# COMBINING NATURAL LANGUAGE AND DIRECT MANIPULATION FOR HUMAN-DATA INTERACTION THROUGH VISUALIZATIONS

A Dissertation
Presented to
The Academic Faculty

By

Arjun Srinivasan

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Interactive Computing

Georgia Institute of Technology

December 2020

**COMBINING NATURAL LANGUAGE AND DIRECT MANIPULATION FOR HUMAN-DATA INTERACTION THROUGH VISUALIZATIONS**

Thesis committee:


Dr. John Stasko, Advisor
School of Interactive Computing
*Georgia Institute of Technology*

Dr. Bongshin Lee
Microsoft Research

Dr. Alex Endert
School of Interactive Computing
*Georgia Institute of Technology*

Dr. Vidya Setlur
Tableau Research

Dr. Keith Edwards
School of Interactive Computing
*Georgia Institute of Technology*



Date approved: July 29, 2020

*To my parents, and Seema*

## ACKNOWLEDGMENTS

Although a thesis is seemingly one's own work, the completion of this dissertation was only made possible by the support from my collaborators, friends, and family.

Anything I have been able to achieve over the past four years is in large part due to the support of my advisor, John Stasko. John's role in my PhD has been instrumental with him wearing many hats including a teacher, sounding-board, supporter, colleague, and a friend. His faith in me in terms of conducting research, mentoring other students, and teaching have kept me motivated throughout this journey and gave me the freedom to pursue research that I was passionate about. There has not been a single instance where I have gone into his office and not come out more positive than I was before I entered. For this unparalleled support, all the advice on conducting research, writing and reviewing papers and giving talks, the casual sports conversations, and unforgettable email subjects like "Accckkk," "heh-heh," "Apparently," "face," among many others, I am forever grateful.

I also want to thank my thesis committee, Alex Endert, Keith Edwards, Bongshin Lee, and Vidya Setlur, for their critical and insightful feedback on this dissertation, as well as the invaluable advice on identifying interesting questions, exploring solutions, and effectively presenting the findings. I especially want to thank Bongshin Lee for her mentorship and guidance over the past four years as it was her paper "Beyond Mouse and Keyboard: Expanding Design Considerations for Information Visualization Interactions" that kindled my interest in research at the intersection of novel user interfaces and information visualization in the first place.

Other than my committee members, I was also lucky to benefit from a wide array of mentors during my Master's and summer internships. Thank you to Steven Drucker, Jim Foley, Rahul Basole, Eytan Adar, Mira Dontcheva, Matthew Brehmer, Ken Hinckley, and Nathalie Henry Riche for their mentorship and support on different projects over the years. In particular, I want to thank Steven Drucker for being my "non-academic advisor." His

encouragement and belief in me throughout the PhD has played a significant role in my intellectual growth and self-confidence.

A sincere thanks also goes to all members of the Georgia Tech Visualization Lab for their constant support, for the unforgettable VisGiving memories, and feedback on paper drafts and practice talks. I particularly want to thank Hyunwoo Park, Alex Godwin, Ramik Sadana, and Chad Stolper for their mentorship during the initial phase of the PhD program, help with recording and editing videos for paper submissions, and the countless Fifa memories. A special thanks also goes to John Thompson, Emily Wall, Bahador Saket, Fred Hohman, and Arpit Narechania who have been an integral part of this journey—thank you for the spontaneous lunches, hours of throwing around of the lab "brain," and the pre- and post-conference trips.

Finally, I am forever indebted to my parents as without their prayers and sacrifices, I would never have had the privilege and freedom to pursue a PhD and find work that I felt passionate about. I am also grateful for my partner, Seema, for being a steady support system, especially over the past four years. Her continued understanding and patience helped me stay positive through all the highs and lows and motivated me to go the extra mile when needed!

---

Research for this thesis was funded in part by the National Science Foundation under Grant IIS-1717111.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

## SUMMARY

Visualization is an indispensable tool for human-data interaction, enabling people to better understand their data, identify patterns, and discover insights. Interaction plays a critical role in data visualization tools as it allows users to express their data-related goals and questions to the system. Traditionally, interaction in visualization tools is facilitated predominantly via a keyboard and mouse, following the window-menu-icon-pointer (WIMP) metaphor and the direct manipulation paradigm. However, recent advances in hardware and input recognition technology present the opportunity to reimagine interaction and explore new user experiences grounded in naturalistic human ways of interacting with people and objects in the real world.

This thesis explores the design of a novel class of multimodal data visualization interfaces that augment current interaction techniques in visualization systems with natural language. I start with an assessment of the role of natural language input in visualization tools, characterizing the goals it can help people accomplish. Subsequently, I describe the design and implementation of a series of multimodal visualization systems that combine natural language and direct manipulation. Through evaluations of these systems, I capture the strengths and challenges of multimodal visualization interfaces, and highlight how they accommodate varying user interaction patterns and preferences during visual analysis. Finally, to assist future research and development, I also contribute a toolkit to help designers/developers prototype natural language-based visualization systems.

# CHAPTER 1
# INTRODUCTION

Today, people have access to seemingly unlimited amounts of data. Both individuals and organizations seek to understand and gain insights from these data in order to make better decisions. By enabling people to leverage their perceptive skills and amplifying their cognitive capabilities, visualizations serve as powerful tools for data exploration, sensemaking, and communication [27].

Interaction is a critical component of visualizations as it allows people to express their data-related goals and questions. Interaction with current visualization tools occurs predominantly via a keyboard and mouse, following the *window-menu-icon-pointer (WIMP)* metaphor and *direct manipulation* (DM) paradigm [176]. With current tools, users typically create visualizations by specifying data attributes of interest through a series of drag-and-drop interactions or invoke analytic functions such as sorting, filtering, or finding correlations through tool-specific menus and control panels (Figure 1.1).

Although WIMP- and DM-based interfaces are clearly valuable and widely used, they also have some limitations. For instance, DM becomes challenging when a system needs to support complex tasks that require multiple steps since users need to navigate through layers of the graphical user interface to accomplish their task [59, 177]. Furthermore, directly manipulating multiple data points can be tedious, requiring users to repeatedly interact with points one at a time and perform desired operations [80]. Lastly, as interactive displays and portable devices become more popular platforms of human-computer interaction (HCI), a growing suite of settings simply do not provide the optimal affordances to design traditional WIMP interfaces. For instance, the smaller size of mobile devices restricts screen space that can be devoted to control panels. Similarly, the lack of the pixel-sized precision of a mouse pointer or interactions like 'hover' makes it challenging to support precise selection or pro-

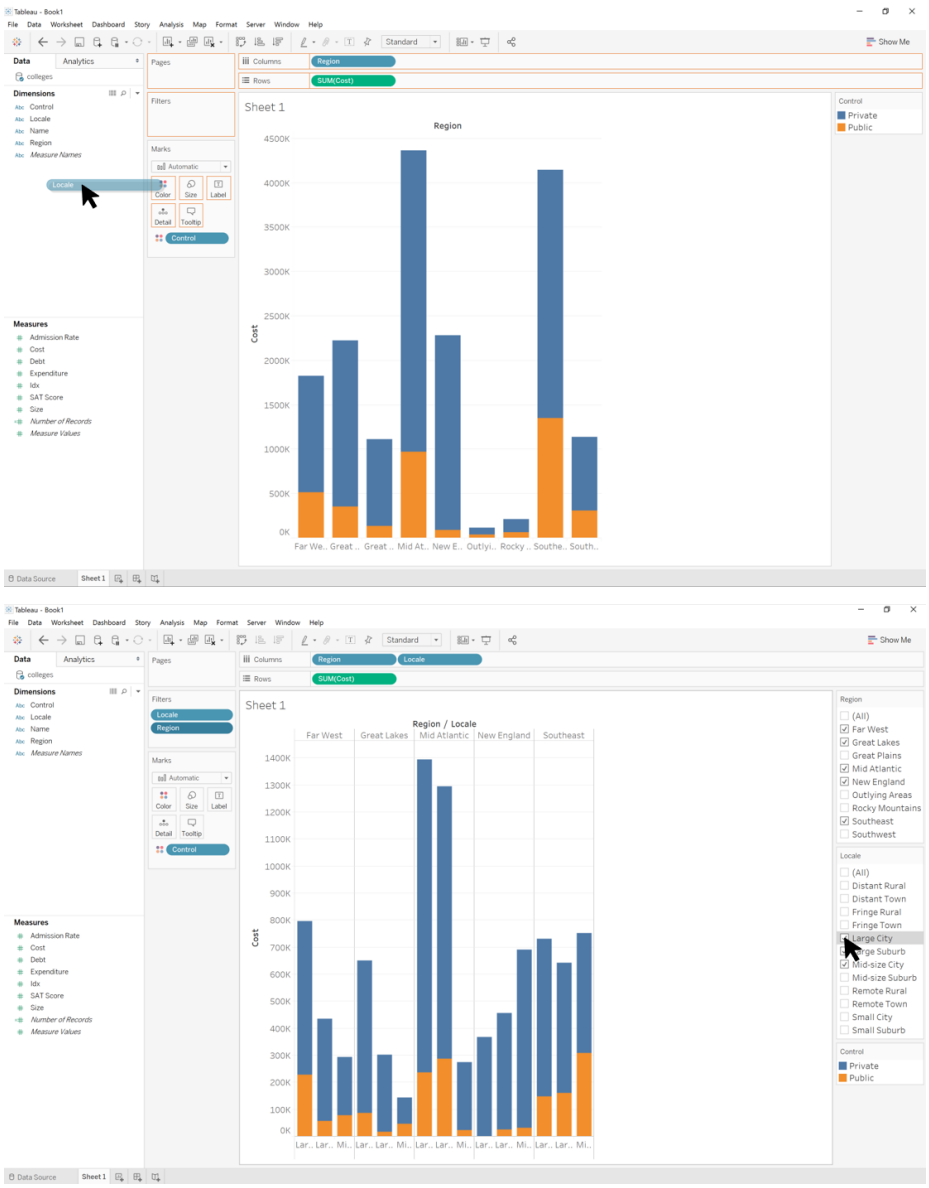



Figure 1.1: Example interactions with a popular WIMP-based visualization interface, Tableau [194]. (Top) As an attribute is dragged from the left panel, the system highlights "shelves" in the interface that it can be dropped into (e.g., rows, columns, filters, color). (Bottom) Data filters are applied on the main view using the filter panel on the right.

vide details-on-demand. In this regard, it is imperative to explore alternative *post-WIMP interfaces* [200] and interactions that are more suited for contemporary devices. But more so than 'porting' existing visualization tools to new settings, advancements in hardware and input recognition technology present us with an opportunity to fundamentally reimagine human-data interaction experiences and design visualization tools that "go beyond the

mouse and keyboard" [107].

One alternative style of interaction that can address the above challenges with DM is *natural language* (NL). NL, once only considered a depiction of futuristic HCI in science-fiction media, has today become a household reality in the form of voice-based systems like Amazon Alexa and Google Home. NL can be a powerful addition to visualization tools as it allows people to directly express their data-related questions without having to learn to use the interface or translate their questions into system actions. Furthermore, given its "invisible" nature, NL can seamlessly complement other modalities to support interaction with visualization tools in post-WIMP settings.

Similar to any NL-based system, however, supporting NL interaction with data visualization tools present several challenges from both a system implementation and usability standpoint. For instance, given an NL query, not only do visualization systems need to identify the various parameters (e.g., attribute names, visualization type) but also infer user intent (e.g., analytic operations such as correlation or filtering). This can be difficult at various levels depending on the level of ambiguity and underspecification in the query. Furthermore, given the free-form nature of NL input, systems need to incorporate ample feedback and discoverability mechanisms to make users aware of the system's actions and what the possible interactions are. Without these, users may feel lost about the possible space of interactions, resulting in more system errors that can ultimately discourage them from interacting with the tool.

Given their complementary strengths and weaknesses, NL and DM together possess the ability to support a more expressive, intuitive, and integrated interaction experience [36]. For instance, while NL input suffers from issues such as ambiguity and high cost of errors, DM allows people to be precise and often facilitates easier error recovery. Conversely, as highlighted earlier, DM interaction may require repeated, tedious actions or require users to be well acquainted with an interface. In such cases, NL input can be advantageous and can allow people to perform multiple steps with a single command or simply state their

Figure 1.2: Scene from the movie “Iron Man 2” illustrating multimodal gesture- and voice-based interactions with a visualization.

intended tasks without having to translate their objectives into system actions. This complementary nature of NL and DM interaction makes them a powerful input combination and has been explored in numerous examples across different application domains including graphics and media editing [71, 103, 147, 184], video creation [30, 192], and document reviewing [223, 224], among others.

Given the prominent use of DM in existing visualization tools and the promise of NL input, the idea of having multimodal systems that combine the two styles of interaction for data analysis with visualizations is an appealing one, in theory. In fact, many visionary sequences from science-fiction movies such as the one shown in Figure 1.2 also illustrate examples of multimodal visualization interfaces where NL is combined with other modalities like touch or gestures to let people freely interact with and analyze data. However, in practice, several open questions pertaining to the design, usability, and utility of such systems remain. First, from a usability and utility standpoint, how well do these systems support common visual data analysis and exploration tasks? Do users of visualization tools actually prefer them over the more familiar unimodal interfaces, and if so, why? From an input standpoint, are there specific visual analytic tasks or operations that align well to a specific

modality? Does one modality prevail as the primary mode of input constraining the types of interactions that can be designed? Do NL- and DM-based multimodal systems enable new workflows or interaction experiences that are not supported by current tools? Lastly, from a system development perspective, what are interface design and implementation challenges in creating such multimodal tools? Given the flexibility in choosing both input modalities and interaction patterns, how should systems effectively capture and interpret user input? How can the knowledge inferred through different modalities be consistently propagated across the interface and combined to support seamless integration? Clearly, these questions cannot be answered solely by conducting a few experiments. Instead, to address these questions, we need to *conceptualize, build, and evaluate visualization systems incorporating both NL and DM interactions*.

With this goal in mind, my research investigates the design space of NL- and DM-based visualization systems, highlighting the challenges and benefits of designing and implementing such multimodal interfaces. More specifically, this dissertation describes a series of research projects involving: i) literature reviews and formative studies, ii) design and implementation of prototype systems, iii) user evaluations of multimodal visualization interfaces, and iv) creation of tools to aid visualization system developers incorporate NL interaction in their own work. The results and insights from this dissertation can inform the design of future data analysis and visualization systems that support NL-only input or multimodal input incorporating NL. More broadly, as data visualizations become popular in settings that inherently support multiple input modalities (e.g., touchscreens, AR/VR environments), this research paves the way for the next generation of post-WIMP visualization interfaces that amplify human perceptual, cognitive, and manipulative abilities.

## 1.1 Thesis Statement & Research Goals

The overarching goal of my work is captured by the following thesis statement:

> ***Combining natural language and direct manipulation leads to the design of novel multimodal visualization interfaces that promote fluid and expressive human-data interaction experiences.***

To clarify the scope of this thesis: in this dissertation, I adapt Oviatt's definition [145] and define *multimodal interfaces* as systems that process combined user input modes—specifically, NL (spoken or typed) and DM (though pen, touch, mouse)—in a coordinated manner with multimedia system output (new visualizations or changes to an existing visualization). Furthermore, I follow Elmqvist et al.'s [52] definition of *fluid interaction* and consider fluid interfaces as ones that promote a state of flow and maximize direct interactions with the data view (as opposed to interface controls that propagate changes to the view). Finally, by *expressive*, I imply the concept of *freedom of expression* as used by Lee et al. [107] and consider an interface expressive if it gives people the flexibility to specify their intent through varying modes of interaction.

To validate the above thesis, I break it down into four more specific research goals (**RG1-4**). **RG1-3** focus on conceptualizing, implementing, and evaluating NL- and DM-based visualization interfaces whereas **RG4** focuses on tools to aid the implementation of future data visualization systems supporting NL interaction. I discuss these goals in more detail below:

**[RG1]** *Inspect and Characterize the potential role of NL in visualization tools.*

Given the nascency of NL interactions with visualizations, I first identify user goals (e.g., specifying charts, interacting with an active chart, formatting charts) that are well suited for NL input in the context of a visualization system. To identify these goals, I review prior work on NL interfaces (NLIs) for databases, research on NLIs in the broader HCI community, and early examples of visualization tools demonstrating NL interaction. I

summarize this review as a goal-driven framework and discuss how the gaps and opportunities identified through this review motivated the subsequent projects of this dissertation.

