/POUR-CAF/. Accessed: 2021-09-07. 4.5

[75] 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)*, page 14–24. IEEE, November 2025. doi: 10.1109/accessviz68666.2025.00008. URL https://doi.org/10.1109/accessviz68666.2025.00008. 6.3.2, 7.1.1

[76] 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. 5.3.1, 5.3.1.3, 5.5.1.1, 5.6, 6.4.3, 6.8.2, **G3:**, 8.3.1, 8.5.1

[77] Frank Elavsky, Lucas Nadolskis, and Dominik Moritz. Data navigator: An accessibility-centered data navigation toolkit. *IEEE Transactions on Visualization and Computer Graphics*, page 1–11, 2023. ISSN 2160-9306. doi: 10.1109/tvcg.2023.3327393. URL https://doi.org/10.1109/tvcg.2023.3327393. 6.2, 6.3.2, 6.3.3, 6.4, 7.2, 7.3.3, 7.9.2

[78] 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

[79] Christin Engel and Gerhard Weber. Analysis of tactile chart design. In *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '17, pages 197–200. ACM, June 2017. doi: 10.1145/3056540.3064955. URL https://doi.org/10.1145/3056540.3064955. 7.3.2

[80] Pavithra Eswaramoorthy, Frank Elavsky, Tania Allard, and gabalafou. Quansight-Labs/bokeh-a11y-audit: v1.0.0, February 2025. URL https://doi.org/10.5281/zenodo.14923642. 6.4.3

[81] Danyang Fan, Alexa Fay Siu, Wing-Sum Adrienne Law, Raymond Ruihong Zhen, Sile O'Modhrain, and Sean Follmer. Slide-tone and tilt-tone: 1-dof haptic techniques for conveying shape characteristics of graphs to blind users. In *CHI Conference on Human Fac-*

tors in Computing Systems, CHI '22, page 1–19. ACM, April 2022. doi: 10.1145/3491102.3517790. URL https://doi.org/10.1145/3491102.3517790. 7.3.2

[82] 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. ISSN 1936-7236. doi: 10.1145/3557899. URL https://doi.org/10.1145/3557899. 1.1.1, 5.3.1, 5.3.1.2, 5.3.1.3, 6.3.3

[83] 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. 6.3.2

[84] Stephen Few. Visual Business Intelligence - Data Visualization and the Blind. https://www.perceptualedge.com/blog/?p=1756, 2013. [Accessed 05-04-2025]. 1.1.1

[85] Danyel Fisher, Igor Popov, Steven Drucker, and m.c. schraefel. Trust me, i'm partially right: incremental visualization lets analysts explore large datasets faster. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, page 1673–1682. ACM, May 2012. doi: 10.1145/2207676.2208294. URL https://doi.org/10.1145/2207676.2208294. 6.2, 7.2, 7.3.1, **G5:**

[86] Chris Fleizach and Jeffrey 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. ISSN 1542-7749. doi: 10.1145/3704627. URL http://dx.doi.org/10.1145/3704627. 2.2.2, 8.7.3, 9.6

[87] 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: The Journal of the Human Factors and Ergonomics Society*, 39(3):341–351, September 1997. doi: 10.1518/001872097778827151. Accessed: 2021-09-03. 2.2.1, 3, 4.3.1

[88] Camilla Forsell and Jimmy Johansson. An heuristic set for evaluation in information visualization. In *Proceedings of the International Conference on Advanced Visual Interfaces*, AVI '10, page 199–206, New York, NY, USA, May 2010. Association for Computing Machinery, ACM. doi: 10.1145/1842993.1843029. URL https://doi.org/10.1145/1842993.1843029. Accessed: 2022-03-06. 4.3.4

[89] Plotly Community Forum. Are plotly tables accessible? https://community.plotly.com/t/are-plotly-tables-accessible/8263, February 2018. Accessed: 2021-09-06. 4.3.2

[90] Plotly Community Forum. Datatables and Accessibility. https://community.plotly.com/t/solved-datatables-and-accessibility/31085, November 2019. Accessed: 2021-09-06. 4.3.2

[91] 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. 1.1.2

[92] Krzysztof Z. Gajos, Daniel S. Weld, and Jacob O. Wobbrock. Automatically generating personalized user interfaces with SUPPLE. *Artificial Intelligence*, 174(12–13):910–950, August 2010. ISSN 0004-3702. doi: 10.1016/j.artint.2010.05.005. URL https://doi.org/10.1016/j.artint.2010.05.005. 2.4.3

[93] Alberto Gallace and Charles Spence. To what extent do Gestalt grouping principles influence tactile perception? *Psychological bulletin*, 137(4):538–561, 2011. ISSN 0033-2909. doi: 10.1037/a0022335. URL https://doi.org/10.1037/a0022335. 4.3.1

[94] 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. ISSN 1097-024X. 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

[95] Sara Garcia-Navarra, Lucia Solares, and Marta Mendez. Virtual reality and mixed reality in the assessment of spatial memory. *Exploration of Digital Health Technologies*, 3, June 2025. ISSN 2996-9409. doi: 10.37349/edht.2025.101151. URL https://doi.org/10.37349/edht.2025.101151. 7.4.1, **G1:**

[96] Paula Gardner and Jess Rauchberg. *Feminist, Postcolonial, and Crip Approaches to Human-machine Communication Methodology*. Sage Publishing, 2023. 10.2

[97] Elena Gherri and Martin Eimer. Active listening impairs visual perception and selectivity: An erp study of auditory dual-task costs on visual attention. *Journal of Cognitive Neuroscience*, 23(4):832–844, April 2011. ISSN 1530-8898. doi: 10.1162/jocn.2010.21468. URL https://doi.org/10.1162/jocn.2010.21468. 7.3.2, **G2:**

[98] Josefina Isabel Gil Urrutia, Eric Brangier, Véronique Senderowicz, and Laurent Cessat. Beyond "usability and user experience", towards an integrative heuristic inspection: From accessibility to persuasiveness in the ux evaluation. *Advances in Intelligent Systems and Computing*, pages 460–470, June 2017. doi: 10.1007/978-3-319-60492-3_44. URL https://doi.org/10.1007/978-3-319-60492-3_44. 4.3.4

[99] Stéphanie Giraud and Christophe Jouffrais. Empowering low-vision rehabilitation professionals with "do-it-yourself" methods. In *Lecture Notes in Computer Science*, page 61–68. Springer International Publishing, 2016. ISBN 9783319412672. doi: 10.1007/978-3-319-41267-2_9. URL https://doi.org/10.1007/978-3-319-41267-2_9. 5.3.3

[100] Kate S Glazko, Momona Yamagami, Aashaka Desai, Kelly Avery Mack, Venkatesh Potluri, Xuhai Xu, and Jennifer Mankoff. An autoethnographic case study of generative artificial intelligence's utility for accessibility. In *The 25th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '23, page 1–8. ACM, October 2023. doi: 10.1145/3597638.3614548. URL http://dx.doi.org/10.1145/3597638.3614548. 10.2

[101] Michele E. Gloede and Melissa K. Gregg. The fidelity of visual and auditory memory. *Psychonomic Bulletin & Review*, 26(4):1325–1332, April 2019. ISSN 1531-5320. doi: 10.3758/s13423-019-01597-7. URL https://doi.org/10.3758/s13423-019-01597-7. 1.1.1

[102] A. Jonathan R. Godfrey, Paul Murrell, and Volker Sorge. An accessible interaction model for data visualisation in statistics. In *Lecture Notes in Computer Science*, page 590–597. Springer International Publishing, 2018. ISBN 9783319942773. doi: 10.1007/978-3-319-94277-3_92. URL https://doi.org/10.1007/978-3-319-94277-3_92. 2.2.1, 4.3.1, 5.2, 5.3.1.1, 5.3.2.1