**[RG2]** *Design NL- and DM-based multimodal interfaces to support fluid and expressive interaction experiences for canonical visual analysis scenarios.*

To test the feasibility of using multimodal interactions for visual analysis, I investigate interfaces that combine speech with touch and/or pen input. In my exploration of this interface and interaction design space, I cover a breadth of visualizations (e.g., bar charts, line charts, node-link diagrams) and operations (e.g., sorting, filtering, finding paths and connections) that are commonly supported in current visualization tools. To illustrate the proposed interactions, I embody them via two multimodal visualization interfaces: ORKO and INCHORUS. Based on user evaluations of these systems, I show that multimodal interfaces not only support a breadth of visual analysis operations prevalent in current tools but do so in a fluid and expressive manner. I also share preliminary user feedback on multimodal visualization interfaces, user preferences for different modalities and operations, and the observed variations in user interaction patterns. Finally, I translate the system design challenges and knowledge gained from the user studies into multimodal interaction design concepts and principles to assist future research and development.

**[RG3]** *Illustrate the expressive potential of NL- and DM-based multimodal interaction by enabling novel, free-form data exploration workflows.*

Going beyond replicating capabilities of current tools, I explore innovative human-data interaction experiences that multimodal visualization interfaces can enable. Specifically, I investigate how pen, touch, and speech can collectively support the idea of "free-form data exploration" where people smoothly transition between systematically-bound views (e.g., scatterplots, unit column charts) and customized views reflecting their mental model of a data space. I depict this idea through a prototype system, DATABREEZE, that interweaves multimodal interaction with flexible unit visualizations that allow people to freely interact

with data items and customize the view based on their subjective preferences. Following an iterative design process, I show how such a multimodal interface can support a wide range of operations including specifying both standard and custom spatial layouts, formatting data points, and freely annotating the view, among others. To summarize, through DATABREEZE's design and evaluation, I illustrate how multimodal visualization interfaces can transcend capabilities offered by current visualization tools and enable a novel, free-form style of visual data exploration and sensemaking.

**[RG4]** *Provide mechanisms to aid visualization system developers prototype NL interactions.*

Through **RG1-3**, I illustrate how NL input provides flexibility in interacting with visualizations and posing data-related queries. However, inherent characteristics of NL input such as ambiguity and underspecification make implementing NL interpreters for data visualization systems a challenging task. To address this challenge, I leverage my experience of implementing NL-based multimodal systems and learnings from related work on NLIs for data visualization to develop a general-purpose toolkit, NL4DV. I detail NL4DV's design goals and describe how the toolkit infers data attributes/values, low-level analytic tasks, and visualization specifications from NL queries. I discuss how NL4DV formalizes the inferred information into a JSON-based analytic specification that can be programmatically parsed by visualization developers having little or no experience with NLP tools and techniques. Finally, through example applications, I showcase how this analytic specification helps developers prototype NL interactions with visualizations in three scenarios: 1) incorporating NL input into an existing DM-based visualization system, 2) implementing new NLIs for visualization, and 3) supporting NL-based visualization specification in data science programming environments.

## 1.2 Contributions

The key contributions of this dissertation include:

- A framework of NL utterances based on user goals in the context of visualization tools.
- The design and implementation of two prototype multimodal visualization interfaces: ORKO and INCHORUS, that combine speech with pen and/or touch to support fluid and expressive interactions for visual analysis with canonical chart types (e.g., line charts, bar charts, node-link diagrams).
- Characterizations of multimodal user input and interaction patterns within visualization tools based on evaluations of ORKO and INCHORUS.
- Design guidelines and concepts that can be used to develop and compare multimodal interactions with visualization tools supporting NL and DM.
- An approach to enable free-form data exploration by interweaving multimodal interaction with flexible unit visualizations, along with an operationalization of this approach through a prototype system, DATABREEZE.
- NL4DV, a toolkit that helps developers prototype NL-based visualization systems.

## 1.3 Prior Publications and Authorship

The content of this dissertation is, in part, based on manuscripts previously published at different venues (associated publications are listed in Table 1.1). Although I am the principal author of the described research, this dissertation is the result of a close collaboration with my advisor, John Stasko, as well as my mentors and colleagues at Georgia Tech and Microsoft Research. In particular, Ayshwarya Saktheeswaran (former Master's student at Georgia Tech) was instrumental in the design, execution, and analysis of ORKO's follow-up

Table 1.1: Dissertation outline and publication summary.

| **Research Goal** | **Chapter(s)** | **Associated Publication(s)** |
|---|---|---|
| [RG1]<br>Inspect and Characterize the potential role of NL in visualization tools | Chapter 3 | **A. Srinivasan** and J. Stasko, "*Natural Language Interfaces for Data Analysis with Visualization: Considering What Has and Could Be Asked.*" Proceedings of EuroVis (Short Papers), 2017. |
| [RG2]<br>Design NL- and DM-based multimodal interfaces to support fluid and expressive interaction experiences for canonical visual analysis scenarios | Chapters 4, 5 | **A. Srinivasan** and J. Stasko. "*Orko: Facilitating Multimodal Interaction for Visual Exploration and Analysis of Networks.*" IEEE TVCG, 2018.<br><br>A. Saktheeswaran, **A. Srinivasan**, and J. Stasko. "*Touch? Talk? or Touch and Talk? Investigating Multimodal Interaction for Visual Network Exploration and Analysis.*" IEEE TVCG, 2020.<br><br>**A. Srinivasan**, B. Lee, N.H. Riche, S.M. Drucker, and K. Hinckley. "*InChorus: Designing Consistent Multimodal Interactions for Data Visualization on Tablet Devices.*" ACM CHI, 2020. |
| [RG3]<br>Illustrate the expressive potential of NL- and DM-based multimodal interaction by enabling novel, free-form data exploration workflows | Chapter 6 | **A. Srinivasan**, B. Lee, and J. Stasko. "*Interweaving Multimodal Interaction with Flexible Unit Visualizations for Data Exploration.*" IEEE TVCG, 2020. |
| [RG4]<br>Provide mechanisms to aid visualization system developers prototype NL interactions | Chapter 7 | A. Narechania*, **A. Srinivasan***, and J. Stasko. "*NL4DV: A Toolkit for Generating Analytic Specifications for Data Visualization from Natural Language Queries.*" IEEE TVCG, 2021 (To appear). [*Equal Contribution] |

study [166] detailed in Chapter 4. Arpit Narechania (PhD student at Georgia Tech) and I co-authored the design and development of the NL4DV toolkit described in Chapter 7.

To reflect my collaborators' contributions, I use the pronoun 'we' when describing the conducted research in Chapters 3-7.

# CHAPTER 2

# RELATED WORK

My research is situated at the intersection of HCI and Information Visualization (InfoVis). More specifically, this dissertation investigates novel multimodal interfaces that enable people to fluidly explore data through visualization systems using both NL and DM. With that in mind, in this chapter, I review and summarize prior work pertaining to four related themes: 1) the importance of interaction in InfoVis, 2) post-WIMP interfaces in visualization, 3) NL- and DM-based multimodal interfaces in HCI, and 4) NLIs for data visualization, discussing how my research relates and contributes to the existing literature.

## 2.1 Interaction in Information Visualization Tools

*Representation* and *interaction* are considered as the two primary components of InfoVis [27, 149, 182, 221]. The role of interaction has been well explored and there exist several categorizations and taxonomies of interactions in the context of visualization tools. For instance, Shneiderman [179] discusses overview, zoom, filter, details-on-demand, relate, history, and extract as seven types of interactive data tasks. Buja et al. [25] list focusing, linking, and arranging views as higher-order categories of interaction with visualization tools. Chuah and Roth [31] highlight encoding data, setting graphical values, and manipulating objects as basic visualization interactions. Dix and Ellis [47] emphasize that simple representations can be made more powerful through interactions, listing highlighting/focus, accessing extra information, overview and context, changing representation parameters, changing representations for the same data, and linking representations as exemplary interaction categories. Although these and other similar studies and taxonomies (e.g., [90, 210]) describe categories of interaction techniques, given the inherent subjectivity of the term, there is still no single agreed upon definition of interaction within visual-

ization tools. To this end, Dimaria and Perin [46] reviewed existing interaction taxonomies and techniques and broadly summarized interaction for data visualization as:

> *Interaction for visualization is the interplay between a person and a data interface involving a data-related intent, at least one action from the person and an interface reaction that is perceived as such.*

In terms of this definition, the categories of interaction techniques discussed above describe the 'actions' that people take when working with visualization tools. On the other hand, researchers have also categorized analytic tasks and intents. For example, Roth and Mattis [157] highlight compare, correlate, accurate value, finding distributions etc., as information-seeking goals and discuss the importance of considering these goals in choosing appropriate visualizations during data analysis. Zhou and Feiner [229] list elaborate, summarize, explore, and compute as examples of visualization intents along with lower-level tasks such as compare, locate, rank etc., as means to accomplish the intents. Amar et al. [3] also present a list of 10 low-level analytic tasks that people perform when analyzing data: retrieve value, filter, compute derived value, find extremum, sort, determine range, characterize distribution, find anomalies, cluster, and correlate. Besides these general data analysis tasks, categories of analysis tasks for specific dataset and visualization types such as networks [109], geographic visualizations [156], spatiotemporal data [7] etc., also exist.

The aforementioned taxonomies and categorizations focus on lower-level tasks/goals or interaction techniques in isolation. However, researchers have also suggested bridging these two efforts and argued that taxonomies which focus strongly on interaction techniques are relatively system-centric, whereas those which focus on user goals do not specifically examine interaction. For instance, Yi et al. [221] discuss how the intent of a person when interacting with a visualization is key, and propose a list of seven intent-focused categories of interaction in information visualization (select, explore, reconfigure, encode, abstract/elaborate, filter, connect) and discuss interaction techniques within each category (e.g., highlighting or adding labels are techniques to enable selections). Heer and Shneiderman [73]

propose a taxonomy of 12 task types grouped into three higher-level categories: data & view specifications, view manipulation, and process & providence, and illustrate interaction techniques that can be used to support these tasks. Brehmer and Munzner [21] present a multi-level typology distinguishing *why* and *how* a visualization task is performed, as well as *what* the task inputs and outputs are.

These taxonomies (in particular, ones pertaining to user interaction intent and low-level tasks) are relevant to my research as they help in identifying operations and functionalities that need to be supported in visualization tools. Specifically, as we explore new input modalities, these taxonomies serve as a valuable starting point to design interfaces and interaction techniques that accommodate different user goals. Furthermore, especially for NL input, the low-level task taxonomies (e.g., [3, 157]) also provide a concrete list of 'data-related intents' (e.g., finding correlations, computing derived values) that systems can be configured to extract. By identifying user intent, systems can, in turn, offer more assistance and generate more relevant responses to user queries/utterances.

## 2.2 Post-WIMP Visualization Interfaces

Van Dam [200] defines post-WIMP interfaces as interfaces "*containing at least one interaction technique not dependent on classical 2D widgets such as menus and icons*." With the growing popularity of interactive displays (e.g., tablets, large touchscreens) and AR/VR devices, over the past decade, there has been a significant push in visualization research to transition from WIMP to post-WIMP interfaces tools that enable human-data interaction in these contemporary settings. For instance, while discussing the idea of a "science of interaction," Pike et al. [149] identify the creation of ubiquitous, embodied interaction on devices ranging from very small (e.g., mobile, handheld computers) to very large displays as a future research challenge. Elmqvist et al. [52] explore the idea of fluid interfaces for information visualization and discuss how post-WIMP systems can enable a "fluid interaction" experience. Lee et al. [107] also advocate for moving past mouse and keyboard

interactions for information visualization. By considering four design dimensions including the individual, the technology, social aspects of interactions between people, and the interspace between a person and the technology, Lee et al. highlight "providing a high freedom of expression" through post-WIMP interaction as a key opportunity for research within the InfoVis community. Along similar lines, Roberts et al. [154] also argue that we are at a cusp in visualization research where we need to develop for and adapt to today's new devices and tomorrows technology, stating that "The '*next big thing*' is multi-sensory visualization that goes beyond the desktop."

Motivated by these viewpoints, there have been a number of systems and studies exploring how people interact with visualizations in post-WIMP settings such as mobile/tablets (e.g., [50, 101, 105, 159, 161]), large interactive displays (e.g., [2, 82, 102, 108, 110, 226]), and AR/VR environments (e.g., [40, 41, 48, 79, 146, 207]) using a variety of input modalities including touch, pen, gestures, and smell, among others [12]. My work is also motivated by the research opportunities arising from exploring natural user interfaces and input modalities for data visualization. Specifically, enabling a *fluid interaction* experience [52] and providing a higher degree of *freedom of expression* [107] during human-data interaction through visualizations are two key themes prevalent throughout the work presented in this dissertation. Furthermore, by considering speech as a new interaction modality for visualization tools, my research expands the input and interaction design space considered by prior work and contributes novel post-WIMP visualization interfaces that combine speech (form of NL) with pen and/or touch (forms of DM).

As stated above, the systems described in this dissertation focus on pen and touch as the primary input modalities for DM interactions. Given this focus, most relevant to this research is prior work investigating visualization systems that are designed for touchscreens operated by a single user (as opposed to a collaborative system). Early examples of this include work by Buering et al. [24] comparing different interaction techniques to support zooming a scatterplot on a PDA using a stylus. Schmidt et al. [170] developed a set

of multi-touch gestures (TouchPlucking, TouchPinning, TouchStrumming, TouchBundling and PushLens) to support link selection and manipulation in node-link diagrams. Tangraphe [196] supported single hand, multi-touch gestures to support common network visualization interactions such as selecting nodes and edges, expanding/collapsing connections, and updating the network layout. SketchVis [23] allowed users to sketch visualizations on a whiteboard with a pen. With SketchVis, users could perform gestures or draw parts of a visualization (e.g., axes, ticks), and the system populated the view accordingly. TouchWave [15] showcased how multi-touch gestures could be used to interact with hierarchical stacked graphs on a tablet, enabling a breadth of operations including retrieving values, extracting layers, sorting, and re-scaling layers, among others. Drucker et al. [50] compared two variants of a tablet-based visualization tool that supported interacting with bar charts: one that supported direct touch interactions with the view and another that mimicked a WIMP-style control panel. They found that people were faster and more efficient with the direct interface while also subjectively preferring the same. Another tablet-based visualization system, Kinetica [158] illustrated how multi-touch gestures coupled with physics-based affordances applied to a scatterplot can enable a fluid and enjoyable data exploration experience. Sadana and Stasko presented a series of tablet-based visualization systems focusing on interactions with individual visualizations such as scatterplots [159], interactions in the context of multiple coordinated views [161], and advanced selection techniques like generalized selection [160]. In doing so, they demonstrated how the interface and interactions in tablet-based visualization tools can be designed to effectively support analytical capabilities offered in desktop systems, highlighting the corresponding design considerations and challenges.

Besides the above systems that focus on unimodal pen or touch input, there has also been a series of research exploring multimodal interfaces combining pen and touch. For instance, Frisch et al. [57] investigated how people use pen and touch to support both structural editing and freehand sketching to edit node-link diagrams. Walny et al. [208]

investigated the use of both pen and touch for data exploration on whiteboards. Their study explored the interaction patterns that people employ during visual data analysis, in particular the interplay between pen and touch interactions. SketchStory [108] demonstrated how free-form sketching with a pen coupled with simple touch interactions can be leveraged to create engaging data-driven presentations (Figure 2.1). PanoramicData [226] and SketchInsight [110] showed how a combination of pen and touch can enable more naturalistic data exploration on an infinite canvas. WordlePlus [84] illustrated how pen and touch together can give users higher control over wordles by allowing people to manipulate word size, placement, and groupings while incorporating additional visuals and animations. TouchPivot [83] combined pen and touch interactions with WIMP-style interface elements and visualization recommendation to help novices conduct data exploration on tablets. DataInk [217] and DataToon [94] offered pen and touch interfaces to create compelling infographics and data comics, respectively. These systems illustrate how the added expressivity afforded by multimodal pen and touch interactions not only help design better user experiences for post-WIMP systems but also augment users' creativity. Besides data exploration and data-driven communication, recent examples have also shown how pen and


Figure 2.1: An illustration of multimodal pen- and touch-interactions in SketchStory [108] being used to create a data-driven presentation.

touch interactions can aid sensemaking and improve analytic provenance. VoyagerInk [95], for instance, illustrated how pen and touch can support improved note-taking during visual data exploration to help users better organize and track their findings. ActiveInk [155] was another example of a pen- and touch-based visualization tool that supported seamless switching between data exploration and externalization to facilitate sensemaking.

The aforementioned research on pen and touch interactions collectively illustrates the value in supporting direct interactions with visualizations and sheds light on design challenges such as occlusion, having a limited vocabulary of gestures, and supporting consistency in interactions across visualizations, among others. In my work, I leverage the knowledge gained from prior work to follow best practices when designing pen- and/or touch-based DM interactions. Incorporating speech as a third modality with pen/touch, I also illustrate how DM- and NL-based multimodal input helps overcome challenges like gesture overloading and maintaining interaction consistency across different chart types that are common in pen/touch-only systems.

## 2.3 NL and DM-based Multimodal Interfaces in HCI

Multimodal interfaces that support both NL and DM interactions have for long been a topic of interest within HCI research. Building upon the tight coupling of gesticulation and speech [91], one of the earliest class of multimodal interfaces include systems that combine voice with mid-air gestures (or hand movements). Possibly the first, and one of the most popular multimodal systems was Bolt's "Put-that-there" [19] that appeared four decades ago in 1980. The system allowed users to command shapes on a large display using a combination of gestural and voice commands (Figure 2.2). A few years later, Hauptmann [71] studied multimodal interaction in the context of graphics manipulation. He found that people strongly prefer to use both gestures and speech for graphics manipulation, intuitively using multiple hands and fingers in all three dimensions. With the Rubber Rocks [34] virtual world, Codella et al. coupled multimodal speech and gestural input with multimodal

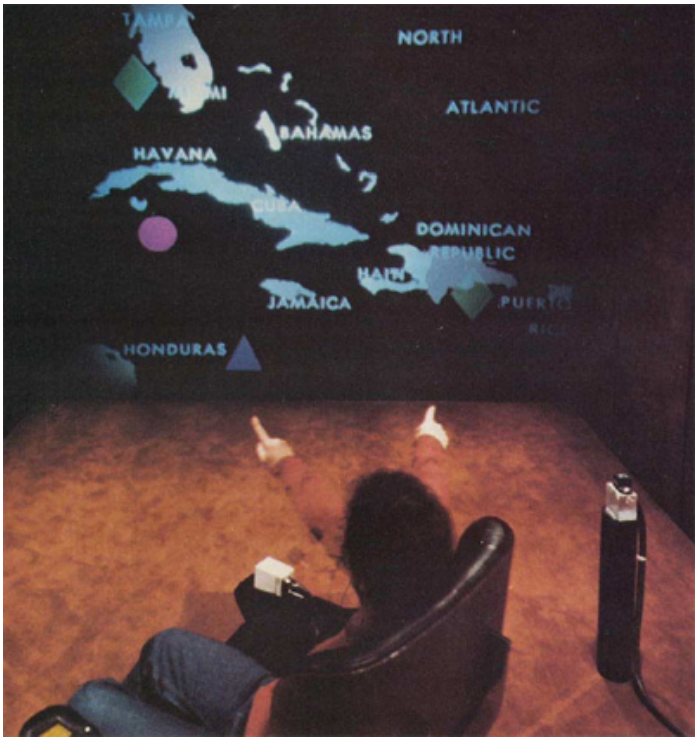



Figure 2.2: Richard Bolt's "Put-that-there" [19] sequence illustrating a multimodal voice- and gesture-based interface for manipulating shapes on a projected wall display.

output including speech, audio, and motion parallax effects. They discussed how having multiple modes of interaction promoted an immersive experience and highlight that the expressiveness afforded by multimodal input allowed different users in a collaborative setting to perform similar operations but through their individually preferred modalities. Koons et al. [98] explored the combination of gestures and speech along with gaze to further improve deictic referencing, illustrating the flexibility afforded by such multimodal input in the context of 2D map manipulation and interacting with a 3D blocks world. As an extension of the "Finger-Pointer" technology [60], Fukumoto et al. demonstrated how hand pointing gestures and voice can be combined to operate devices like VCRs or projected drawing interfaces by temporally synchronizing the input events from the different modalities. More recent systems such as DemoDraw [30] also illustrate how voice and gestures can be combined to author physical demonstrations useful for creating instructional videos.

Given the widespread use of DM interactions across desktop applications, many re-


Figure 2.3: The QuickSet system running on a handheld PC that supports multimodal interaction via pen and speech. Image shows a screenshot of the system from [37].

searchers have also explored the combination of voice and mouse-based DM interaction. For instance, comparing a keyboard and mouse interface to a multimodal interface including voice input, Martin [123] found that VLSI circuit designers were able to complete 24% more tasks when spoken commands were available. Salisbury et al. [167] developed a multimodal interface that combined mouse and speech to view radar data and issue commands to an aircraft. Pausch and Leatherby [147] conducted an empirical study by adding voice input to a mouse-controlled graphics editor. They found that the combination of modalities sped up participant task performance times by up to 56% with an average reduction of over 21%. Gourdo et al. [65] presented Voice-Paint and Notebook, two multimodal interfaces that combined speech with mouse and keyboard input. Besides describing the system capabilities and interactions, they also laid out a design space of multimodal input integration ranging from exclusive (modalities operate in isolation) to synergic (modalities collectively specify operations).

Another popular combination of modalities that has been investigated in prior research is that of speech with pen and/or touch. Early examples included work by Vo and Waibel [206] demonstrating how pen and speech can be combined in the context of text editors. Cheyer

and Julia [29] presented an agent-based approach to process and combine input from different modalities, illustrating their architecture through a pen and speech based prototype map application for travel planning. A series of systems and studies from Oviatt, Cohen et al. [35, 37, 38, 39, 138, 139, 140, 141, 143] also investigated this combination of modalities. One of the most popular systems among these was the pen- and speech-based mapping application, QuickSet [37, 137] (Figure 2.3) that was used during training simulations for the US Marine Corps.

Beyond its domain-specific application, QuickSet was also instrumental in conceptualizing and furthering the space of multimodal interfaces through its formative studies (e.g., [137, 141, 143]) and extensions to tangible and collaborative scenarios (e.g., [37, 126, 127]). For instance, an initial motivating hypothesis for multimodal interaction was that it is significantly faster than its unimodal counterparts. This effect was not always observed, however. In studies comparing the use of speech and pen input individually to a combination of both, Oviatt et al. [137, 143] showed that multimodal interaction only sped up task completion by small amounts (10%). Conversely, multimodal interaction significantly improved error handling and reliability: people made 36% fewer errors with a multimodal interface than with a unimodal interface. Although this was just one specific study, it illustrates that our intuition about such interfaces may not always be correct [144]. In another study, Oviatt et al. [141] leveraged QuickSet to investigate user interaction patterns in pen- and speech-based multimodal interfaces. They discuss integration patterns including how people frequently used modalities in isolation (but switched between modalities for different operations) and when they combined them, depending on the intended operation, the combination was performed sequentially (i.e., with a time lag) or simultaneously. The authors also highlight how the phrasing of speech commands issued multimodally was different and often shorter than commands during speech-only interactions. Augmenting paper-based interfaces with pen-, touch-, and speech-based multimodal interactions, McGee et al. [126, 127, 128] extended the learning from QuickSet's studies and architec-


Figure 2.4: An example of multimodal interaction in PixelTone [103] demonstrating the use of a deictic command followed by a speech command to edit a specific portion of an image.

ture to develop tangible interfaces for collaborative planning tasks in military command posts.

In more recent examples, PixelTone [103] offered a touch- and speech-based multimodal interface for image editing (Figure 2.4). PixelTone highlighted how deictic commands simplify otherwise tedious and complex tasks of adding filters to specific portions of an image or adding gradient effects along a specific direction. MozArt [174] allowed users to leverage speech and touch interactions to perform 3D modeling. A comparative study of MozArt against a multitouch system also revealed that people preferred the multimodal interface over a touch-only interface. TalkingDraw [218] presented a pen, touch, and speech interface for basic diagram editing, investigating techniques to support fluid triggering of voice input during multimodal interactions. RichReview [223, 224] demonstrated how complementing basic touch interactions (tap, zoom/pan) with a rich suite of pen and speech interactions to add and edit annotations can facilitate document reviewing.

The systems and studies covered in this section are not an exhaustive list and only cover commonly referenced examples and systems that are most relevant to this dissertation (specifically, pen-, touch-, and speech-based multimodal systems). A more comprehensive survey and review of prior work can be found in other manuscripts such as [51, 124, 138, 142, 145, 175].

The wealth of examples described above highlight that NL- and DM-based multimodal

interfaces have been a long standing idea and have constantly evolved with advances in hardware and input recognition technology. While the broader HCI community has probed this design space from different perspectives and investigated a breadth of application domains (e.g., military simulations, photo editing, graphics manipulation), few, if any, examples within the visualization literature explore such naturalistic voice- and DM-based multimodal interfaces. My research bridges this gap and investigates multimodal visualization interfaces that are similar, in spirit, to the aforementioned systems. In doing so, my work contributes novel visualization tools and creates new interface and interaction design opportunities for visualization research. On the other hand, by specifically investigating multimodal interaction for data visualization, this dissertation also contributes to general multimodal interface research by extending ideas from prior work to a new domain.

## 2.4 NLIs for Data Visualization

In 2001, Cox et al. [42] presented an initial prototype of a NLI for data visualization. With their system, to produce a visualization, users had to specify a well-structured query or a set of partial queries while engaging in an actual dialogue with the system. This was followed by a gap of almost a decade before the appearance of a second NLI for data visualization. In 2010, with the Articulate system, Sun et al. [193] presented a model to create visualizations from NL queries by deriving mappings between tasks and data attributes in user queries (Figure 2.5). In more recent years, however, given the advances in NL understanding technology and NLIs for databases (e.g., [18, 150, 228]), there has been a surge in research on NLIs for data visualization [44, 55, 62, 77, 89, 93, 99, 100, 172, 173, 198, 225].

DataTone [62] uses a combination of lexical, constituency, and dependency parsing to let people specify visualizations through NL. Furthermore, detecting ambiguities in the input query, DataTone leverages mixed-initiative interaction to resolve these ambiguities through GUI widgets such as dropdown menus. FlowSense [225] uses semantic parsing

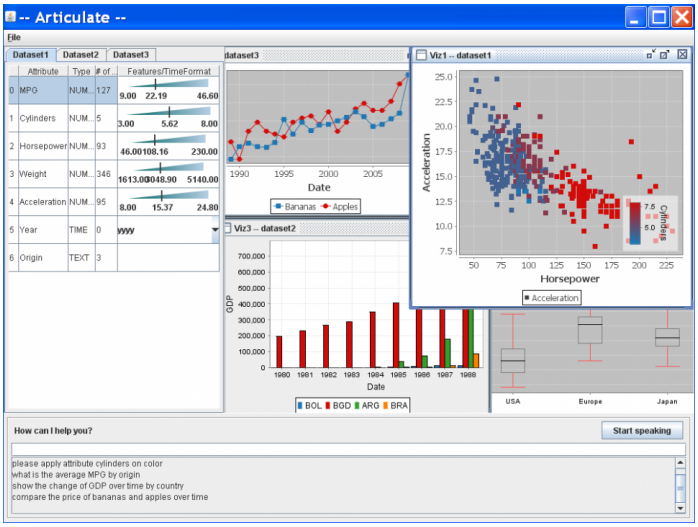



Figure 2.5: Articulate's [193] interface displaying NL queries (at the bottom) and visualizations corresponding to those queries (top-right).

techniques to support NL interaction within a dataflow system, allowing people to specify and connect components without learning the intricacies of operating a dataflow system. Valletto [89] also lets people specify visualizations through NL and models this interaction within a chat-like interface.

While the above systems primarily focus on helping people specify or create visualizations through standalone NL queries, another series of systems place emphasis on supporting a richer dialog between users and visualizations. For instance, Eviza [172] incorporates a probabilistic grammar-based approach and a finite state machine to allow people to interact with an active visualization through a series of NL commands. Extending Eviza's capabilities and incorporating additional pragmatics concepts, Evizeon [77] allows both specifying and interacting with visualizations through standalone and follow-up utterances (Figure 2.6). Articulate2 [99] also presents a dialogue based system to let people specify a visualization and then drill down using a sequence of NL questions. Although their core goal is to support computational data science tasks, systems like Ava [85] and Iris [55] also allow specifying visualizations in the context of a dialog about the underlying machine learning model or statistical tests.

Outside research prototypes, NLIs for data visualization are also becoming popular as

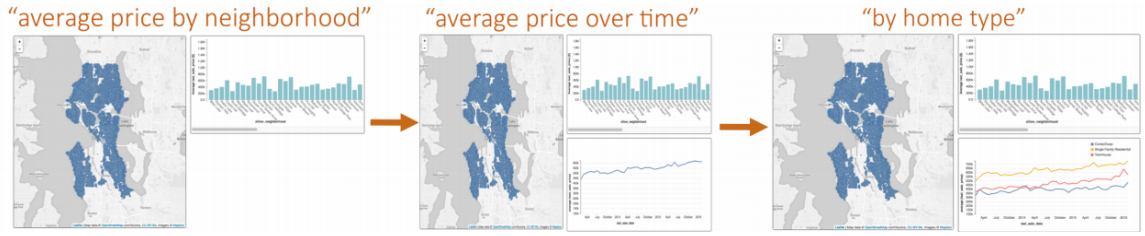



Figure 2.6: An example sequence of queries illustrating conversational interaction during visual analysis with Evizeon [77].

commercial software [81, 129, 195, 197]. Microsoft's Power BI, for instance, provides the Q&A feature that lets people specify visualizations through NL (Figure 2.7). Tableau also offers an NL service, Ask Data (Figure 2.8), that was developed based on learnings from implementing and evaluating Eviza [172] and Evizeon [77], and a wizard-of-oz study understanding people's expectations when interacting with Tableau through NL [198]. Besides offering pragmatics capabilities similar to the research prototypes [77, 172], Ask Data also incorporates intelligent inference techniques and algorithms to support underspecified queries (e.g., queries that do not include data aggregations or minimum number of attributes required to create a specific type of chart) [173].

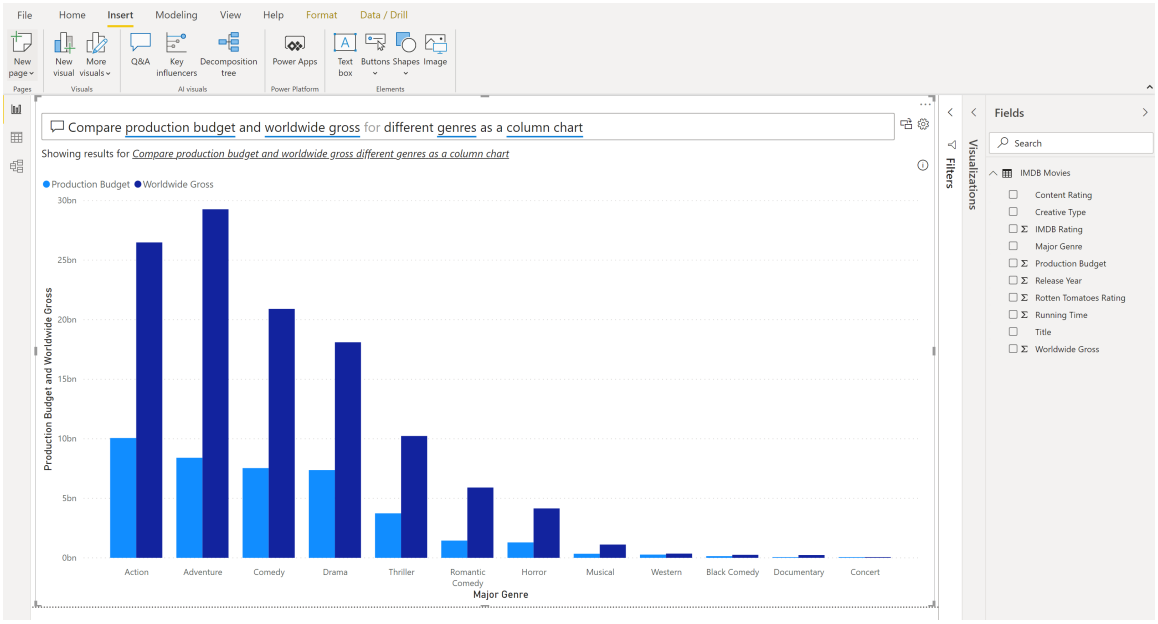



Figure 2.7: An example of NL interaction in Microsoft's Power BI Q&A.