[103] Jenna L. Gorlewicz, Jennifer L. Tennison, P. Merlin Uesbeck, Margaret E. Richard, Hari P. Palani, Andreas Stefik, Derrick W. Smith, and Nicholas A. Giudice. Design guidelines and recommendations for multimodal, touchscreen-based graphics. *ACM Transactions on Accessible Computing*, 13(3):1–30, August 2020. ISSN 1936-7236. doi: 10.1145/3403933. URL https://doi.org/10.1145/3403933. 7.3.2, 7.4.1, 7.5.2

[104] David Graeber and Charlie Rose. Never cower to authority, 2006. URL https://www.youtube.com/watch?v=qt5op6wft-8. Accessed [2026]. 10.3

[105] R.L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters*, 1(4):132–133, 1972. ISSN 0020-0190. doi: 10.1016/0020-0190(72)90045-2. URL http://dx.doi.org/10.1016/0020-0190(72)90045-2. 6.5.4.2

[106] Colin M. Gray, Erik Stolterman, and Martin A. Siegel. Reprioritizing the relationship between hci research and practice. In *Proceedings of the 2014 conference on Designing interactive systems*, DIS '14, pages 725–734, New York, NY, USA, June 2014. Association for Computing Machinery, ACM. doi: 10.1145/2598510.2598595. 4.4

[107] 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. 1.1.2

[108] Catherine Grevet and Eric Gilbert. Piggyback prototyping: Using existing, large-scale social computing systems to prototype new ones. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, page 4047–4056. ACM, April 2015. doi: 10.1145/2702123.2702395. URL https://doi.org/10.1145/2702123.2702395. 2.1.2

[109] Karl Groves. Overlay fact sheet: What it is and how to make it work for you - karl groves, March 2023. URL https://karlgroves.com/overlay-fact-sheet-what-it-is-and-how-to-make-it-work-for-you/. 9.6

[110] Péter Ferenc Gyarmati, Dominik Moritz, Torsten Möller, and Laura Koesten. Structured visualization design knowledge for grounding generative reasoning and situated feedback, 2025. URL https://doi.org/10.48550/arxiv.2512.20306. 4.11

[111] Nathan Hahn, Joseph Chang, Ji Eun Kim, and Aniket Kittur. The knowledge accelerator: Big picture thinking in small pieces. In *Proceedings of the 2016 CHI Conference on*

Human Factors in Computing Systems, CHI'16, page 2258–2270. ACM, May 2016. doi: 10.1145/2858036.2858364. URL https://doi.org/10.1145/2858036.2858364. 2.1.1

[112] 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. 2.2.1, 4.3.1

[113] 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, page 1–11. ACM, December 2018. doi: 10.1145/3283458.3283478. URL https://doi.org/10.1145/3283458.3283478. 2.1.1

[114] 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, page 1–13. ACM, October 2024. doi: 10.1145/3663548.3675665. URL https://doi.org/10.1145/3663548.3675665. 6.1.1, 6.2, 8.6.2

[115] 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-environmental-design/, 2019. [Online]. 2.3, 3, 8.3.1

[116] Aimi Hamraie. *Building Access: Universal Design and the Politics of Disability*. University of Minnesota Press, Minneapolis, MN, 2017. ISBN 9781517901639. ISBN: 978-1-5179-0163-9. 2.3, 2.4.3

[117] Aimi Hamraie. Designing collective access: A feminist disability theory of universal design. In *Disability, space, architecture*, pages 78–297. Routledge, 2017. 2.3, 9.7.2

[118] Aimi Hamraie and Kelly Fritsch. Crip technoscience manifesto. *Catalyst: Feminism, Theory, Technoscience*, 5(1):1–33, April 2019. ISSN 2380-3312. doi: 10.28968/cftt.v5i1.29607. URL https://doi.org/10.28968/cftt.v5i1.29607. 2.3

[119] Gunnar Harboe and Elaine 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*, CHI '15, page 95–104. ACM, April 2015. doi: 10.1145/2702123.2702561. URL https://doi.org/10.1145/2702123.2702561. 6.6.3, 7.6.7, 8.7

[120] 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. 9.6, 10.1

[121] 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. 4.1.1, 6.2, 6.4, 10.2

[122] Jeffrey Heer and Maneesh Agrawala. Design considerations for collaborative visual analytics. In *2007 IEEE Symposium on Visual Analytics Science and Technology*, page 171–178. IEEE, October 2007. doi: 10.1109/vast.2007.4389011. URL https://doi.org/10.1109/vast.2007.4389011. 6.2, 7.3.1

[123] Jeffrey Heer and Dominik Moritz. Mosaic: An architecture for scalable & interoperable data views. *IEEE Transactions on Visualization and Computer Graphics*, page 1–11, 2023. ISSN 2160-9306. doi: 10.1109/tvcg.2023.3327189. URL https://doi.org/10.1109/tvcg.2023.3327189. 1.1.2, 2.2.3, 2.4.2, 3, 7.3.1, **G5:**, 7.5.3, 7.9.1

[124] 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. 10.3

[125] Thomas Hermann, Andy Hunt, and John G Neuhoff, editors. *The Sonification Handbook*. Logos Verlag Berlin, Berlin, Germany, December 2011. ISBN 9783832528195. ISBN: 978-3-8325-2819-5. 2.2.1, 6.3.2, 7.3.2

[126] 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, page 1–17. ACM, April 2023. doi: 10.1145/3544548.3581249. URL https://doi.org/10.1145/3544548.3581249. 2.1.2

[127] 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

[128] Highsoft. Highcharts - interactive charting library for developers. https://www.highcharts.com/, 2009. [Online]. 2.2.1, 4.3.2, 8.2

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

[130] Highsoft. Highcharts accessibility module: information and demos. https://highcharts.com/docs/accessibility/accessibility-module, 2018. Accessed: 2021-09-06. 4.6.3, 5.3.2.1

[131] 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

[132] Toan Hoang. The TableauMagic DataTables Extension. https://tableau.toanhoang.com/the-tableau-magic-datatables-extension-now-available/,

September 2018. Accessed: 2021-09-03. 4.3.2

[133] Megan Hofmann, Devva Kasnitz, Jennifer Mankoff, and Cynthia L Bennett. Living disability theory: Reflections on access, research, and design. In *Proceedings of the 22nd International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '20, page 1–13. ACM, October 2020. doi: 10.1145/3373625.3416996. URL http://dx.doi.org/10.1145/3373625.3416996. 2.3, 10.2

[134] Fred Hohman, Mary Beth Kery, Donghao Ren, and Dominik Moritz. Model compression in practice: Lessons learned from practitioners creating on-device machine learning experiences. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, page 1–18. ACM, May 2024. doi: 10.1145/3613904.3642109. URL http://dx.doi.org/10.1145/3613904.3642109. 2.1.2

[135] 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://doi.org/10.48550/arxiv.2401.15836. 6.3.2, 7.2

[136] 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, page 1–13. ACM, April 2022. doi: 10.1145/3491102.3517465. URL https://doi.org/10.1145/3491102.3517465. 6.3.2, 7.2

[137] 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

[138] 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. 1.1.1, 2.3, 3, 8.3.1, 9.7.2, 10.2

[139] 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, June 2013. Association for Computing Machinery, ACM. doi: 10.1145/2485760.2485883. Accessed: 2021-09-03. 2.1.1, 4.8, 5.3.3, 9.7.1, 10.2

[140] 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, page 11–18. ACM, October 2011. doi: 10.1145/2049536.2049541. URL https://doi.org/10.1145/2049536.2049541. 2.1.1, 7.5