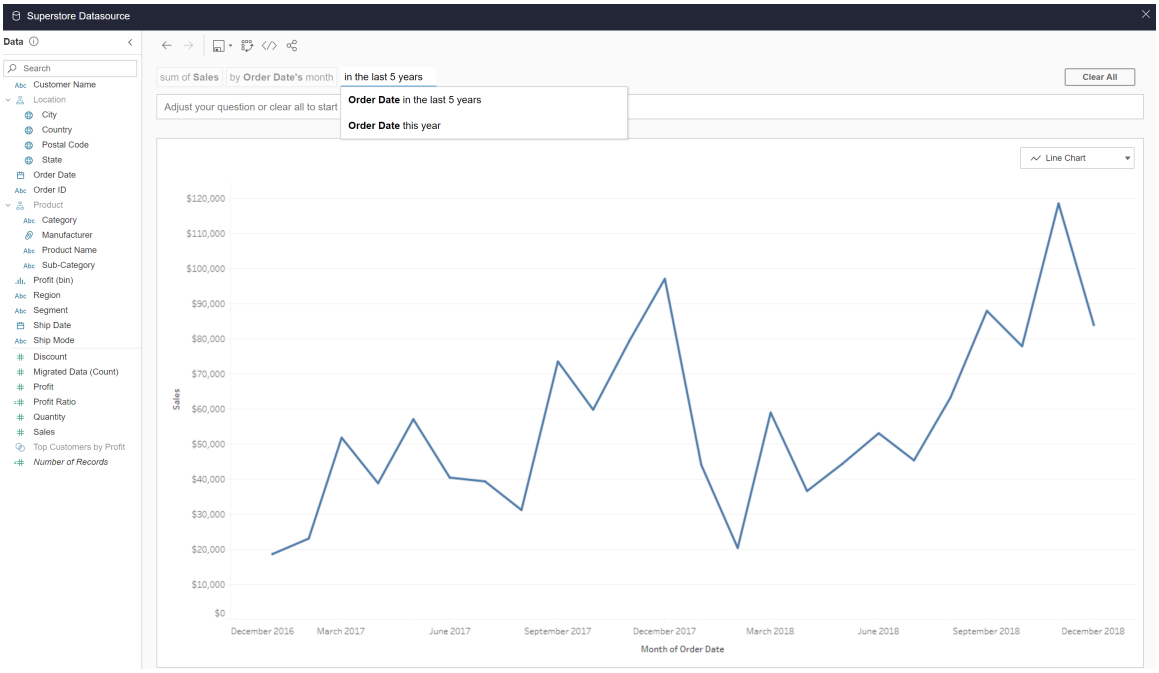



Figure 2.8: An example of NL interaction in Tableau's Ask Data.

In addition to the aforementioned systems that focus on specifying visualizations and exploring data, there are other systems that investigate the use of NL in different contexts such as data-driven communication and question answering with visualizations. For example, Text-to-Viz [44] offers an interface to convert proportion-related NL statements (e.g., "*More than 20% of smartphone users are social network users*") into stylized infographics. Vis-Annotator [100] lets people enter textual descriptions about a given visualization (e.g., "*The large point above 250 represents China.*") and annotates the visualization based on these descriptions. Kim et al. [93] present a model that takes as input NL questions about a given visualization (e.g., "*What country has the shortest orange bar?*") and answers those questions along with an explanation of the steps performed to get to the response (e.g., "*China. I looked up 'Country' of the shortest orange bar.*").

Collectively, the existing systems and studies highlight the potential of NL interaction with visualization tools for both data exploration and communicating data-driven findings. These efforts reaffirm the motivation for this dissertation's research and also serve as important references for implementing NL interpreters (e.g., by suggesting metrics to map

queries to dataset entities, parsing rules to interpret queries). While I leverage what prior work has taught us, my research also contributes to the design space of NL-based visualization interfaces in two significant ways. First, compared to the current systems that focus on NL-first interactions, my work investigates multimodal systems that combine NL and DM in more synergistic ways. Second, my research focuses on speech as the primary form of NL input (as opposed to typing in current systems). Together, these distinctions enable me to expand the design space of NL-based visualization interfaces and investigate new modalities and settings that go beyond traditional desktops (e.g., mobile devices, large displays). Furthermore, since the phrasing of utterances varies notably between typing and speech, as well as between speech-only and multimodal utterances [137], my research also contributes novel input mechanisms and query interpretation techniques not considered by existing NLIs for data visualization.

# CHAPTER 3
# CHARACTERIZING THE ROLE OF NATURAL LANGUAGE INTERACTION WITHIN VISUALIZATION TOOLS

The premise of this thesis involves the integration of two styles of interaction: DM and NL. Given the prevalence of DM in desktop-based systems over the past three decades, there has been extensive research investigating DM both in the context of general user interfaces (e.g., [58, 59, 80]) and visualization tools (e.g., [53, 148, 163]). However, due to the nascency of NLIs for data visualization when I started this research (2016), it was less clear what role NL could play in the context of visualization tools. To better understand this, I conducted a literature review, analyzing and characterizing NL queries supported in visualization NLIs at the time. Besides queries supported in the reviewed visualization NLIs, I also identified new classes of NL queries based on additional brainstorming and examples from the broader HCI and databases literature. In this chapter, I summarize this activity as a framework to highlight the potential goals (e.g., creating and formatting charts, performing analytical computations) that NL may be used to accomplish within a visualization system[1]. Besides describing the framework, I also discuss how the gaps and opportunities identified through this review motivated the subsequent projects of this dissertation.

## 3.1 Methodology

To understand the potential role of NL within visualization tools, we examined the five NLIs for data visualization that were published at the time: InfoStill modified with the Sisl framework by Cox et al. [42], Articulate by Sun et al. [193], DataTone by Gao et al. [62], Eviza by Setlur et al. [172], and Articulate2 by Aurisano et al. [9, 99]. Additionally, to gain

[1]The content of this chapter is based on work previously published as a short paper in EuroVis 2017 [188].

a broader perspective of the possibilities with NL interaction, we also considered related work in the broader HCI domain (e.g., NLIs for graphics manipulation [65, 71] and geographic information systems [37, 137]) and NLIs for databases, or NLIDBs (e.g., [8, 150]). We considered these related topics given their overlap with visualizations (e.g., graphics manipulation operations can be used for formatting and creating illustrative visualizations) and the general task of querying data (e.g., similar to visualization tools, NLIDBs also help people understand data albeit by executing computational functions through NL).

We began by considering queries/utterances illustrated in the reviewed systems (NLIs for visualization, general HCI, and NLIDBs). We then used an affinity diagramming approach, grouping queries with similar intents and iteratively refining the groups according to what we believed to be the core objective of the queries within each group. We finally combined these query groups under broader categories of user goals based on existing visualization task and interaction taxonomies [3, 134, 221]. This process resulted in three high-level categories of NL queries that people may pose to a visualization system: *visualization-oriented* queries, *data-oriented* queries, and *system control-oriented* queries. The next section details these categories along with their sub-categories and example queries.

## 3.2 Goal-based Framework of NL Queries for Data Visualization Tools

Table 3.1 summarizes the categories of user goals identified through our review and analysis (along with the system support for these goals). More descriptions of the individual goal and categories along with sample queries[2] are provided below.

### 3.2.1 Visualization-oriented queries

Visualization-oriented queries are those that focus on generating visualizations or updating an existing visualization. These are the most popular type of queries and typically involve references to visualization types, objects within an active visualization (e.g., axes, marks),

[2]Some sample utterances/queries presented in this section are taken from published articles corresponding to the surveyed visualization systems [42, 62, 99, 172, 193].

Table 3.1: Characterizing existing NLIs for data analysis with visualization based on goals/tasks they allow people to perform **via natural language**. ☐ indicates a system provides minimal support for a task. ✎ indicates a system moderately supports a task. ■ indicates a system focuses on a task. "*" indicates that a task was identified as a part of an initial/formative study but it was not clear if the task was supported in the implemented system.

| | | InfoStill | Articulate | DataTone | Eviza | Articulate2 |
|---|---|---|---|---|---|---|
| **Visualization-oriented** | Generating Visualizations | ☐ | ✎ | ■ | | ✎ |
| | Visualization Specific Interactions | ☐ | ☐ | ☐ | ■ | ✎ |
| | Formatting Visualizations | | | | * | |
| **Data-oriented** | Low-level Analytic Operations/Questions | | ☐ | | ☐ | * |
| | High-level Questions | | | | * | |
| **System control-oriented** | | ☐ | | | | ☐ |

and one or more visual analysis operations (e.g., filtering, changing visual encodings). More specifically, *generating visualizations* in Table 3.1 refers to the task of constructing or requesting new visualizations. *Visualization specific interactions*, on the other hand, involve performing specific interactions with respect to an active visualization. The seven categories of interactions proposed by Yi et al. [221] (*Select*, *Explore*, *Reconfigure*, *Encode*, *Abstract/Elaborate*, *Filter*, *Connect*) can be considered as examples of interactions people may try to perform. *Formatting visualizations* involves updating an active chart purely from a graphical perspective (e.g., show/hide labels, change color of marks from red to blue). Sample utterances within this category include:

*Show me medals for hockey and skating by country. — Show Africa only — Select the largest bar in View 3. — Sort by average unemployment — Can you show it around the Loop by year broken down by crime type? — Bind SAT average to the x-axis and color by Region. — Visualize the correlation between MPG and Horsepower as a scatterplot. — Show labels for all schools in the Far West — Color movies that grossed over 100M green.*

### 3.2.2 Data-oriented Queries

Data-oriented queries are questions targeted at the underlying dataset and may not contain a reference to a visualization or specific visual analysis operations. We divide data-oriented queries into two main categories: *low-level analytical questions* and *high-level questions*. Examples of low-level questions typically include some of the analytical tasks proposed by Amar et al. [3] such as *retrieve value*, *compute derived value*, *find extremum*, *determine range*, among others. On the other hand, high-level questions are queries (e.g., "*Which stock should I buy today?*," "*Are schools in the Western region better than those in New England?*") that do not express a specific goal or task but request an answer for a more general question that needs to be derived using one or more analytical operations and mathematical functions. Responses for data-related tasks may not need to show or modify a visualization but could be just plain text (e.g., a yes/no response or a number indicating a requested value). However, visualizations can be used to enhance the understanding of the response (e.g., showing a scatterplot with a regression line when the person asks for a correlation value). Sample utterances for data-related tasks include:

*What is the range of MPG? — What is the male population at the age of 20? — How many services had a total cost above 50 million? — During what time is the crime rate maximum, during the day or the night? — What is the best time to produce decaffeinated coffee? — Is there a seasonal trend for bike usage? — Which stock should I buy today?*

### 3.2.3 System Control-oriented Queries

The final category, system control-oriented queries, refers to queries where people leverage NL to perform interface operations such as to move windows, change interaction tools, inquire about what queries can be issued, or ask questions to confirm their understanding of the data or view they are looking at. In some sense, there are meta-commands that are issued to augment UI operations or to refer to the ongoing analytic session/process. Sample

utterances for this category include:

*Move this window to the top right corner. — Help, what can I ask now? — Can you close the graph? — What other chart can I look at? — Bookmark this chart — Activate the annotation mode and give me a red brush.*

Note that the above categorization is neither exhaustive nor mutually exclusive (e.g., a query may request a new chart and a selection within that, or one might ask the system to compute the correlation and overlay that onto the view). The goal of Table 3.1 was for us to broadly understand the potential role of NL in visualization tools, assess the state of the field at the time, observe similarities and differences between existing systems, and identify themes to focus on going forward.

## 3.3 Motivation to Explore Speech Input and Post-WIMP Multimodal Interfaces

Four out of the five systems reviewed systems were designed for desktops and supported NL querying through typed input. Furthermore, while the formative elicitation study for the fifth system, Articulate2 [9], considered how participants would speak to the system, the eventual prototype [99] required users to type commands into a laptop that were subsequently propagated to a larger, interactive display. Although valuable, focusing only on desktop settings constrained the input and interaction space that could be explored and led to interfaces that strongly resembled existing DM-based visualization tools. As an alternative, drawing inspiration from research on voice user interfaces (e.g., [33, 151]) and keeping in mind the growing popularity of post-WIMP visualization interfaces [52, 107, 154], I decided to consider speech-based NL interaction (as opposed to only typed input) and investigate its effects in the context of visualization tools.

Given the prevalence of DM in visualization tools, it was no surprise that three out of the five reviewed systems [42, 62, 172] supported some level of multimodal interaction involv-

ing typed NL and mouse-based actions (e.g., clarifying ambiguous references in a query through a dropdown menu [62], typing a NL query + lassoing regions on a map [172]). However, this style of multimodal interaction falls well short of examples from the broader HCI literature (e.g., [19, 36, 37, 139]) that illustrate the symbiotic relationship between NL and DM through fluid interactions involving speech and gestures, or pen/touch and speech. As part of my work, I wanted to explore how this same fluidity can be brought to visualization tools and inspect the expressive potential of NL input. With this in mind, I decided to focus on NL- and DM-based multimodal interaction in post-WIMP settings (e.g., tablets, large interactive displays) as one of the themes for my research.

## 3.4 Summary

In this chapter, I described a literature review and analysis to understand the potential role of NL in visualization tools. The two main outcomes from this exercise include a goal-based framework of NL queries for data visualization tools **[RG1]** and a characterization of the visualization NLIs that existed at the time based on this framework. Furthermore, this review also helped me identify research gaps (e.g., no support for visualization formatting through NL) and opportunities (e.g., investigating NL interaction in post-WIMP settings) that motivated, in part, the projects presented in the remainder of this dissertation.

# CHAPTER 4
# INVESTIGATING TOUCH- AND SPEECH-BASED MULTIMODAL INTERACTIONS WITH VISUALIZATIONS

As discussed in the previous chapter, through a review and analysis of prior work, we identified the exploration of post-WIMP multimodal visualization interfaces as an open research opportunity. Next, to validate the feasibility of this idea and investigate multimodal interaction with visualizations, we started by considering speech and touch input.

We focused on touch since it was (and continues to be) one of the most popular means for DM in post-WIMP visualization interfaces. This popularity and prior research on touch provided two benefits: i) it offered a basic set of interactions as a reference point and ii) it increased the plausibility of generalizing the findings from this research to other systems (both existing and new touch-based tools). Since there were no other touch- and speech-based systems when we commenced this research, to scope the initial design and exploration, we began by focusing on network visualizations (specifically, node-link diagrams). Questions we considered as part of this work included: How can multimodal interfaces support fundamental operations during visual network exploration? Do they lead to an improved user experience, and if so, why? How do people leverage touch and speech to interact with network visualizations? Are there specific mappings between operations and modalities?

The remainder of this chapter is organized as follows[1]. First, I discuss the rationale for focusing on network visualizations and highlight, in particular, challenges in supporting NL interactions with network visualizations. Then I describe a prototype system ORKO, that supports multimodal interactions with network visualizations. I detail two user studies we conducted using ORKO as a testbed to gather preliminary feedback and better understand

[1]The content of this chapter is based on work previously published in IEEE TVCG [166, 189].

multimodal interactions with visualizations. Collectively, ORKO's design and feedback from the two studies described in this chapter show (in the context of network visualizations) that multimodal interfaces can support common visual analysis scenarios while enabling a fluid and expressive interaction experience **[RG2]**.

## 4.1 Why Design Multimodal Interfaces for Network Visualizations?

Network visualizations are an extensively studied class of representations within the visualization literature (e.g., [17, 74, 205]) and are useful for describing and exploring data relationships in many domains such as biology [125], the social sciences [131], transportation planning [120], and business intelligence [13], just to name a few. When visually exploring networks, people often need to focus on subgraphs of interest (e.g., by selecting specific nodes and links, filtering), investigate specific connections (e.g., finding adjacent nodes, following paths), and adjust the visual properties of the network (e.g., changing graphical encodings such as color and size). Given this multitude of tasks, interaction plays a vital role during visual network exploration (e.g., [109, 152, 165]). As with most existing visualization tools, current network visualization tools predominantly support interactions via DM and WIMP-style controls. Users leverage DM interactions to engage with both elements in the view (e.g., nodes, links) and interface controls (e.g., sliders, dropdown menus). However, as the size and complexity of the networks grow, such interactions become tedious to perform and require users to constantly switch their attention between the main view and interface controls that are external to the visualization itself. Besides tedium, this constant switching may also disrupt the analytic flow and have detrimental effects on the data exploration process.

One approach to address these issues is to incorporate NL interaction. Specifically, the freedom of expression afforded by NL can be a powerful addition to DM interactions commonly supported in network visualization tools. Together, NL and DM may facilitate a better analytical flow by allowing people to perform operations such as filtering and

changing visual encodings (through speech) while directly interacting with the network (through touch). To examine this idea, we first broadly considered the general space of NL utterances for network visualizations.

## 4.2 Challenges in Interpreting NL Queries for Network Visualizations

While the designers of network visualization systems generally understand the challenges and issues of implementing DM interfaces, NLIs provide an altogether different set of challenges. For instance, consider the different types of queries or utterances that people may pose while working with a network visualization system.

To help understand the possibilities, we collected a set of sample queries by referring to existing network visualization task taxonomies [109, 152, 165] and through pilot studies with students and research colleagues. We then used an affinity diagramming approach, grouping similar queries and iteratively refining the groups according to how precise queries were in terms of conveying user intent. We made the assumption that user intent is conveyed by tasks and the values they map to. We then combined groups under broader categories. This process resulted in three higher-order categories of queries: *explicit*, *contextual and follow-up*, and *high-level* (Figure 4.1a).

For the remainder of the article, we will use a specific example, a network of European football (soccer) players, to help ground our discussions and make concepts more explicit. The network contains 552 players (nodes) and 6772 connections (links) between those players. A link is created between two players if they play for the same club team (league team) or the same national team. In addition to the name, club, and country information, other attributes associated with players include number of goals scored, market value (in USD), age, club, country, preferred foot, and field position.

Of the three categories of queries introduced above, explicit queries typically provide sufficient information in terms of both tasks and values for a system to parse the query and generate a response. Command-like queries can be considered as a subset of explicit

| | |
|---|---|
| **Explicit** | Find Ronaldo. — Show Pepe's connections. — Show connections between Pogba and Bale. — Show the shortest path from Evra to Kroos. — Color by position. — Size nodes by betweenness centrality. — What is the clustering coefficient of this network. — Only show German forwards. — Clear all filters. — Resize graph to fit the screen. — Add a filter widget for country. — Change value of the age slider to show players over the age of 30. — Change red nodes to blue. |
| **Follow-up & Deictic** | Are any of these players right footed. — Filter by this player's club. — Show connections of these players. — Do any of these players play for the same club and national team. — Show the different countries players come from. — Ronaldo and Rooney. — Color nodes by country > Now club > How about position? |
| **High-level** | How are France and Italy connected. — Players from which countries tend to play more with clubs in the same country. — Which clubs have more left footed players. — Which countries have highest number of common players. — Modify the network layout to focus on England players. — Which three nodes have highest betweenness centralities. — Modify layout to show least edge crossings. — Find clusters. |

**(a)** Possible query types

***Show nodes connected to Ronaldo.***

Show Ronaldo's connections.
Find players linked to Ronaldo.
Highlight players who play with Ronaldo.
Which players play in the same team as Ronaldo.
Show nodes directly connected to Ronaldo.
Find nodes adjacent to Ronaldo.
Show Ronaldo's teammates.
Who all is Ronaldo directly connected to.
Find players with a direct link to Ronaldo.
Find direct connections of Ronaldo.

**(b)** Query phrasing variations

Figure 4.1: An illustration of the variety of potential NL utterances in network visualization systems.

queries. Examples of these types of queries include "*Find Ronaldo*" or "*Show the shortest path from Evra to Kroos*".

Given the conversational nature of NLIs, users may frequently pose queries that are follow-ups to their previous queries. Such queries typically lack references to tasks or values associated with a task. For example, consider the query "*Color nodes by country*" followed by "*Now club*", followed by "*How about position?*". While the queries following the first one appear incomplete individually, they refer to the coloring task implied by the first query. In a multimodal setting, users may even present deictic queries that are follow-ups to their DM actions. For example, if the user selects a subset of nodes and utters the query "*Show connections of these players*", the system needs to detect that the user is referring to the selected players and automatically map the task of finding connections to those players.

Finally, high-level queries are generally open-ended user questions. These questions typically do not specify explicit tasks and can be interpreted and answered in multiple ways depending on the interpretation. Examples include questions like "*How are France and Italy connected?*" or "*Which countries have most of their national team players in their local clubs?*" To generate a response for such queries, systems typically need to break the question into smaller tasks, solve those tasks and combine the results into a final response.

The sheer variety of query phrasings poses another challenge. Given the freedom of

expression associated with NL, a person's particular intent can be stated in multiple ways. For example, Figure 4.1b shows some of the many ways that a person could state a query to find the connections of a node. Additionally, other challenges of NL such as ambiguity exist as well. Ambiguity may exist not only at a syntactic level (e.g., misspelled words) but also at a semantic level in the context of words (e.g., synonyms and hypernyms) and the data (e.g., "*high goal scorers*" can refer to players with over 10 goals, 20 goals, etc.).

The presented examples and classifications in Figure 4.1 are not exhaustive, nor is the goal here to provide a definitive taxonomy of query types. Instead, the purpose of this section is to present a general overview of the input space of NL queries for network visualizations in the targeted multimodal setting.

Next, with this query set and associated design/implementation challenges in mind, we developed a touch- and speech-based multimodal network visualization tool, ORKO.

## 4.3 ORKO

### 4.3.1 Design Goals

Although our general objective of building ORKO was to explore and assess NL- and DM-based multimodal interactions with network visualizations, two primary high-level goals drove the system design.

**DG1. Facilitate a variety of network visualization tasks.** A core goal for ORKO was to support exploratory analysis similar to that done in existing desktop-based network visualization systems (e.g., [14, 115]), but in a multimodal setting. More specifically, we wanted to focus on supporting a variety of tasks including topology-based, attribute-based, and browsing tasks in context of the taxonomy by Lee et al. [109], a subset of structure-based and attribute-based tasks at a node (entity) level per the taxonomy by Pretorius et al. [152], and finally, a small subset of group-only, and group-node tasks as specified in the taxonomy by Saket et al. [165].

**DG2. Facilitate a variety of input integration patterns for multimodal interaction.** Multimodal interfaces provide more freedom of expression allowing users to interact with the system in multiple ways. For such systems, input patterns are typically categorized based on the number of modalities used and temporal synchronization between modalities [140, 141, 145]. Given the system's primary usage setting (touch and speech input), our goal was to support a variety of these input patterns, including unimodal input (touch-only, speech-only), sequential multimodal input (e.g., selecting nodes via touch followed by a pause followed by a *find connections* query), and simultaneous input (e.g., issuing a *Color nodes* query while highlighting a node's connections). More specifically, in addition to incorporating both input modalities individually, this goal required us to consider synergies between the two modalities while designing the interface, and support not just explicit and follow-up but also contextual queries (Figure 4.1a). We chose not to focus on high-level questions (Figure 4.1a) as we believe they are more specific to NLIs. Further, these queries pose a challenge of overcoming the variability in responses that is beyond the scope of our current work focusing on multimodal interaction.

### 4.3.2 System Overview

Implemented as a web-based tool, ORKO runs on both conventional desktops/laptops and other devices supporting touch and speech interaction. Figure 4.2 shows the initial version of ORKO's interface.

At the top of the window (Figure 4.2A), is an input box that shows typed or spoken NL queries. Users can input NL queries in two ways: pressing the microphone button (🎙) next to the input box and speaking the query, or by saying "System" and then speaking the actual query (similar to the interaction with Amazon.com's Alexa [4] or Google's Assistant [63]). Below the input box is an action feedback row that conveys the changes made to the interface as part of the response to a query. ORKO also provides optional audio feedback where the system narrates the content of the feedback row. Some examples of feedback

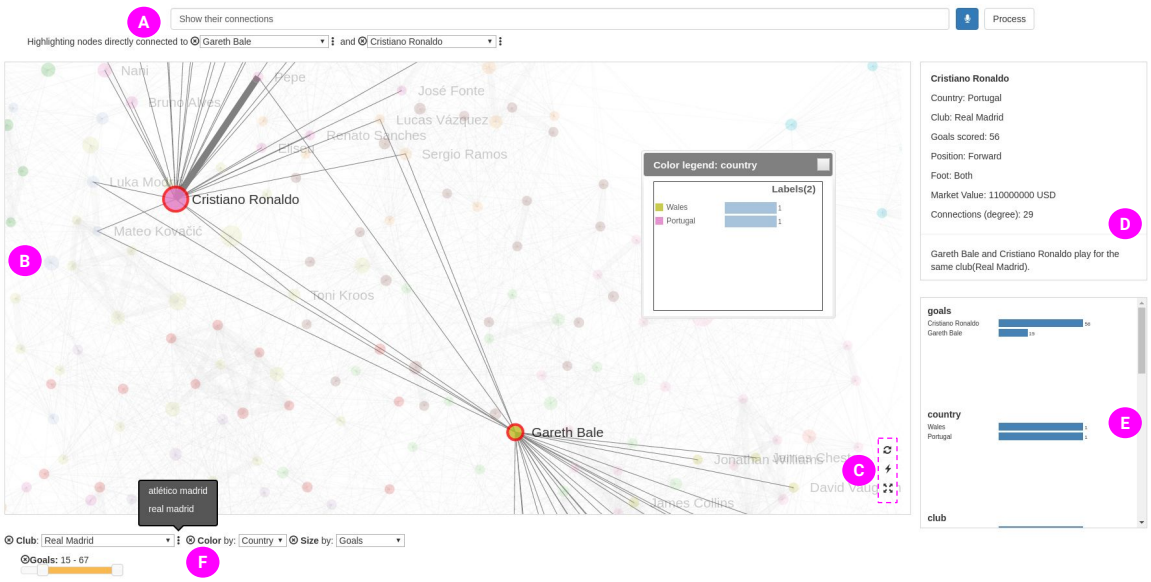



Figure 4.2: ORKO's initial user interface shown in the context of exploring a network of European soccer players. The players Cristiano Ronaldo, Gareth Bale, and their connections are highlighted. Connected nodes with lower opacity do not meet all the filtering criteria. Interface components include: (A) NL input and action feedback row, (B) network canvas, (C) quick-access icons, (D) details container, (E) summary container, and (F) filters and visual encodings row.

messages include "Highlighting nodes directly connected to Gareth Bale", "Changed coloring attribute to Club", or "Sorry I'm unable to process that query. Can you try asking the question differently?".

The network canvas (Figure 4.2B) displays the network visualization with nodes represented as circles and links between nodes represented as lines connecting the circles. Node positions are determined by D3's force-directed layout [20]. By default, labels are only shown when nodes are selected or highlighted to avoid clutter. Quick access icons (Figure 4.2C) are provided at the bottom right of the canvas to reset the view by clearing all selections and filters (🔄), unpin all pinned nodes and reset the force-layout (🗲), and re-center the network (🔀).

We implemented the DM interactions with the visualization such that they do not rely on hover (unavailable on commonly found touchscreens [75]) and can work on both touch and pointing devices (e.g., mouse, stylus). Users can tap to select individual nodes, draw a

lasso to select a group of nodes, double-tap to highlight a node's connections, drag a node to pin it and modify the layout, long press (individually) on two nodes to highlight the shortest path between them, and finally, zoom and pan using pinch-and-drag gestures on the canvas background. Users can clear selections by tapping on the canvas background. When a node's connections are highlighted, details of individual links can be seen using a single-tap on the link. Keeping in mind issues like the fat-finger problem [181], we sized nodes such that it is easy to tap them and detected interactions within a broader vicinity of links to handle offset touches. The details container (Figure 4.2D) shows attributes of selected nodes and link descriptions.

The summary container (Figure 4.2E) shows bar charts that complement the selections on the main canvas. The charts present attribute-level summaries for active nodes (i.e. nodes which meet any applied filtering criteria). The bar charts are coordinated with the network visualization and facilitate brushing-and-linking. The bars within charts are sorted in descending order of width from top to bottom to facilitate ordered comparisons. The charts dynamically reorder based on the sequence of user interactions with attributes—the most recently used attribute is always shown on top of the container. We made this design decision of reordering charts for two reasons. First, users would find it beneficial to see the summary statistics for attributes they most recently used to answer possible questions they have in mind for the attribute. For instance, if a user filters nodes by goals scored, the summary chart for the 'Goals' attribute would show up on top presenting a ranked list of the highlighted players and the number of goals they scored. Second, since the container shows all attributes available in the dataset, it could help facilitate an analytical conversation by triggering questions in users' minds about potential attributes they may not have considered. For instance, looking at the summary charts one may notice that there are twice as many right-footed players as there are left-footed players. This, in turn, could lead to a potential action of using the 'Foot' attribute to color or filter nodes to visually compare these values and identify players in each category.

Table 4.1: Operations supported in ORKO with corresponding touch-only, speech-only, and multimodal interactions. Note that the speech commands shown are only examples and the systems supported a wider variety of phrasings. Speech commands preceded by " $>$ " are examples of follow-up commands.

| **Operation** | **Touch** | **Speech** | **Multimodal** |
|---|---|---|---|
| Find Nodes | Type label using virtual keyboard | *"Find Ronaldo", "Highlight David de Gea", "Search for Gareth Bale and Rooney"* | *"Find Pepe"* + update node using query manipulation widgets (Figure 4.2A) |
| Find Connections | Double tap on a node | *"Show players linked to Iniesta", "Highlight Griezmann's teammates"* | Select nodes + *"Show connections", "Find players who play with Marcelo"* + Update node using query manipulation widgets |
| Find Paths | Long press on source node and tap on target node | *"How are James McCarthy and Toni Kroos connected?", "Highlight a path from Sergio Busquets to Patrice Evra"* | Select nodes + *"Highlight path", "Show connection between Giroud and Neuer"* + Update nodes using query manipulation widgets |
| Filter Nodes | Adjust dropdowns and sliders in filters & encodings row (Figure 4.2F) | *"Just show left footed Madrid players", "Filter to show German and French players" >"Highlight ones with more than 10 goals"* | *"Filter by degree"* + Adjust degree slider, *"Only highlight strikers and midfielders"* + Change field positions using dropdown |
| Change Visual Encodings | Adjust dropdowns in filters & encodings row | *"Color players by their field positions", "Size by goals scored" >"Now by age"* | *"Color by country"* + Change attribute using color dropdown, *"Resize nodes"* + Select attribute from dropdown |
| Navigate | Two-finger pinch for zoom, one-finger drag on canvas for pan | *"Zoom out", "Center graph", "Zoom in more", "Pan left" >"Some more", "Move right"* | — (only supported through touch *or* speech) |
| Interface Actions | Tap quick-access icons (Figure 4.2C) | *"Refresh canvas", "Show all node labels"* | Select nodes + *"Show labels"* |

The filters and encodings row (Figure 4.2F) presents the various filtering and encoding options. Filtering widgets include range sliders for numerical values and dropdown menus for categorical values. Visual encoding widgets include dropdowns for assigning node coloring and sizing attributes. Users can choose to keep the widgets on or remove them at any point using the ⊗ icon next to each widget.

To support the targeted categories of network analysis tasks **(DG1)**, ORKO supports currently provides a set of seven low-level operations listed in Table 4.1. Table 4.1 also illustrates how the different operations can be performed using touch, speech, or a combination of the two modalities.

With respect to the categorization of potential user goals presented in Chapter 3, ORKO currently focuses on supporting *visualization-specific interactions*, and provides basic support for *low-level analytical operations* and *system control-related tasks*.