[141] Edwin L. Hutchins, James D. Hollan, and Donald A. Norman. Direct manipulation inter-

faces. *Human–Computer Interaction*, 1(4):311–338, December 1985. ISSN 1532-7051. doi: 10.1207/s15327051hci0104_2. URL https://doi.org/10.1207/s15327051hci0104_2. 6.2, 6.3.1

[142] 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: inspiring design for and with families. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI03, page 17–24. ACM, April 2003. doi: 10.1145/642611.642616. URL https://doi.org/10.1145/642611.642616. 6.1.1, 6.2

[143] Web Accessibility Initiative. Web Accessibility Laws & Policies. https://www.w3.org/WAI/policies/, August 2021. Accessed: 2021-08-19. 4.3.3

[144] Lilly C. Irani and M. Six Silberman. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, CHI '13, page 611–620, New York, NY, USA, April 2013. Association for Computing Machinery, ACM. doi: 10.1145/2470654.2470742. URL https://doi.org/10.1145/2470654.2470742. 4.8

[145] Liz Jackson, Alex Haagaard, and Rua Williams. Disability Dongle. https://blog.castac.org/2022/04/disability-dongle/, 2022. 10.2

[146] 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. 10.3

[147] Yvonne Jansen, Pierre Dragicevic, Petra Isenberg, Jason Alexander, Abhijit Karnik, Johan Kildal, Sriram Subramanian, and Kasper Hornbæk. *Opportunities and Challenges for Data Physicalization*, page 3227–3236. CHI '15. ACM, New York, NY, USA, April 2015. doi: 10.1145/2702123.2702180. URL https://doi.org/10.1145/2702123.2702180. Accessed: 2021-09-03. 4.3.1

[148] Chandrika Jayant, Matt Renzelmann, Dana Wen, Satria Krisnandi, Richard Ladner, and Dan Comden. Automated tactile graphics translation: in the field. In *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, ASSETS07, page 75–82, New York, NY, USA, October 2007. ACM. ISBN 978-1-59593-573-1. doi: 10.1145/1296843.1296858. URL https://doi.org/10.1145/1296843.1296858. Accessed: 2021-09-03. 4.3.1

[149] Bin Jiang, Ferjan Ormeling, and Wolfgang Kainz. Visualization support for fuzzy spatial analysis. In *1995 ACSM/ASPRS Annual Convention & Exposition: Technical Papers*, volume 4: Auto Carto 12, pages 291–310, United States, 1994. American Congress on Surveying and Mapping. ISBN 1-57083-019-3. 12th International Symposium on Computer-Assisted Cartography, Auto-Carto 1995, Auto-Carto XII ; Conference date: 27-02-1995 Through 01-03-1995. 7.8.2

[150] 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, page 1–15. ACM, October 2023. doi: 10.1145/3597638.3608423. URL https://doi.org/10.1145/3597638.3608423. 2.1.1

[151] 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, page 1–14. ACM, May 2024. doi: 10.1145/3613904.3641970. URL https://doi.org/10.1145/3613904.3641970. 2.4.3, 3, 6.3.2, 6.5.2, 8.3.3

[152] Ger Joyce, Mariana Lilley, Trevor Barker, and Amanda Jefferies. Mobile Application Usability: Heuristic Evaluation and Evaluation of Heuristics. In Ben Amaba, editor, *Advances in Human Factors, Software, and Systems Engineering*, Advances in Intelligent Systems and Computing, page 77–86, Cham, 2016. Springer International Publishing, Springer International Publishing. ISBN 9783319419350. doi: 10.1007/978-3-319-41935-0_8. URL https://doi.org/10.1007/978-3-319-41935-0_8. 4.3.4

[153] 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, page 1–19. ACM, April 2022. doi: 10.1145/3491102.3517630. URL https://doi.org/10.1145/3491102.3517630. 2.2.3, 3, 6.2, 6.3.3, 6.8.2, 8.3.1, 9.1, 10.2

[154] 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. ISSN 2160-9306. doi: 10.1109/tvcg.2021.3114846. URL https://doi.org/10.1109/tvcg.2021.3114846. 2.2.1, 2.4, 4.3.1, 5.2, 5.3.1.2, 6.3.2, 7.2, 7.3.2

[155] Shaun K. Kane, Jeffrey P. Bigham, and Jacob O. Wobbrock. Slide rule: Making mobile touch screens accessible to blind people using multi-touch interaction techniques. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS08, page 73–80. ACM, October 2008. doi: 10.1145/1414471.1414487. URL https://doi.org/10.1145/1414471.1414487. 2.4.2

[156] 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, November 2022. ISSN 1557-7325. doi: 10.1145/3530013. URL https://doi.org/10.1145/3530013. 2.1.1

[157] 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. 10.3

[158] 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, page 1–13. ACM, April 2023. doi: 10.1145/3544548.3581139. URL https://doi.org/10.1145/3544548.3581139. 6.3.2

[159] 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, page 1–19. ACM, May 2024. doi: 10.1145/3613904.3642442. URL https://doi.org/10.1145/3613904.3642442. 6.3.2, 6.3.3, 7.2, 7.3.2

[160] 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*, CHI '23, page 1–15. ACM, April 2023. doi: 10.1145/3544548.3581532. URL https://doi.org/10.1145/3544548.3581532. 2.2.1, 5.3.1.2, 6.3.2, 7.3.3

[161] 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, 4.3.1, 5.6, 6.3.2

[162] 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. 2.4.3, 6.3.2, 6.3.3, 8.1.1, 8.5.1

[163] Rebecca Klein. Power BI Accessibility Best Practices. https://rklein324.github.io/PowerBIAccessibility/, 2020. Accessed: 2021-09-06. 4.3.2

[164] 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

[165] Nicolas Kruchten, Jon Mease, and Dominik Moritz. Vegafusion: Automatic server-side scaling for interactive vega visualizations. In *2022 IEEE Visualization and Visual Analytics (VIS)*, page 11–15. IEEE, October 2022. doi: 10.1109/vis54862.2022.00011. URL http://dx.doi.org/10.1109/VIS54862.2022.00011. 5.8

[166] Richard E. Ladner. Design for user empowerment. *Interactions*, 22(2):24–29, February 2015. ISSN 1558-3449. doi: 10.1145/2723869. URL https://doi.org/10.1145/2723869. Accessed: 2022-03-07. 4.4.3

[167] Chufan Lai, Zhixian Lin, Ruike Jiang, Yun Han, Can Liu, and Xiaoru Yuan. *Automatic Annotation Synchronizing with Textual Description for Visualization*, page 1–13. CHI '20. ACM, New York, NY, USA, April 2020. doi: 10.1145/3313831.3376443. URL https://doi.org/10.1145/3313831.3376443. Accessed: 2021-09-06. 4.3.1

[168] Susan J. Lederman, Georgie Thorne, and Bill Jones. Perception of texture by vision and touch: Multidimensionality and intersensory integration. *Journal of experimental psy-*

chology. Human perception and performance, 12(2):169–180, 1986. ISSN 0096-1523. doi: 10.1037//0096-1523.12.2.169. URL https://doi.org/10.1037//0096-1523.12.2.169. 4.3.1

[169] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. Evaluation strategies for HCI toolkit research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–17. ACM, April 2018. doi: 10.1145/3173574.3173610. URL https://doi.org/10.1145/3173574.3173610. 2.1.2, 2.2.3

[170] 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, March 2020. ISSN 1558-1756. doi: 10.1109/MCG.2020.2968244. URL https://doi.org/10.1109/mcg.2020.2968244. 4.3.1