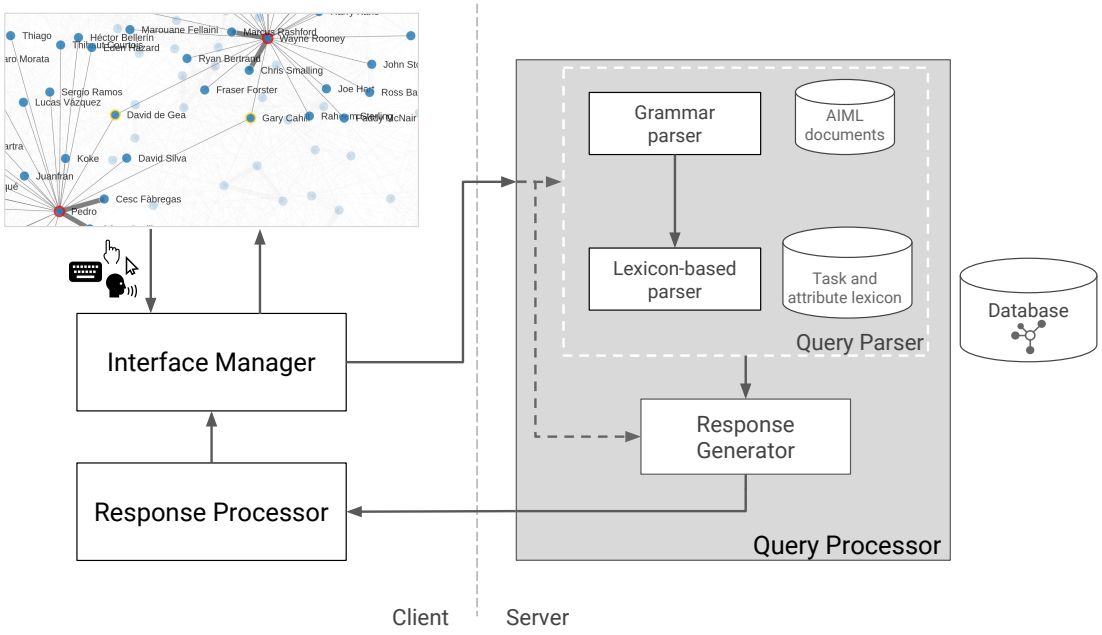



Figure 4.3: ORKO system architecture highlighting the key modules and their connections.

### 4.3.3 System Architecture and Design

To facilitate multimodal interaction and support the different combinations of input patterns **(DG2)**, ORKO employs a client-server architecture shown in Figure 4.3. Below I describe the individual components highlighting their specific functions along with the interplay between the different components.

*Processing unimodal DM or NL input*

All DM events triggered on ORKO's interface (e.g., tapping a node, changing a dropdown value, adjusting a slider) are collected and handled by the *interface manager*. We use the HTML5 webkit speech recognition API for detecting voice-input and performing speech-to-text translation. To improve recognition accuracy, we also trained the recognizer with system keywords (e.g., 'find,' 'color,' 'path') and dataset attributes/values. Given a query string, the interface manager sends it to the server for interpretation. On the server, the *query processor* parses NL queries and generates a list of actions that need to be taken in response to a query.

To parse NL queries, we implement a two-step approach combining both grammar and lexicon based parsing techniques. This approach lets the system parse queries that match a grammar pattern instantaneously while also having a more general back-up parsing strategy based on a combination of keyword-matching and heuristics for query patterns not covered by the specified grammar.

Grammar patterns are specified using Artificial Intelligence Markup Language (AIML) [26]. The patterns were generated based on question types and examples presented in existing task taxonomies [109, 152, 165] and pilot studies with students and research colleagues. We use a Python-based AIML interpreter (PyAIML) trained with the AIML files to parse incoming query strings. The interpreter builds a directed pattern tree and employs a backtracking depth-first search algorithm for pattern matching. For a given query, the grammar parser seeks to identify operations (Table 4.1) specified and substrings containing references to attributes or values the operations apply to (analogous to a set of non-terminal symbols in a context-free grammar [43]). If there is no matching pattern found, the entire query is forwarded to the second parser, else, only the target referencing substring is sent to the lexicon-based parser. For instance, given a query like "*Show connections of Ronaldo*", the grammar parser identifies that the operation is *find connections* and the target is 'Ronaldo' (which is sent to the second parser). Alternatively, a query like "*Show only if Barcelona and left foot*" may not match an existing pattern and will be forwarded as-is to the lexicon-based parser.

The lexicon used consists of attributes derived from the dataset (e.g., *goals*, *country*, *names*) and manually specified keywords (e.g., teammates, adjacent, striker) that help identify attributes, values, and operations in a given query. Some of these keywords are generic and apply to multiple datasets (e.g., 'connections', 'adjacent') while others are dataset-specific (e.g., 'striker', 'teammate'). While we leverage existing lexical databases like WordNet [130] to support using synonyms (discussed further below), there always will be dataset-specific cases that are not supported by such general databases (e.g., using "striker"

instead of "forward" for position). For such cases, in our current implementation, both, domain-specific grammar patterns and dataset-specific keywords should be manually added the first time a dataset is loaded.

Given a portion of the query or the entire query string, the lexicon-based parser first performs stemming and removes stop words (with the exception of conjunction/disjunction phrases). It then extracts n-grams (with n ranging from 1 to the number of words in the input string). For each n-gram, it identifies POS tags (e.g., noun, verb) and entity types (e.g., person, location, organization) using NLTK [114] and Stanford CoreNLP [122]. n-grams not containing entity types relevant to the dataset or values that may apply to filters (e.g., numbers) are discarded. This filtering helps improve performance by ignoring n-grams that do not contain relevant information. Next, the relevant n-grams are compared to logically similar lexical entries (those that have related POS tags or entity types). This similarity-based comparison again helps boost performance by avoiding matches against potentially irrelevant values (e.g., comparing people to locations). Building upon existing work [62, 172], we use the cosine similarity and the Wu-Palmer similarity score [216] when comparing n-grams to lexical entries. These scores help in detecting both syntactic (e.g., misspelled words) and semantic (e.g., synonyms, hypernyms) ambiguities. If there are no operations identified by the grammar parser, similar to the logic in DataTone [62], we use keyword-matching and a combination of POS-tags and dependency parsing techniques [122] to identify operations specified in a query. In summary, for the query "*Show only if Barcelona and left foot*", the lexicon-based parser identifies a *filter* operation, a *club* ("Barcelona FC"), and a value for *foot* ("left").

*Managing multimodal input*

In addition to its focus on network visualizations, ORKO's primary difference compared to earlier systems (e.g., [42, 172]) was its support for various multimodal interaction patterns listed in **DG2**. As an example of the input patterns ORKO supports, consider the case

of finding connections of a set of top goal scoring players for England. A user could accomplish this via only touch by applying multiple filters (for country and goals) and double tapping nodes to highlight connections. Instead, one could also use speech alone to perform the same task (using a single query like "*Show connections of English players with more than 20 goals*" or multiple smaller queries). Alternatively, a user could use a combination of the two modalities and: (1) apply filters (via touch) and follow it with a spoken query (e.g.,"*Show adjacent nodes*"), or (2) apply filters via speech and then double-tap nodes, or (3) do both filtering and uttering a query simultaneously (starting with either of the two modalities). In cases (1) and (3), the context generated by one input is used to complement the second and highlight connections of the filtered nodes. For (2), the system processes the two inputs individually as described in the subsection above, preserving filters from the spoken query.

To support the patterns described above, the system needs to first classify input patterns and then share relevant information collected across input modes to appropriately respond to the user input. To accomplish this, ORKO first classifies an input pattern as unimodal, sequential, or simultaneous. To classify an input pattern, we use a combination of interface context and time lag between user inputs. The interface context is tracked using an object that stores information about active/highlighted nodes, filters applied, encodings used, previous interaction modality used, and operations and target values in the last specified query. Both the interface manager and the query processor continually update this context object based on user inputs and actions.

When a user input (touch or speech) event occurs, we check in parallel if there is a change in the modality used between inputs. If so, we further check if there is any missing information in the input (e.g., missing target value in a query) and a corresponding interface context that can be applied to the current input. For example, if a user selects two nodes (via touch) and then issues a query "*Find connections*", ORKO can leverage the context of the selected nodes and apply it to the user query. We use a heuristic approach and mappings

between operations and attribute types (e.g., topology operations such as *Find Nodes*,*Find Path* applies only to label attributes, *Filter* applies to all attributes) to decide if a context applies to an input. However, an applicable interface context could be generated in both sequential and simultaneous input.

To differentiate between the two, when there is an applicable interface context, we also check the time lag between the previous and recent input. Based on prior work on multimodal input patterns [141] and our pilot studies, we differentiate between sequential and simultaneous input based on a time lag of two seconds between modalities. We make this differentiation to decide when context from the previous input should not be applied to the current one. For example, consider a case where there are no selected nodes and a user issues a query "*Show nodes connected to*" and follows it by a long pause. Now, the user adds a filter (via touch) to highlight the Spanish players. If the context from the query is applied by default, connections of the filtered nodes will automatically be shown. However, after such a pause, it is likely that the user was trying to perform a new filter action and wanted to ignore the previous query. In such cases, since we know that the pattern in this case was sequential, we can choose to not apply the system context and ignore the previous query instead.

*Supporting follow-up queries and query refinement*

To handle follow-up queries, we implement a conversational centering [67, 68] (or immediate focusing [66]) based approach. The centers are maintained by the query processor and include operations, attributes, and values. We *retain*, *shift*, or *continue* centers across utterances [68] depending on the information shared or added between queries. Consider the query "*Show only Real Madrid players*" followed by "*Show strikers*". In this case, the *club* filter 'Real Madrid' is retained across queries, and a *position* filter ('striker') is added after the second query (a *continue* operation). Now, when another query "*Show defenders for Barcelona*" is presented, the center is shifted to 'defender' and 'Barcelona FC'

respectively.

Since we wanted to deploy ORKO in a speech+touch setting where typing to modify specific parts of a query or repeatedly uttering similar queries can become tedious, we also explored interface elements that may assist users in modifying their queries. We considered ways in which we could assist users to ask follow-up queries that refer to the same operation but different target(s) (e.g., "*Show Real Madrid players*" followed by "*Show Barcelona players*"). Such queries can be very common during network exploration, particularly while users try to scan through different node groups. To assist in constructing such follow-up queries, ORKO adds *query manipulation widgets* (dropdowns and sliders) to the action feedback row highlighting the domain of input values for an attribute alongside the operation being performed (Figure 4.4A). Hence, for the club example discussed above, the user can specify a *filter* by club query once and then keep updating the club names for consecutive queries using the dropdown. Users can still ask follow-up queries or modify existing queries by typing if they prefer to do so. To allow users to instantly revert back to the original query, it is preserved in the text box (Figure 4.2A).

*Highlighting ambiguity in queries*

ORKO also presents ambiguity widgets [62, 172] to highlight and help users interactively resolve ambiguity in NL queries. Supported ambiguity widgets include range sliders (for numerical attributes) and interactive tooltips (for label and categorical attributes). The interactive tooltips work as follows: vertical ellipsis icons (⋮) are added next to query manipulation dropdowns to notify users that the system detected ambiguous matches. When pressed, these icons display a tooltip showing the list of matched values (Figures 4.2F and 4.4B). Selecting an item in the tooltip updates the value of the adjacent dropdown. By default, the query manipulation dropdown is populated with the value most similar to the ambiguous string.

We also explored how the system can suggest operations in cases where a presented

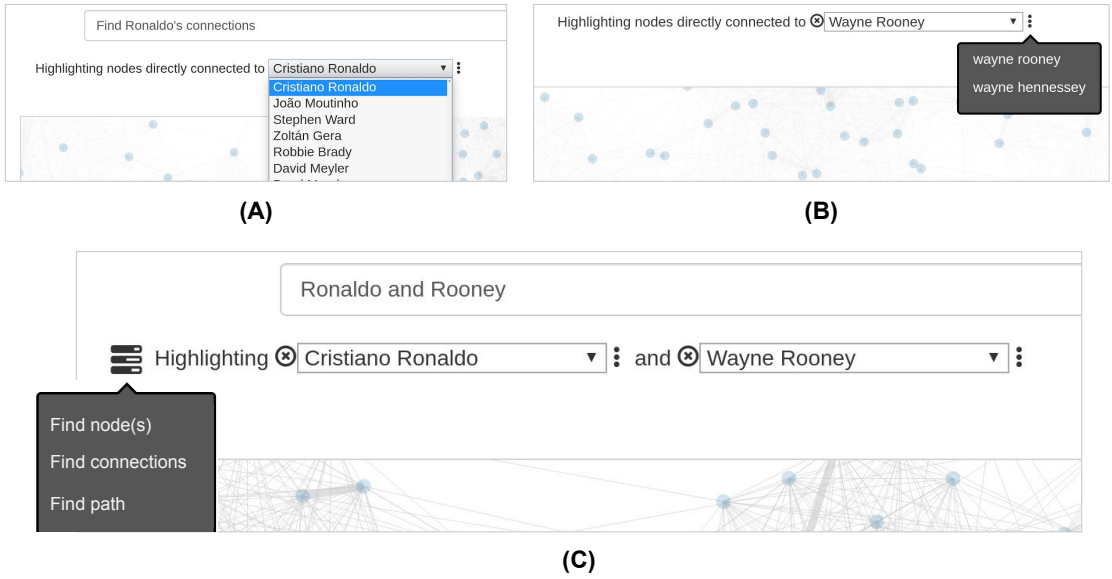



Figure 4.4: Query manipulation and ambiguity widgets. (A) Dropdown menu to change player name in a query, (B) tooltip showing values matched to an ambiguous word 'Wayne', (C) tooltip suggesting tasks guessed by the system for an underspecified query.

query lacks references to operations (or there is ambiguity in operations) but there is no preceding query or an applicable interface context to leverage. Such queries can be common with issues in speech detection that lead to partial detection of queries (e.g., "*Rooney and Ronaldo*" instead of "*Find connections of Rooney and Ronaldo*"). When the response generator detects attributes or values in a query but is unable to map them to a specific operation, it makes a "guess" at the operation that a user could perform based on a reverse mapping from attribute types to the list of available operations. An example of this is shown in Figure 4.4C where three operations, *find nodes*, *find connections*, and *find path* are suggested in response to an underspecified query using an interactive tooltip.

## 4.4 Study 1: Preliminary Assessment of ORKO's Usability and Interactions

With a functional prototype, we conducted a preliminary user study to evaluate ORKO's design and multimodal approach for interacting with network visualizations. Our primary objectives were to 1) assess ORKO's usability, 2) collect user feedback on the designed set

of touch and speech interactions, and 3) understand how people leverage these interactions when working with network visualizations.

### 4.4.1 Study Details

*Participants and Setup*

We recruited six participants (P1-P6), ages 22 to 42, five male and one female. All participants had prior experience with visualization tools such as Tableau and had at least once worked with Gephi [14], a popular network visualization tool. Two participants stated they had some prior experience working with touch-based visualization systems and only one participant (P2) had never used a voice-based interface (e.g., Siri). All participants interacted with ORKO running on Google's Chrome browser on a 55" Microsoft Perceptive Pixels (PPI) device. The PPI was set to a resolution of 1920 x 1080 pixels.

*Dataset and Tasks*

All participants explored the network of European soccer players introduced earlier in Section 4.2 and performed 10 tasks. In terms of dataset's domain knowledge, two participants were well acquainted with the sport of soccer and the data, and remaining four stated they were aware of it but did not follow the game or know much about the data. Due to the availability of speech as an input modality, phrasing the tasks was a challenge in itself. Specifically, posing tasks as direct questions was not an option since participants could simply repeat (say) the questions. Thus, we adopted the Jeopardy-style evaluation approach proposed by Gao et al. [62]. We gave participants a set of facts and asked them to modify the visualization to show each fact. For example, one such fact was "Robbie Keane only plays with Irish players." To 'answer' this, participants would need to show that all of Robbie Keane's connections belong to Ireland (i.e., participants would need to find connections + color nodes by country, or find connections + scan summary charts, or find connections + scan individual nodes). We also added some entity naming tasks that required partici-

pants to explore the network to identify specific entities. The tasks again were framed so that participants could not simply parrot them into the system to get an answer. For example, one of the questions was "Name an FC Barcelona midfielder. Identify at least two non-Barcelona midfielders the player is connected to." To respond to the first part of this question, participants would have to name (using filter or find) and highlight a Barcelona midfielder. For the second part, they would have to highlight the player's connections with two other non-Barcelona players and show that those two are also midfielders.

*Procedure*

Participants were given a brief introduction (approximately 5-7 minutes) to the system's UI and the European soccer player dataset. For the UI, we highlighted the different components (Figure 4.2), possible touch-interactions (tap, double-tap, drag etc.), and told participants that they could use typed (through a virtual keyboard) or speech for NL interaction. We did not show any trials or videos of how participants could perform any specific task since we wanted to observe their natural behavior and reactions. Participants were then asked to try the touch and speech input using any interactions and commands they wanted to test, until they felt comfortable (approximately 1-3 minutes).

Next, we gave participants the list of 10 tasks printed on a sheet of paper and 30 minutes to interact with the system. The order of the tasks was randomized for each participant. Participants were asked to interact as naturally as possible and were made aware that they we would not be judged based on task completion times. We recorded both video and audio of participants interacting with the system during the 30 minute task phase. Participants who finished the tasks before 30 minutes could continue exploring the data using the system if desired. Participants were then given a post-session questionnaire that consisted of System Usability Scale (SUS) questions and questions asking them about their experience with ORKO. We also conducted informal interviews asking the participants about what they liked/disliked most about the system and recorded their responses as audio files.

| | P1 | | | P2 | | | P3 | | | P4 | | | P5 | | | P6 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | S | T | ST | S | T | TS | S | T | ST | S | T | ST | S | T | ST | S | T | ST | TS |
| T1 | | | 1 | 2 | | | | | 1 | | | 1 | | | 1 | 1 | | | |
| T2 | 2 | | | 1 | | | 1 | | | | | 1 | | | 1 | 1 | | | |
| T3 | 2 | 2 | 1 | 3 | 1 | | 1 | | 1 | 3 | 1 | | 3 | 1 | | 2 | | | |
| T4 | 2 | | 1 | 3 | | | 4 | | | | | 3 | | 6 | | 3 | | | |
| T5 | 2 | | | 2 | | | | 1 | 1 | | | 1 | 2 | 4 | | 4 | 1 | 1 | |
| T6 | 1 | | 1 | 1 | | | | 2 | 1 | 1 | | | 1 | 3 | | 4 | | | |
| T7 | 1 | 1 | | 2 | 3 | | 1 | 1 | 1 | | 1 | 1 | 3 | 1 | | 2 | 2 | | |
| T8 | 1 | | 1 | 1 | | | 1 | 1 | 1 | | | 1 | 2 | 1 | | 1 | | | |
| T9 | 2 | | | 2 | | | | | 2 | 2 | | | 2 | 1 | | 1 | | 2 | |
| T10 | 2 | 2 | 2 | 8 | 1 | 2 | | 6 | 2 | 2 | 5 | | 2 | 5 | | 2 | 3 | | 1 |

| | P1 | P2 | P3 | P4 | P5 | P6 | Average |
|---|---|---|---|---|---|---|---|
| **Overall SUS scores** (out of 100) | 80 | 70 | 82.5 | 80 | 52.5 | 87.5 | **75.42** |
| **Would want to use the system frequently** (out of 5) | 4 | 5 | 5 | 5 | 3 | 5 | **4.5** |
| **Found various functions well integrated** (out of 5) | 5 | 5 | 4 | 3 | 5 | 4 | **4.33** |
| **Natural language query interpretation** (out of 5) | 4 | 4 | 3 | 4 | 4 | 5 | **4** |

Figure 4.5: (Left) Summary of interactions per study task for each participant. *S*: Speech, *T*: Touch, *ST*: Sequential speech+touch, *TS*: Sequential touch+speech. (Right) Participant respones for specific SUS questions and ORKO's query interpretation.

Overall, sessions lasted between 40-60 minutes and participants were compensated with a $20 Amazon gift card.

### 4.4.2 Results and Discussion

*System Usability Scale responses*

All participants attempted each of the 10 tasks and on average, provided correct responses for 8.67 tasks. Figure 4.5 (right) summarizes overall SUS scores. Participants gave ORKO an average SUS score of 75.42. SUS scores have a range of 0 to 100 and a score of around 60 and above is generally considered as an indicator of good usability [116]. The SUS scores indicate that even though the prototype was in its initial stages, participants in general found the interface and the interactions facilitated by ORKO satisfactory and usable.

*Feedback on multimodal interaction*

Overall, participants felt that the various features of the system were well integrated (Figure 4.5-right). Participants found multimodal interaction to be intuitive and stated they would want to use such a system frequently (Figure 4.5-right). One participant (P3) wrote "*It was fun to use and a very intuitive way to explore a network.*". Other participants even stated that they felt DM and NL-based multimodal input should become a part of network

visualization tools in general. For example, one participant (P4) who worked with network visualizations almost on a daily basis wrote, "*The ability to perform simple actions like "find node" and "find path between two nodes" was really fun to use, and I see this being highly used in general network visualization tools, especially for novice users*". He further stated that he felt that the speech input worked particularly well for navigation and topology-based tasks. He suggested that using NL for such tasks would be a great addition to keyboard and mouse based network visualization systems and can speed up user performance. He did state, however, that he still wanted to use DM for tasks like selecting specific values for graphical encodings or tuning parameters for analytical operations, emphasizing the value in having both modalities.

*Understanding user interaction patterns*

Figure 4.5 (left) summarizes interactions for the six participants for each study task. The cell values indicate the number of times an input modality was used to accomplish operations in Figure 4.1. For example, for a *find connections* operation, P1 used a combination of speech and touch (*find* query + double-tap) once and two speech queries (*find* + *find connections*) the second time (first and second row of the table respectively).

**Interaction Summary.** Of 181 total constructions, 92 (50.8%) instances contained spoken queries alone. Unimodal touch accounted for 55 (30.9%) and multimodal interaction where both speech and touch were used sequentially made up the remaining 33 (18.3%) constructions. No instances existed where modalities were used simultaneously (a myth of multimodal interaction [144]). However, all participants used more than one input modality at least once while performing the study tasks. Interaction patterns varied for the same task across participants (e.g., P1 performed task T1 using a multimodal pattern of speech+touch whereas P2 performed the same task using a single speech query). Similarly, individual participants' patterns varied as they performed similar tasks multiple times too. For instance, P6 performed task T5 using a series of spoken, touch, and multimodal interactions

but when performing a similar task T6, used only speech. The use of multiple modalities (individually and simultaneously) to accomplish tasks and the variable nature of interaction patterns across participants highlights the potential value of multimodal interfaces in accommodating such varying user preferences.

**NL interaction and interpretation.** Participants commended ORKO's natural language capabilities and felt it interpreted queries fairly well (Figure 4.5-right). Multiple participants were initially skeptical about NL input based on their previous experiences but were pleasantly surprised by the system's capabilities and the usefulness of speech input. For instance, P6, who reported that she frequently used applications like Siri and Notes stated "*I was surprised by the speech feature. I did not expect it to work as well as it did*". She also mentioned that speech not only worked well but actually improved her experience with visualization tools. She said "*having worked with many visualization programs before, having to go through and manually clicking is really annoying epecially when you have a ton of dropdowns. So I really like the speech feature, I know it's still in a rudimentary stage but it does a really good job*".

In terms of query interpretation, there were only seven instances where ORKO either did not respond or responded incorrectly to a query. Some of these were queries included operations that were not yet supported (e.g., layout change) while others queries had multiple values that were not separated by conjunctions. For example, for the query "*Show connections of Rooney McCarthy and Stones*" P5 expected the system to find connections of three players. The system, however, only showed connections of two players (Rooney and Stones), but still listing McCarthy within the ambiguity tooltip widget (Figure 4.4B). In such cases, participants typically thought of an alternative way to perform operations via touch or broke the query further into more explicit ones.

Although speech recognition was viewed favorably by participants in general, it was not perfect. On average, 16% of queries were missed or incorrectly translated from speech-to-text. The percentage was higher for some participants (e.g., 30% for P3) due to variations

in accent and speaking tone. Speech detection issues did cause some frustration among participants. For example, P3 stated "*It was a little frustrating when the system did not understand my voice or did not react at all to voice*". Ambiguity widgets did help for incorrectly detected player names, but only twice. Participants typically used the virtual keyboard to fix their utterances since it happened only occasionally. The more common case was the system failing to detect queries. In such cases, participants either repeated queries by tapping the 🎙 icon (Figure 4.2A) or used touch input to proceed with the task. For example, when the system did not detect a participant's (P4) *find connections* query, the participant simply double-tapped the node to see its connections.

These observations further motivate the need to study multimodal interaction for visualization systems, also illustrating how people can leverage DM to counterbalance issues such as speech detection.

**Deictic and follow-up queries.** For follow-up utterances, queries involving continue operations [68] were most common (e.g., adding new filters). Follow-up queries with references to new values (e.g., "*Filter to show Real Madrid*" > "*now Barcelona*" > "*now strikers*") were only used five times (thrice by P2 and twice by P5), all during a filtering operation. Instead, participants preferred to repeat entire queries (e.g., "*Filter to show Real Madrid*" > "*Filter to show Barcelona*" > "*Filter by strikers*") and often also repeated existing filters (e.g., "*Filter to show strikers*" > "*Show only strikers for England*"). Given this behavior, the query manipulation widgets were not frequently used.

Based on prior work that has shown a high preference for queries where DM (touch/pen) is followed by speech input [141, 214], we hypothesized that such queries would be a common pattern. Only two participants (P2, P6) uttered deictic queries that referred to nodes highlighted via touch interaction, however. Both queries were used in the context of highlighting connections within a group of nodes. During these queries, participants applied a *country* filter through the dropdown and then said "*Show the connections of these nodes*" (P2) and "*Highlight connections*" (P6). We suspected that the nature of the study tasks and

the participants' prior experience with visualization tools may have had an effect on the reduced usage of this pattern. Additionally, ORKO only supported a limited set of touch gestures, which may have limited the possibilities for deictic queries.

This preliminary study allowed us to conduct a basic assessment of ORKO and identify areas of improvement in terms of the interface design. It also served as a validation for the general idea that multimodal interactions can effectively support common visual network exploration scenarios. However, generalizing these findings and gaining a deeper understanding of the value of a multimodal interface deemed another study involving a larger set of users with more diverse experience levels in using visualization tools.

### 4.5 Study 2: Understanding User Interaction Preferences and Patterns

As a follow-up to the preliminary study described in the previous section, we leveraged ORKO to conduct a second qualitative study to better understand touch- and speech-based multimodal interactions with network visualizations. Specifically, we had three goals in mind when conducting this second study:

- **G1:** Understand if and why multimodal interaction is preferred over unimodal interaction.
- **G2:** Understand if and how prior experience of interacting using one input modality impacts subsequent multimodal interaction.
- **G3:** Identify different input and interaction patterns people use for common operations during visual network exploration.

While **G3** could be addressed with ORKO directly, addressing **G1** and **G2** required having comparative unimodal system(s). To this end, we derived two additional systems: ORKO-T (a touch-only variant of ORKO) and ORKO-S (a speech-only variant of ORKO) mimicking

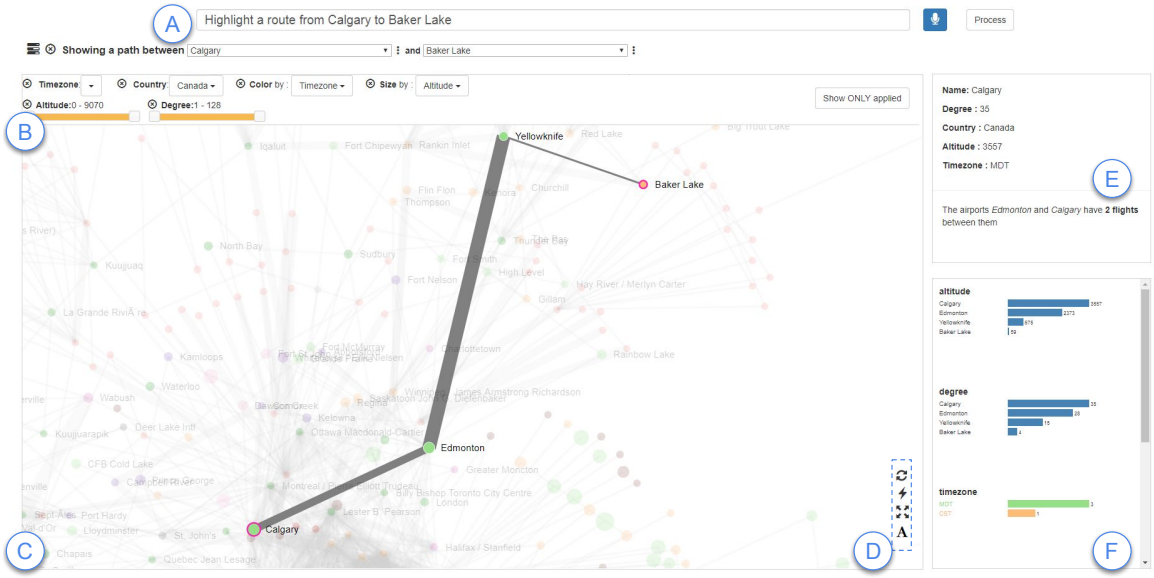



Figure 4.6: ORKO's updated interface. (A) Speech input and feedback, (B) filters and encodings, (C) visualization canvas, (D) quick-access icons, (E) details panel, and (F) Summary panel. In this case, the system shows a screenshot taken during an exploration of a US-Canada flight network dataset where a path between the Calgary and Baker Lake airports is highlighted.

ORKO's interface components. We also incorporated the feedback from the preliminary study to modify the interface design and NL interpretation capabilities of the original multimodal ORKO system. Below I first summarize these three interface variants and then detail the conducted study and its findings.

### 4.5.1 Study Interfaces

**Multimodal interface.** Figure 4.6 shows the final multimodal interface we used for this second study. Similar to the original system, the interface consists of the speech input and feedback row (Figure 4.6A), the visualization canvas (Figure 4.6C), quick-access icons (Figure 4.6D), and details and summary panels (Figure 4.6E,F). However, to enhance its visibility, we repositioned the filters and encoding row (Figure 4.6B) by placing it above the visualization canvas as opposed to below the canvas in the original system.

**Touch-only interface.** This interface functioned comparably to current DM-based network

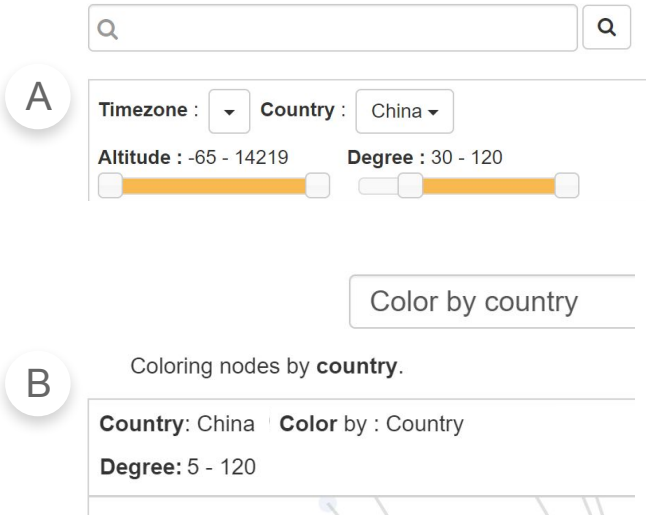



Figure 4.7: Screenshots from the unimodal study systems displaying the input, filters, and encodings rows in the (A) touch-only and (B) speech-only interface.

visualization tools, allowing people to select nodes and links and navigate the view with simple touch gestures, and adjust sliders and dropdown menus to filter points or change visual encodings. Additionally, we replaced the speech input and feedback row (Figure 4.6A) with a single input box that facilitated searching for nodes (Figure 4.7A). Entering search terms was supported through a virtual keyboard since this was a touch-only system.