[171] Xiaoou Li, Laura A. Carlson, Weimin Mou, Mark R. Williams, and Jared E. Miller. Describing spatial locations from perception and memory: The influence of intrinsic axes on reference object selection. *Journal of Memory and Language*, 65(2):222–236, August 2011. ISSN 0749-596X. doi: 10.1016/j.jml.2011.04.001. URL https://doi.org/10.1016/j.jml.2011.04.001. 7.4.1

[172] 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

[173] 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. 2.4.2, 3, 6.2, 7.2, 7.3.1, **G5:**, 7.10, 9.6

[174] Tania Llana, Marta Mendez, M.-Carmen Juan, and Magdalena Mendez-Lopez. Navigational object-location memory assessment in real and virtual environments: A systematic review. *Behavioural Brain Research*, 480:115388, March 2025. ISSN 0166-4328. doi: 10.1016/j.bbr.2024.115388. URL https://doi.org/10.1016/j.bbr.2024.115388. 7.4.1, **G1:**

[175] 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

[176] Panagiotis Louridas. Design as bricolage: anthropology meets design thinking. *Design Studies*, 20(6):517–535, November 1999. ISSN 0142-694X. doi: 10.1016/s0142-694x(98)00044-1. URL https://doi.org/10.1016/s0142-694x(98)00044-1. 5.6

[177] 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. ISSN 2160-9306. doi: 10.1109/tvcg.2021.3114770. URL https://doi.org/10.1109/tvcg.2021.311477

0. 2.2.1, 2.4, 4.3.1, 5.3.1.2, 5.3.1.3, 5.5.1.1, 5.5.3, 5.6, 6.3.2, 7.2, 7.3.2

[178] Alan Lundgard, Crystal Lee, and Arvind Satyanarayan. Sociotechnical considerations for accessible visualization design. In *2019 IEEE Visualization Conference (VIS)*, page 16–20. IEEE, October 2019. doi: 10.1109/visual.2019.8933762. URL https://doi.org/10.1109/visual.2019.8933762. 4.3.1, 4.3.3, 5.3.1.3, 5.5.1.1, 5.5.3, 5.6, 6.3.3, 10.2

[179] 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, page 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

[180] 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

[181] Kelly Mack, Emma McDonnell, Dhruv Jain, Lucy Lu Wang, Jon E. Froehlich, and Leah Findlater. What do we mean by "accessibility research"?: A literature survey of accessibility papers in chi and assets from 1994 to 2019. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, page 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. 2.2.3, 4.3.1

[182] Shalini Madan, Sreelakshmi Surabiyil Bindu, and Venkatesh Potluri. Vibe check: Accessibility heuristics for vibe coding interfaces. In *Proceedings of the 23rd International Web for All Conference*, W4A '26, page 229–241. ACM, April 2026. doi: 10.1145/3800424.3800455. URL http://dx.doi.org/10.1145/3800424.3800455. 4.11

[183] 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. 10.3

[184] Tlamelo Makati, Garreth W. Tigwell, and Kristen Shinohara. The promise and pitfalls of web accessibility overlays for blind and low vision users. In *The 26th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '24, page 1–12. ACM, October 2024. doi: 10.1145/3663548.3675650. URL http://dx.doi.org/10.1145/3663548.3675650. 9.6

[185] 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

[186] Jennifer Mankoff, Gillian R. Hayes, and Devva Kasnitz. Disability studies as a source of critical inquiry for the field of assistive technology. In *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*, ASSETS '10, page

3–10, New York, NY, USA, October 2010. Association for Computing Machinery, ACM. doi: 10.1145/1878803.1878807. URL https://doi.org/10.1145/1878803.1878807. Accessed: 2021-09-06. 4.4.3, 10.2

[187] 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. 2.2.1, 4.3.1, 6.3.2

[188] Mapbox. Mapbox gl accessibility. https://github.com/mapbox/mapbox-gl-accessibility, August 2021. Accessed: 2021-09-06. 4.3.2

[189] Mapbox. Mapbox-gl js. https://www.npmjs.com/package/mapbox-gl, 2022. Accessed: 2021-09-03. 4.3.2

[190] 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. 1.1.1, 9.1

[191] 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: a call to action. *Interactions*, 28(3):47–51, April 2021. ISSN 1558-3449. doi: 10.1145/3457875. URL https://doi.org/10.1145/3457875. 2.2.3, 4.3.1, 5.3.1, 5.3.3, 5.6, 6.3.2, 7.2, 8.3.1

[192] 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. 6.3.2, 7.2

[193] 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. 4.3.1, 4.3.4

[194] Michele Mauri, Elena Aversa, Anna Cattaneo, and María De Los Ángeles Briones Rojas. Ddenso: an evaluation-driven methodology for information visualisations. In *Proceedings of DRS*, DRS2026. Design Research Society, June 2026. doi: 10.21606/drs.2026.1396. URL http://dx.doi.org/10.21606/drs.2026.1396. 4.11

[195] Melanie Mazanec. Semiotic. https://semiotic.nteract.io/guides/accessibility, 2019. Accessed: 2021-09-03. 4.3.2

[196] 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*, NORDICHI06, page 145–154, New York, NY, USA, October 2006. Association for Computing Machinery, ACM. doi: 10.1145/1182475.1182491. URL https://doi.org/10.1145/1182475.1182491. Accessed: 2021-09-03. 2.2.1, 4.3.1

[197] Andrew M. McNutt. No grammar to rule them all: A survey of json-style dsls for visualization. *IEEE Transactions on Visualization and Computer Graphics*, page 1–11, 2022. ISSN 2160-9306. doi: 10.1109/tvcg.2022.3209460. URL https://doi.org/10.1109/tvcg.2022.3209460. 2.1.2, 6.2

[198] Andrew Michael McNutt. *Understanding and Enhancing JSON-based DSL Interfaces for Visualization*. PhD thesis, University of Chicago, 2023. URL https://doi.org/10.31219/osf.io/fy246. 2.1.2

[199] 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, page 1–17. ACM, October 2025. doi: 10.1145/3663547.3746342. URL https://doi.org/10.1145/3663547.3746342. 2.2.1, 6.3.2, 6.3.3, 7.2

[200] Mia Mingus. Access intimacy: The missing link. https://leavingevidence.wordpress.com/2011/05/05/access-intimacy-the-missing-link/, May 2011. Blog post, *Leaving Evidence*. 1.1.1, 2.3, 9.7.2

[201] Ehsan Mohammadi, Mike Thelwall, Stefanie Haustein, and Vincent Larivière. Who reads research articles? an altmetrics analysis of ¡scp¿m¡/scp¿endeley user categories. *Journal of the Association for Information Science and Technology*, 66(9):1832–1846, April 2015. ISSN 2330-1643. doi: 10.1002/asi.23286. URL http://dx.doi.org/10.1002/asi.23286. 10.2

[202] 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. 10.3

[203] Priscilla Moraes, Gabriel Sina, Kathleen McCoy, and Sandra Carberry. Evaluating the accessibility of line graphs through textual summaries for visually impaired users. In *Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility - ASSETS '14*, ASSETS '14, page 83–90, New York, NY, USA, October 2014. Association for Computing Machinery, ACM Press. doi: 10.1145/2661334.2661368. URL https://doi.org/10.1145/2661334.2661368. 4.3.1

[204] Dominik Moritz, Bill Howe, and Jeffrey Heer. Falcon: Balancing interactive latency and resolution sensitivity for scalable linked visualizations. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–11. ACM, May 2019. doi: 10.1145/3290605.3300924. URL https://doi.org/10.1145/3290605.3300924. 3, 7.3.1

[205] 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, page 1–17. ACM, April 2026. doi: 10.1145/3772318.3790289. URL https://doi.org/10.1145/3772318.3790289. Best Paper Award. 6.3.3

[206] Tamara Munzner. *Visualization Analysis and Design*. A K Peters/CRC Press, December

2014. ISBN 9780429088902. doi: 10.1201/b17511. URL http://dx.doi.org/10.1201/b17511. 4.1.1

[207] Brad A. Myers. Taxonomies of visual programming and program visualization. *Journal of Visual Languages & Computing*, 1(1):97–123, March 1990. ISSN 1045-926X. doi: 10.1016/S1045-926X(05)80036-9. URL https://doi.org/10.1016/S1045-926X(05)80036-9. 6.3.1

[208] I. Nassi and B. Shneiderman. Flowchart techniques for structured programming. *ACM SIGPLAN Notices*, 8(8):12–26, August 1973. ISSN 1558-1160. doi: 10.1145/953349.953350. URL https://doi.org/10.1145/953349.953350. 6.3.1

[209] Jakob Nielsen. 10 Usability Heuristics for User Interface Design. https://www.nngroup.com/articles/ten-usability-heuristics/, 1994. Accessed: 2021-09-07. 4.3.4

[210] Jakob Nielsen. *Heuristic evaluation*, pages 25–62. John Wiley & Sons, Inc., USA, June 1994. Accessed: 2021-09-06. 4.3.4

[211] 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. 4.3.1

[212] Jason Obeid and Enamul Hoque. Chart-to-Text: Generating Natural Language Descriptions for Charts by Adapting the Transformer Model. *arXiv Preprint*, November 2020. Accessed: 2021-09-06. 4.3.1

[213] Catherine A. Okoro, NaTasha D. Hollis, Alissa C. Cyrus, and Shannon Griffin-Blake. Prevalence of disabilities and health care access by disability status and type among adults — united states, 2016. *MMWR. Morbidity and Mortality Weekly Report*, 67(32):882–887, August 2018. ISSN 1545-861X. doi: 10.15585/mmwr.mm6732a3. URL https://doi.org/10.15585/mmwr.mm6732a3. 1.1.1, 4.2

[214] Manuel M. Oliveira. Towards More Accessible Visualizations for Color-Vision-Deficient Individuals. *Computing in Science & Engineering*, 15(5):80–87, September 2013. doi: 10.1109/MCSE.2013.113. Conference Name: Computing in Science Engineering. 4.3.1, 4.3.4

[215] World Health Organization. *World report on vision*. World Health Organization, 2019. ISBN 978-92-4-151657-0. Accessed: 2021-09-03. 4.2

[216] Daniela Quiñones Otey. A methodology to develop usability / user experience heuristics. In *Proceedings of the XVIII International Conference on Human Computer Interaction*, Interacción '17, pages 1–2, New York, NY, USA, September 2017. Association for Computing Machinery, ACM. doi: 10.1145/3123818.3133832. 4.3.4

[217] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, January 1979. ISSN 2168-2909. doi: 10.1109/tsmc.1979.4310076. URL http://dx.doi.org/10.1109/TSMC.

1979.4310076. 6.5.4.3

[218] 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

[219] Swaroop Panda. Llm-based visualization evaluation: How well do literacy-stratified personas approximate human judgments?, 2026. URL https://doi.org/10.48550/arxiv.2606.10095. 4.11

[220] Amandalynne Paullada, Inioluwa Deborah Raji, Emily M. Bender, Emily Denton, and Alex Hanna. Data and its (dis)contents: A survey of dataset development and use in machine learning research. *Patterns*, 2(11):100336, November 2021. ISSN 2666-3899. doi: 10.1016/j.patter.2021.100336. URL http://dx.doi.org/10.1016/j.patter.2021.100336. 9.6

[221] Jorge Poco and Jeffrey Heer. Reverse-engineering visualizations: Recovering visual encodings from chart images. *Computer Graphics Forum*, 36(3):353–363, 2017. ISSN 1467-8659. doi: 10.1111/cgf.13193. URL http://dx.doi.org/10.1111/cgf.13193. 6.5.4.3

[222] 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, page 433–442, New York, NY, USA, May 2012. Association for Computing Machinery, ACM. doi: 10.1145/2207676.2207736. URL https://doi.org/10.1145/2207676.2207736. 2.2.2, 2.3, 3, 4.2, 8.6.2

[223] Xin Qian, Eunyee Koh, Fan Du, Sungchul Kim, Joel Chan, Ryan A. Rossi, Sana Malik, and Tak Yeon Lee. *Generating Accurate Caption Units for Figure Captioning*, page 2792–2804. WWW '21. ACM, New York, NY, USA, April 2021. doi: 10.1145/3442381.3449923. URL https://doi.org/10.1145/3442381.3449923. Accessed: 2021-09-06. 4.3.1

[224] 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, page 1–8. ACM, October 2023. doi: 10.1145/3597638.3614546. URL https://doi.org/10.1145/3597638.3614546. 6.4, 7.3.2

[225] 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

[226] Blake Ellis Reid. The curb-cut effect and the perils of accessibility without disability. *SSRN Electronic Journal*, 2022. ISSN 1556-5068. doi: 10.2139/ssrn.4262991. URL https://doi.org/10.2139/ssrn.4262991. 1.1.1, 5.6

[227] 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. 6.3.2, 7.2, 7.3.3

[228] Yvonne Rogers, Jeni Paay, Margot Brereton, Kate L. Vaisutis, Gary Marsden, and Frank Vetere. Never too old: engaging retired people inventing the future with makey makey. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, page 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

[229] Michele Root-Bernstein and Robert Root-Bernstein. Imaginary worldplay in childhood and maturity and its impact on adult creativity. *Creativity Research Journal*, 18(4): 405–425, October 2006. ISSN 1040-0419. doi: 10.1207/s15326934crj1804_1. URL http://dx.doi.org/10.1207/s15326934crj1804_1. 11

[230] 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. 1.1.1

[231] Azriel Rosenfeld and John L. Pfaltz. Sequential operations in digital picture processing. *Journal of the ACM*, 13(4):471–494, October 1966. ISSN 1557-735X. doi: 10.1145/321356.321357. URL http://dx.doi.org/10.1145/321356.321357. 6.5.4.3

[232] Maurício Rossi de Oliveira and Celmar Guimarães da Silva. Adapting heuristic evaluation to information visualization - a method for defining a heuristic set by heuristic grouping. In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, page 225–232. INSTICC, SCITEPRESS - Science and Technology Publications, 2017. ISBN 978-989-758-228-8. doi: 10.5220/0006133202250232. URL https://doi.org/10.5220/0006133202250232. 4.3.4

[233] 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. 10.3

[234] 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. 10.3

[235] 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

[236] City San Francisco. Covid-19 data and reports San Francisco. https://sf.gov/resource/2021/covid-19-data-and-reports, 2021. Accessed: 2021-09-03. 4.3.2

[237] 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. ISSN 1745-3755. doi: 10.1080/15710882.2014.888183. URL https://doi.org/10.1080/15710882.2014.888183. 2.1.1

[238] Beatriz Sousa Santos, Samuel Silva, and Paulo Dias. Heuristic Evaluation in Visualization: An Empirical Study : Position paper. In *2018 IEEE Evaluation and Beyond - Methodological Approaches for Visualization (BELIV)*, page 78–85. IEEE, October 2018. doi: 10.1109/BELIV.2018.8634108. URL https://doi.org/10.1109/beliv.2018.8634108. 4.3.4

[239] 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, December 2022. ISSN 1557-7341. doi: 10.1145/3543509. URL https://doi.org/10.1145/3543509. 5.3.3

[240] SAS. Sas Graphics Accelerator. https://support.sas.com/software/products/graphics-accelerator/, 2022. Accessed: 2021-09-08. 2.2.1, 4.3.2

[241] 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. ISSN 1077-2626. doi: 10.1109/tvcg.2016.2599030. URL https://doi.org/10.1109/tvcg.2016.2599030. 1.1.2, 2.2.3, 5.3.2.2, 5.3.3, 5.4.3.1, 5.5.2.1, 5.8, 6.2, 6.3.1, 6.5.2, 6.5.4.1, 9.5

[242] Manolis Savva, Nicholas Kong, Arti Chhajta, Li Fei-Fei, Maneesh Agrawala, and Jeffrey Heer. Revision: automated classification, analysis and redesign of chart images. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, page 393–402. ACM, October 2011. doi: 10.1145/2047196.2047247. URL http://dx.doi.org/10.1145/2047196.2047247. 6.5.4.3

[243] 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, May 2021. Association for Computing Machinery. Accessed: 2021-09-06. 2.2.1, 4.3.1

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

[245] Doug Schepers. Why Accessibility Is at the Heart of Data Visualization by Doug Schepers Nightingale Medium. https://medium.com/nightingale/accessibility-is-at-the-heart-of-data-visualization-64a38d6c505b, 2020. Accessed: 2021-09-06. 4.4.2

[246] Doug Schepers. On underlays. Internet Archive, 2026. URL https://web.archive.org/web/20260616140431/https://www.linkedin.com/posts/daschepers_accessibility-overlays-are-good-for-business-activity-7317652815399587840-lwVL?utm_source=share&utm_medium=

member_desktop&rcm=ACoAADDAwBkBOdoW11I9B5DHy57VfR5jIs33Kq0. Archived version available at https://web.archive.org/web/20260616140431/https://www.linkedin.com/posts/daschepers_accessibility-overlays-are-good-for-business-activity-7317652815399587840-lwVL?utm_source=share&utm_medium=member_desktop&rcm=ACoAADDAwBkBOdoW11I9B5DHy57VfR5jIs33Kq0; original URL: linkedin.com/posts/daschepers_accessibility-overlays-are-good-for-business-activity-7317652815399587840-lwVL. Accessed: 2026-06-16. 9.6

[247] William Schiff and Emerson Foulke, editors. *Tactual Perception*. Cambridge University Press, Cambridge, England, March 1982. ISBN 0521240956. 2.2.1, 4.3.1, 7.3.2

[248] Jochen Schneider. Constructing the Yellow Brick Road: Route Bricks on Virtual Tactile Maps, 2000. 4.3.1

[249] Jean Scholtz. Developing guidelines for assessing visual analytics environments. *Information Visualization*, 10(3):212–231, July 2011. doi: 10.1177/1473871611407399. Accessed: 2022-03-06. 4.3.4

[250] Jonathan Schwabish, Susan J Popkin, Alice Feng, Doug Schepers, Frank Elavsky, Elizabeth Hare, Leonie Watson, Sarah Fossheim, Larene Le Gassick, Dax Castro, Amy Cesal, and Melanie Mazanec. Do no harm guide: Centering accessibility in data visualization. https://www.urban.org/research/publication/do-no-harm-guide-centering-accessibility-data-visualization, 2022. [Accessed 01-04-2025]. 4.11

[251] Donald Schön and John Bennett. Reflective conversation with materials, April 1996. URL https://doi.org/10.1145/229868.230044. 6.8.1, 9.3

[252] Muthu Selvam. Implementing ai-driven web accessibility: Wcag 2.2 features to improve usability for users with disabilities. In *2025 3rd International Conference on Artificial Intelligence, Blockchain, and Internet of Things (AIBThings)*, pages 1–6. IEEE, September 2025. doi: 10.1109/aibthings66987.2025.11296155. URL http://dx.doi.org/10.1109/AIBThings66987.2025.11296155. 4.11

[253] 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, page 1–22. ACM, May 2024. doi: 10.1145/3613904.3642730. URL https://doi.org/10.1145/3613904.3642730. 6.3.2, 10.2

[254] 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, pages 3389–3406. ACM, July 2024. doi: 10.1145/3643834.3661508. URL http://dx.doi.org/10.1145/3643834.3661508. 10.3

[255] Ather Sharif and Babak Forouraghi. evographs — a jquery plugin to create web accessible graphs. In *2018 15th IEEE Annual Consumer Communications & Networking Conference*

(CCNC), page 1–4. IEEE, January 2018. doi: 10.1109/ccnc.2018.8319239. URL https://doi.org/10.1109/ccnc.2018.8319239. 4.3.1, 5.3.1.1, 6.3.3

[256] 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, page 1–16. ACM, October 2021. doi: 10.1145/3441852.3471202. URL https://doi.org/10.1145/3441852.3471202. 2.2.1, 4.3.1, 4.3.3, 5.2, 5.3.1.2, 6.3.2, 6.3.3, 7.3.3, 9.1

[257] 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*, CHI '22, page 1–19. ACM, April 2022. doi: 10.1145/3491102.3517431. URL https://doi.org/10.1145/3491102.3517431. 2.2.1, 5.3.1.1, 6.3.3

[258] 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, page 1–20. ACM, October 2024. doi: 10.1145/3663548.3675625. URL https://doi.org/10.1145/3663548.3675625. 2.2.3, 3, 6.2, 6.3.3, 6.8.2, 8.3.1, 9.1, 10.2

[259] Ather Sharif, Venkatesh Potluri, Jazz Rui Xia Ang, Jacob O. Wobbrock, and Jennifer Mankoff. Touchpad mapper: Examining information consumption from 2d digital content using touchpads by screen-reader users. In *The 26th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '24, page 1–4. ACM, October 2024. doi: 10.1145/3663548.3688505. URL https://doi.org/10.1145/3663548.3688505. 7.3.4, 7.5.2

[260] Ashley Shew. *Against Technoableism: Rethinking Who Needs Improvement*. W. W. Norton, New York, NY, 2023. ISBN 9781324036661. ISBN: 978-1-324-03666-1. 2.3

[261] Lei Shi, Idan Zelzer, Catherine Feng, and Shiri Azenkot. Tickers and talker: An accessible labeling toolkit for 3d printed models. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI'16, page 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. 4.3.1

[262] Shneiderman. Direct manipulation: A step beyond programming languages. *Computer*, 16(8):57–69, August 1983. ISSN 0018-9162. doi: 10.1109/mc.1983.1654471. URL https://doi.org/10.1109/mc.1983.1654471. 6.3.1

[263] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *The Craft of Information Visualization*, page 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

[264] Ben Shneiderman, Christopher Williamson, and Christopher Ahlberg. Dynamic queries: database searching by direct manipulation. In *Proceedings of the SIGCHI conference on*

Human factors in computing systems - CHI '92, CHI '92, page 669–670. ACM Press, 1992. doi: 10.1145/142750.143082. URL https://doi.org/10.1145/142750.143082. 6.2, 7.3.1, **G6:**

[265] 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*, ASSETS '21, page 1–15. ACM, October 2021. doi: 10.1145/3441852.3471212. URL https://doi.org/10.1145/3441852.3471212. 5.3.3

[266] 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

[267] Aleksandra Slavkovic and Karen Cross. Novice heuristic evaluations of a complex interface. In *CHI '99 extended abstracts on Human factors in computing systems - CHI '99*, CHI '99, page 304, New York, NY, USA, May 1999. Association for Computing Machinery, ACM Press. doi: 10.1145/632716.632902. URL https://doi.org/10.1145/632716.632902. 4.3.4

[268] Jim Smiley, Benjamin Lee, Siddhant Tandon, Maxime Cordeil, Lonni Besançon, Jarrod Knibbe, Bernhard Jenny, and Tim Dwyer. The made-axis: A modular actuated device to embody the axis of a data dimension. *Proceedings of the ACM on Human-Computer Interaction*, 5(ISS):1–23, November 2021. ISSN 2573-0142. doi: 10.1145/3488546. URL https://doi.org/10.1145/3488546. 7.3.4, 7.5.2

[269] Jim Smiley, Swamy Ananthanarayan, Leona M Holloway, Matthew Butler, and Tim Dwyer. Magnepins: A modular, affordable, and diy refreshable braille and tactile display. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*, UIST '25, pages 1–12. ACM, September 2025. doi: 10.1145/3746059.3747692. URL https://doi.org/10.1145/3746059.3747692. 7.2, 7.5

[270] David Canfield Smith. *Pygmalion: A Computer Program to Model and Stimulate Creative Thought*. Birkhäuser Basel, 1977. ISBN 9783034857444. doi: 10.1007/978-3-0348-5744-4. URL https://doi.org/10.1007/978-3-0348-5744-4. Reissue of the author's 1975 Stanford University doctoral dissertation. 6.3.1

[271] Volker Sorge. Polyfilling accessible chemistry diagrams. In *Lecture Notes in Computer Science*, page 43–50. Springer International Publishing, 2016. ISBN 9783319412641. doi: 10.1007/978-3-319-41264-1_6. URL https://doi.org/10.1007/978-3-319-41264-1_6. 2.2.1, 4.3.1, 5.2, 5.3.1.1, 5.3.2.1, 6.3.2

[272] 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

[273] Laura South and Michelle Borkin. Generating Seizure-Inducing Sequences with Interactive Visualizations. In *Proc. Of the IEEE VIS 2020 posters*. Center for Open Science, October 2020. doi: 10.31219/osf.io/85gwy. URL https://doi.org/10.31219/osf.io/85gwy. 4.3.1

[274] Laura South, David Saffo, and Michelle Borkin. Detecting and Defending Against

Seizure-Inducing GIFs in Social Media. Technical report, OSF Preprints, January 2021. URL https://doi.org/10.31219/osf.io/4kgu6. Accessed: 2021-08-20. 4.3.1

[275] Katta Spiel, Kathrin Gerling, Cynthia L. Bennett, Emeline Brulé, Rua M. Williams, Jennifer Rode, and Jennifer Mankoff. Nothing about us without us: Investigating the role of critical disability studies in hci. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–8. ACM, April 2020. doi: 10.1145/3334480.3375150. URL https://doi.org/10.1145/3334480.3375150. 1.1.1, 2.3, 9.7.3, 10.1, 10.2

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

[277] Maureen Stone, Danielle Albers Szafir, and Vidya Setlur. An engineering model for color difference as a function of size. *Color and Imaging Conference*, 22(1):253–258, November 2014. ISSN 2166-9635. doi: 10.2352/cic.2014.22.1.art00045. URL http://dx.doi.org/10.2352/CIC.2014.22.1.art00045. 4.1.1

[278] 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. 1.1.1

[279] Cella M Sum, Rahaf Alharbi, Franchesca Spektor, Cynthia L Bennett, Christina N Harrington, Katta Spiel, and Rua Mae Williams. Dreaming disability justice in hci. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, CHI '22, page 1–5. ACM, April 2022. doi: 10.1145/3491101.3503731. URL http://dx.doi.org/10.1145/3491101.3503731. 2.3, 10.2

[280] Cella M. Sum, Franchesca Spektor, Rahaf Alharbi, Leya Breanna Baltaxe-Admony, Erika Devine, Hazel Anneke Dixon, Jared Duval, Tessa Eagle, Frank Elavsky, Kim Fernandes, Leandro S. Guedes, Serena Hillman, Vaishnav Kameswaran, Lynn Kirabo, Tamanna Motahar, Kathryn E. Ringland, Anastasia Schaadhardt, Laura Scheepmaker, and Alicia Williamson. Challenging ableism: A critical turn toward disability justice in hci. *XRDS: Crossroads, The ACM Magazine for Students*, 30(4):50–55, June 2024. ISSN 1528-4980. doi: 10.1145/3665602. URL http://dx.doi.org/10.1145/3665602. 2.3, 10.2

[281] Ruoting Sun, Rong Luo, Xiwen Yao, Xinran She, Kotaro Hara, and Yang Jiao. Tactile data comics: A step-by-step multimodal presentation method on a refreshable tactile display for blind and visually impaired individuals. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, CHI EA '25, page 1–8, New York, NY, April 2025. ACM. doi: 10.1145/3706599.3720192. URL https://doi.org/10.1145/3706599.3720192. 7.2, 7.3.2

[282] Ivan E. Sutherland. Sketchpad: A man-machine graphical communication system. In *Proceedings of the May 21-23, 1963, spring joint computer conference on - AFIPS '63 (Spring)*, AFIPS '63 (Spring), page 329, New York, NY, USA, 1963. ACM Press. doi: 10.1145/1461551.1461591. URL https://doi.org/10.1145/1461551.1461

591. 6.3.1

[283] Henny Swan, Ian Pouncey, Heydon Pickering, and Léonie Watson. Inclusive Design Principles. https://inclusivedesignprinciples.org/, 2019. Accessed: 2021-09-07. 4.4.2

[284] Sarit Felicia Anais Szpiro, Shafeka Hashash, Yuhang Zhao, and Shiri Azenkot. How people with low vision access computing devices: Understanding challenges and opportunities. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '16, page 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, 4.3.1

[285] Pierre Tchounikine. Designing for appropriation: A theoretical account. *Human–Computer Interaction*, 32(4):155–195, July 2016. ISSN 1532-7051. doi: 10.1080/07370024.2016.1203263. URL https://doi.org/10.1080/07370024.2016.1203263. 2.1.2

[286] Tim Teitelbaum and Thomas Reps. The cornell program synthesizer. *Communications of the ACM*, 24(9):563–573, 1981. doi: 10.1145/358746.358755. URL https://doi.org/10.1145/358746.358755. 6.3.1

[287] 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. 1.1.1

[288] J. Thatcher. Screen reader/2: access to os/2 and the graphical user interface. In *Proceedings of the first annual ACM conference on Assistive technologies - Assets '94*, Assets '94, page 39–46. ACM Press, 1994. doi: 10.1145/191028.191039. URL http://dx.doi.org/10.1145/191028.191039. 6.3.1

[289] 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*, CHI '23, page 1–18. ACM, April 2023. doi: 10.1145/3544548.3581186. URL https://doi.org/10.1145/3544548.3581186. 2.2.1, 2.4.1, 5.2, 5.3.1.1, 5.4.1.1, 5.5.3, 6.3.2, 6.3.3, 6.4, 7.3.2

[290] 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, 10.2

[291] 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

[292] US Government, 10x. Improving Accessibility in Data Visualizations (formerly USWDS Visualization Tool) on 10x. https://trello.com/c/OOOres7a/63-improving-accessibility-

in-data-visualizations-formerly-uswds-visualization-tool, 2020. Accessed: 2022-03-04. 4.3.4

[293] US Government, General Services Administration. Data visualizations U.S. Web Design System (USWDS). https://designsystem.digital.gov/components/data-visualizations/, 2021. Accessed: 2022-03-04. 4.3.4

[294] Jacob VanderPlas, Brian Granger, Jeffrey Heer, Dominik Moritz, Kanit Wongsuphasawat, Arvind Satyanarayan, Eitan Lees, Ilia Timofeev, Ben Welsh, and Scott Sievert. Altair: Interactive statistical visualizations for python. *Journal of Open Source Software*, 3(32): 1057, December 2018. ISSN 2475-9066. doi: 10.21105/joss.01057. URL http://dx.doi.org/10.21105/joss.01057. 5.8

[295] Visa. Visa Chart Components. https://github.com/visa/visa-chart-components, 2022. Accessed: 2022-12-01. 2.2.1, 4.3.2, 5.1.1, 5.3.2.1

[296] W3C. Web content accessibility guidelines (wcag) 2.1. https://www.w3.org/TR/WCAG21/, 2018. Accessed: 2021-09-03. 2.2.2, 3, 4.3.1, 4.3.3, 8.5.1

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

[298] WAI. Understanding Success Criterion 1.3.1 — Understanding WCAG 2.0. https://www.w3.org/TR/UNDERSTANDING-WCAG20/content-structure-separation-programmatic.html, 2008. [Accessed 01-04-2025]. **G3:**

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

[300] WAI. Understanding success criterion 4.1.2: name, role, value. WCAG standard, W3C, 2016. Accessed: 2022-03-04. 2.2.2, 5.3.1.3

[301] WAI. Accessible rich internet applications (WAI-ARIA 1.1). Technical report, W3C, 2017. Accessed: 2022-12-11. 5.1.1, 3

[302] 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, 6.3.2

[303] WAI. Accessibility principles. WCAG Standard, W3C, 2019. URL https://www.w3.org/WAI/fundamentals/accessibility-principles/. Accessed: 2021-09-06. 4.2, 4.3.3

[304] WAI. W3C Accessibility Guidelines (WCAG) 3.0. WCAG standard, W3C, 2021. Accessed: 2022-03-04. 4.8

[305] Bruce N. Walker and Michael A. Nees. Theory of sonification. In Thomas Hermann, Andy Hunt, and John G. Neuhoff, editors, *The Sonification Handbook*, chapter 2, pages 9–39. Logos Verlag, Berlin, Germany, 2011. 2.2.1

[306] Chris Weaver. Multidimensional visual analysis using cross-filtered views. In *2008 IEEE Symposium on Visual Analytics Science and Technology*, page 163–170. IEEE, October

2008. doi: 10.1109/vast.2008.4677370. URL https://doi.org/10.1109/vast.2008.4677370. 3, 7.2, 7.3.1

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

[308] 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. 4.5.1, 9.7.1, 10.3

[309] Simon Wheatcroft. Making Graphs And Plots Accessible For The Blind. https://andadapt.com/making-graphs-and-plots-accessible-for-the-blind/, June 2020. Accessed: 2021-09-06. 4.3.2

[310] 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

[311] David Gray Widder and Richmond Wong. Thinking upstream: Ethics and policy opportunities in ai supply chains, 2023. URL https://doi.org/10.48550/arxiv.2303.07529. 1.1.2

[312] 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. 1.1.2

[313] Leland Wilkinson. The grammar of graphics. In *Handbook of Computational Statistics*, page 375–414. Springer Berlin Heidelberg, December 2011. ISBN 9783642215513. doi: 10.1007/978-3-642-21551-3_13. URL https://doi.org/10.1007/978-3-642-21551-3_13. 6.2, 9.5

[314] Rua M. Williams and LouAnne E. Boyd. Prefigurative politics and passionate witnessing. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '19, page 262–266. ACM, October 2019. doi: 10.1145/3308561.3355617. URL http://dx.doi.org/10.1145/3308561.3355617. 10.2

[315] Rua M. Williams, Kathryn Ringland, Amelia Gibson, Mahender Mandala, Arne Maibaum, and Tiago Guerreiro. Articulations toward a crip hci. *Interactions*, 28(3):28–37, April 2021. ISSN 1558-3449. doi: 10.1145/3458453. URL http://dx.doi.org/10.1145/3458453. 2.3, 10.2

[316] Rua Mae Williams, Louanne Boyd, and Juan E. Gilbert. Counterventions: a reparative reflection on interventionist hci. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, page 1–11. ACM, April 2023. doi: 10.1145/3544548.3581480. URL http://dx.doi.org/10.1145/3544548.3581480. 10.2

[317] 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.

2.2.3, 8.3.1

[318] 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://doi.org/10.48550/arxiv.2601.19168. 6.3.3

[319] Langdon Winner. Do artifacts have politics? In *Computer Ethics*, page 177–192. Routledge, May 2017. ISBN 9781315259697. doi: 10.4324/9781315259697-21. URL https://doi.org/10.4324/9781315259697-21. 1.1.2

[320] Jacob O. Wobbrock, Shaun K. Kane, Krzysztof Z. Gajos, Susumu Harada, and Jon Froehlich. Ability-based design: Concept, principles and examples. *ACM Transactions on Accessible Computing*, 3(3):1–27, April 2011. ISSN 1936-7236. doi: 10.1145/1952383.1952384. URL https://doi.org/10.1145/1952383.1952384. 1.1.2, 2.1.2, 2.4.3, 4.4.3, 5.6, 8.3.2

[321] 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. 10.3

[322] 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

[323] 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. 2.4.3, 3, 8.3.3, 8.4.1.3, 8.7.4.3

[324] 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, page 1–16, New York, NY, USA, May 2021. Association for Computing Machinery, ACM. doi: 10.1145/3411764.3445743. URL https://doi.org/10.1145/3411764.3445743. Accessed: 2021-09-03. 4.3.1, 8.3.1

[325] Cindy Xiong, Lisanne Van Weelden, and Steven Franconeri. The Curse of Knowledge in Visual Data Communication. *IEEE Transactions on Visualization and Computer Graphics*, 26(10):3051–3062, October 2020. ISSN 2160-9306. doi: 10.1109/TVCG.2019.2917689. URL https://doi.org/10.1109/tvcg.2019.2917689. Conference Name: IEEE Transactions on Visualization and Computer Graphics. 4.5.4

[326] 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

[327] Katherine Ye, Wode Ni, Max Krieger, Dor Ma'ayan, Jenna Wise, Jonathan Aldrich, Joshua Sunshine, and Keenan Crane. Penrose: from mathematical notation to beautiful diagrams. *ACM Transactions on Graphics*, 39(4), August 2020. ISSN 1557-7368. doi: 10.1145/3386569.3392375. URL https://doi.org/10.1145/3386569.3392375. 5.5.3.1

[328] 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. 1.1.1

[329] Xiaoyi Zhang, Tracy Tran, Yuqian Sun, Ian Culhane, Shobhit Jain, James Fogarty, and Jennifer Mankoff. Interactiles: 3d printed tactile interfaces to enhance mobile touchscreen accessibility. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '18, page 131–142. ACM, October 2018. doi: 10.1145/3234695.3236349. URL https://doi.org/10.1145/3234695.3236349. 7.3.4, 7.5.2

[330] Haixia Zhao, Catherine Plaisant, Ben Shneiderman, and Jonathan Lazar. Data sonification for users with visual impairment: A case study with georeferenced data. *ACM Transactions on Computer-Human Interaction*, 15(1):1–28, May 2008. ISSN 1557-7325. doi: 10.1145/1352782.1352786. URL https://doi.org/10.1145/1352782.1352786. 2.2.1, 4.3.1

[331] Jonathan Zong. Using real names of disabled participant-contributors to practice citational justice in accessibility. In *2025 IEEE Workshop on Accessible Data Visualization (AccessViz)*, page 30–33. IEEE, November 2025. doi: 10.1109/accessviz68666.2025.00011. URL https://doi.org/10.1109/accessviz68666.2025.00011. 2.3, 6.4

[332] 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. 2.2.1, 2.3, 2.4.1, 2.4.2, 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, 6.3.2, 6.3.3, 6.4, 7.2, 7.3.3

[333] 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

[334] 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, page 1–20. ACM, May 2024. doi: 10.1145/3613904.3641996. URL https://doi.org/10.1145/3613904.3641996. 2.2.1, 6.3.3, 6.8.4, 7.2, 7.3.2, 9.7.3