**Speech-only interface.** This interface supported all operations listed in Table 4.1 through speech alone. Touch input was disabled throughout the interface. Visually, this interface had the same components as shown in Figure 4.6 with one key difference: since it was a voice-based system, participants could not directly manipulate the sliders and dropdown menus (Figure 4.7B). In other words, to adjust filters or apply visual encodings, participants always had to use voice commands (e.g., "*Change the timezone to CST*," "*Size by degree*").

In addition to the operations listed in Table 4.1, participants could also select nodes by tapping, drawing a lasso, or through speech (e.g., "*Select all airports in China*").

### 4.5.2 Study Details

*Methodology*

We conducted a qualitative study where two groups of participants first interacted with either the ORKO-S or ORKO-T interface followed by the multimodal ORKO interface. This allowed us to collect participant preferences and feedback to compare unimodal and multimodal interaction (**G1**). As a baseline to see how people interact with the multimodal interface when they encounter it for the first time (without having worked with the speech- or touch-only version), we also included a third group of participants who only interacted with the multimodal version of ORKO. Comparing the interactions of the first two groups with the third group allowed us to check if prior experience of using the system with one of the modalities resulted in any notable differences in terms of interaction behavior (**G2**).

We considered different study designs including a three condition (touch vs. speech vs. multimodal) within-subjects study and a study where participants used unimodal touch or speech input and multimodal input in counterbalanced orders. However, a within-subjects study with three conditions would last over three hours ($\sim$60 min. per condition) which was impractical. In the second alternative, having participants use the unimodal system after the multimodal system would not allow us to assess the priming effects of an individual modality. In other words, if participants interacted with the multimodal interface first, they would already have experienced all the supported interactions, not allowing for any assessment based on prior experience using individual modalities.

*Participants and Experimental Setup*

We recruited 18 participants (P1-P18), ages 18-66, five females and 13 males. 14 participants were native English speakers and the remaining four participants self-reported as being fluent English speakers. We sent recruitment emails to university mailing lists and recruited participants on a "first come first serve" basis. Participants who only interacted

with the multimodal system (P13-P18) received a $10 Amazon Gift Card as compensation whereas participants who interacted with both the unimodal and multimodal systems (P1-P12) received a $20 Amazon Gift Card as compensation.

In terms of their backgrounds, only eight participants said they had some prior experience of working with network visualization tools but 14/18 participants (except P2, P7, P8, P9) had some experience working with general visualization tools (e.g., Tableau, Excel). All participants had prior experience working with touch-based devices including phones, tablets, and laptops. All but two participants (P2, P14) said they used speech-based systems (e.g., Siri, Alexa) frequently. None of the participants had any prior experience working with touch- or speech-based visualization systems. All participants interacted with the system running on Google's Chrome browser on a 55" Microsoft Perceptive Pixels device. The screen was set to a resolution of 1920 x 1080 pixels.

*Dataset and Tasks*

As the primary focus of this study was understanding user interactions, we had to ensure that the network selected for the study encouraged interaction with the visualization and allowed us to cover a wide variety of network visualization tasks [109, 165, 222]. Given this high-level goal, we wanted to select a dataset where: (1) nodes had both numerical and categorical attributes so participants could filter and change visual encodings and (2) the connections had an intrinsic meaning so the tasks could emulate real-world scenarios. Additionally, to avoid differences due to domain knowledge, we wanted a dataset from a domain that was familiar to all participants (i.e. participants knew what the different attributes meant). With these criteria in mind, we selected two undirected flight networks as our datsets for the study.

The first dataset contained 551 airports (nodes) in the Asia-Pacific region and 2263 bidirectional flights between airports (links) whereas the second dataset contained 556 airports in United States and Canada and 2219 flights between those airports. Each airport in the

dataset had four attributes including its *altitude*, *country*, *timezone*, and a derived attribute indicating number of airports it was connected to (*degree*). Participants explored the Asia Pacific network in the unimodal condition, and the US-Canada network in the multimodal condition.

Participants performed six tasks (T1-T6) with each dataset. Similar to the first study (Section 4.4), the study tasks were generated based on existing network visualization task taxonomies [109, 152, 165] and included topology-level tasks, attribute-level tasks, browsing-tasks, and a group-level comparison task. For instance, in terms of Lee et al.'s taxonomy [109], T1 and T4 corresponded to topology-based tasks (finding paths and connections), while T2, T3, and T5 included a combination attribute-based tasks (filtering), browsing (following paths), and topology-based tasks. The tasks between the unimodal and multimodal conditions were designed such that they had a comparable level of difficulty. The order of tasks was randomized between conditions for each participant to prevent them from memorizing the operations they performed for a task.

To prevent participants from reading out the tasks as commands into the system as-is, we framed the tasks as a combination of scenario-based questions and jeopardy-style facts [62] that participants had to prove true/false. In other words, to "solve" a task, similar to the first study, participants had to interact with the system and get to a point where the visualization either proved or disproved the given statement or highlighted the required sub-graph. For instance, consider the task "*Suppose you want to fly from Fairbanks to Wales. Find a set of airports through which you must fly.*" To "solve" this task, participants could either find the path between the Fairbanks and Wales airports or they could manually, incrementally explore connections out of one of these airports until they reached the other. In either case, since there were multiple correct answers, participants had to visually highlight or show the list of airports (path) that one would need to travel through.

*Procedure*

Sessions lasted between 50-60 minutes for participants who only interacted with the multimodal system (P13-P18) and 125-135 minutes for participants who interacted with both the unimodal and multimodal systems (P1-P12). The study procedure was as follows:

**Consent and Background** (3-5 min): Participants signed a consent form and answered a questionnaire describing their background with visualization tools and touch- and speech-based applications.

**System Introduction** (3-5 min): The experimenter introduced the system, describing the user interface and supported operations. For speech interaction, participants were only informed about the operations the system supported and were not given a detailed vocabulary or list of possible commands for each operation. Instead, participants were encouraged to interact with the systems as naturally as possible, using any commands they felt were appropriate in the context of the given datasets.

**Practice** (3-5 min): Participants tested the touch and speech input until they felt comfortable using them. In this phase, participants interacted with a network of European soccer players described earlier (Section 4.2).

**Dataset and Task Introduction** (3-5 min): Participants were given a description about the flight network dataset along with the six tasks printed on a sheet of paper.

**Task Solving** (30 min): Participants interacted with the system to solve the six tasks. This phase was capped at 30 minutes. Participants were encouraged (but not mandated) to think aloud and communicate with the experimenter, particularly when they felt the system functioned unexpectedly. To avoid prompting interactions or disrupting the participants' workflow, the experimenter did not intervene during the session and only responded when participants initiated the discussion.

**Debrief** (10-15 min): Participants filled out a post-session questionnaire and engaged in an

interview describing their experience with the system.

Participants who performed tasks with both systems (P1-P12) were given a 15-minute break between the two sessions. After this break, except for the consent and background step, we followed the same procedure as with the first system. These participants were also asked to state and describe their preference between the unimodal and multimodal versions of ORKO during the debrief. We captured audio+video for all participant interactions with the system and audio recorded all interviews.

*Data Analysis*

Two experimenters individually reviewed both the audio and video data collected during the study to identify themes in interaction patterns and participant feedback. The resulting themes were then collectively discussed and iteratively refined into groups of observations using an affinity diagramming approach. This helped us characterize subjective feedback and participant behavior to qualitatively answer the initial questions driving the study (**G1**, **G2**). Furthermore, we also performed closed coding of the session videos to categorize the different types of interactions performed during the study (**G2**, **G3**). For this analysis, we used the operations in Table 4.1 as our set of pre-established codes. For each attempt at performing an operation, we noted if a participant used speech, touch, or a combination of the two. For instance, if a participant filtered nodes using a single spoken query (e.g., "*Show airports located at over 2100 feet*"), we would count this as one speech-only interaction. Alternatively, to filter, one could also directly adjust the slider (touch-only) or use a combination of the two modalities ("*Filter by altitude*" + drag slider). The intended operations were generally apparent due to the 'think aloud' protocol, the design of the tasks, and by the participant's reaction to system's interpretation of their interaction. The closed coding was also performed by two experimenters individually and conflicting observations or mismatches in counts were collectively resolved. We also used the session videos to determine the task completion times.

### 4.5.3 Results and Discussion

Addressing the study goals (**G1-3**), in this section, I describe our key findings corresponding to the preference for multimodal interaction (**G1**), the effect of priming users with one modality (**G2**), and the different input and interaction patterns employed by the participants to perform common network visualization operations (**G3**).

*Task and Interaction Overview*

11/12 participants (P1-P12) who interacted with the unimodal interface completed all six tasks, whereas one participant (P8) completed five. In terms of the correctness of task responses, four participants made errors: P3 and P7 answered one of the six tasks incorrectly and P8 and P9 responded incorrectly to two tasks. In the 18 sessions with the multimodal interface, all participants except P18 (who completed five tasks) completed all six tasks with only three participants (P3, P10, P15) making one error (each) while responding to the six study tasks. In terms of time, the task phase lasted, on average, 24 minutes with the touch-only interface, 23 minutes with the speech-only , and 21 minutes with the multimodal interface.

We recorded a total of 1052 interactions corresponding to the seven operations in Table 4.1 across the 18 participants and the two study interfaces. Figure 4.8 shows the distribution of 945/1052 interactions for six operations (*Find Nodes*, *Find Connections*, *Find Path*, *Filter*, *Change Visual Encodings*, *Navigate*) that are common across network visualization systems. We exclude *interface actions* from Figure 4.8 since these are generic tool-level operations (e.g., refreshing the canvas) and are not specific to network visualizations.

*Preference for multimodal interaction*

When asked which of the two systems they preferred, all 12 participants (P1-P12) who worked with both the unimodal and multimodal interfaces said that they preferred the mul-

**P1-P6: Unimodal Touch + Multimodal**

| | | Find nodes | | | Find connections | | | Find Path | | | Filter | | | Change Visual Encodings | | | Navigation | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | T | S | ST | T | S | ST | T | S | ST | T | S | ST | T | S | ST | T | S | ST |
| P1 | U | 9 | - | - | 12 | - | - | 5 | - | - | 6 | - | - | 4 | - | - | 7 | - | - |
| | M | | 4 | | | 7 | | | 2 | | 5 | 5 | | 2 | 1 | | 4 | | |
| P2 | U | 14 | - | - | 15 | - | - | 3 | - | - | 25 | - | - | | - | - | 3 | - | - |
| | M | | 1 | | 3 | | 8 | | 1 | | 3 | 7 | 1 | | | | 4 | | |
| P3 | U | 12 | - | - | 22 | - | - | 2 | - | - | 10 | - | - | | - | - | 3 | - | - |
| | M | | 1 | | | 10 | | | 1 | | 2 | 5 | 3 | | | | 1 | | |
| P4 | U | 7 | - | - | 14 | - | - | 4 | - | - | 7 | - | - | 6 | - | - | 2 | - | - |
| | M | | 7 | | 1 | 4 | 3 | | 2 | | 5 | 5 | 2 | | 1 | | 2 | 2 | |
| P5 | U | 12 | - | - | 19 | - | - | 4 | - | - | 16 | - | - | | - | - | 2 | - | - |
| | M | 1 | 4 | | 5 | 2 | | | 1 | | 3 | 6 | | | | | 6 | | |
| P6 | U | 7 | - | - | 11 | - | - | 5 | - | - | 5 | - | - | 1 | - | - | 2 | - | - |
| | M | | 3 | | 7 | | | | 2 | | 2 | 4 | 1 | | | | 3 | 3 | |
| **Total** | **U** | **61** | - | - | **93** | - | - | **23** | - | - | **69** | - | - | **11** | - | - | **19** | - | - |
| | **M** | **1** | **20** | | **16** | **23** | **11** | | **9** | | **20** | **32** | **7** | **2** | **2** | | **20** | **5** | |

**P7-P12: Unimodal Speech + Multimodal**

| | | Find nodes | | | Find connections | | | Find Path | | | Filter | | | Change Visual Encodings | | | Navigation | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | T | S | ST | T | S | ST | T | S | ST | T | S | ST | T | S | ST | T | S | ST |
| P7 | U | - | | - | - | 8 | - | - | 6 | - | - | 12 | - | - | 2 | - | - | 5 | - |
| | M | | 4 | | 2 | 4 | 1 | 2 | 1 | | 3 | 4 | | | | | 5 | | |
| P8 | U | - | 2 | - | - | 6 | - | - | 3 | - | - | 9 | - | - | | - | - | 2 | - |
| | M | | 2 | | 13 | | | | 2 | | 1 | 5 | 1 | | | | 5 | | |
| P9 | U | - | 4 | - | - | 10 | - | - | 4 | - | - | 6 | - | - | 1 | - | - | 11 | - |
| | M | | 7 | | 4 | 3 | 2 | | 1 | | 8 | | | | | | 4 | | |
| P10 | U | - | 1 | - | - | 9 | - | - | 3 | - | - | 4 | - | - | 1 | - | - | 11 | - |
| | M | 3 | 2 | | 6 | | 2 | 1 | | | | 3 | | | | | 1 | | |
| P11 | U | - | 3 | - | - | 8 | - | - | 3 | - | - | 2 | - | - | 3 | - | - | 6 | - |
| | M | | 2 | | 4 | 2 | 4 | | 1 | | 1 | 2 | | | 4 | | 2 | | |
| P12 | U | - | 7 | - | - | 7 | - | - | 3 | - | - | 10 | - | - | 3 | - | - | 5 | - |
| | M | | 5 | | 1 | 1 | 6 | 2 | 1 | 1 | 2 | 5 | 1 | | 4 | | 2 | | |
| **Total** | **U** | - | **17** | - | - | **48** | - | - | **22** | - | - | **43** | - | - | **10** | - | - | **40** | - |
| | **M** | **3** | **22** | | **30** | **10** | **15** | **5** | **6** | **1** | **15** | **19** | **2** | | **8** | | **19** | | |

**P13-P18: Multimodal Only**

| | Find nodes | | | Find connections | | | Find Path | | | Filter | | | Change Visual Encodings | | | Navigation | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | T | S | ST | T | S | ST | T | S | ST | T | S | ST | T | S | ST | T | S | ST |
| P13 | | 1 | 1 | 6 | | 5 | 1 | 4 | | 9 | | | | | | 4 | | |
| P14 | | 2 | | 7 | 9 | 7 | | 1 | | 9 | | | | | | 1 | | |
| P15 | | 2 | | 1 | | 5 | | | 1 | 7 | | | | | | 1 | | |
| P16 | | 7 | | 5 | 6 | 3 | | 1 | | 1 | 2 | | 1 | | | | | |
| P17 | | 5 | | | 11 | 1 | | 4 | 2 | | 5 | 1 | | 5 | | 1 | | |
| P18 | | 3 | 2 | 2 | 2 | 1 | | 1 | | 1 | 4 | 1 | | 1 | | 2 | 1 | |
| **Total** | | **20** | **3** | **21** | **28** | **22** | **1** | **11** | **3** | **27** | **11** | **2** | **1** | **6** | | **9** | **1** | |

Figure 4.8: Distribution of 945 interactions used to perform six common network visualization operations during the study. **U**: Unimodal interface, **M**: Multimodal interface, **S**: Speech, **T**: Touch, **ST**: Multimodal interactions. A '-' indicates that a modality was not supported in a condition or that participants were not assigned to a condition.

timodal system over the unimodal system. This was not surprising given similar findings in earlier studies [141, 147] and the simple fact that the multimodal system provided all capabilities that the unimodal system did. Hence, we were more interested in understanding what aspects of the combination of speech- and touch-based interaction with the system led participants to prefer it (**G1**).

One hypothesis for why participants preferred the multimodal system, developed after reviewing their interaction counts in Figure 4.8, was that in some cases, multimodal input allowed them to perform tasks with fewer interactions. However, tasks could be performed using multiple strategies through varied operations, each resulting in a different number of steps. Thus, basing the preference on interaction counts alone would be unjustified because we did not control for which strategy or operations participants used during a task. Instead, we coupled the participants' verbal comments and our observations of their interactions to identify three factors listed below that we believe led to their preference for multimodal interaction.

**Freedom of expression.** Out of the 945 interactions, 489 were performed in the context of the mulitmodal interface (P1-P6 [M], P7-P12 [M], and P13-P18 in Figure 4.8). Among these, 233 (48%) used unimodal speech input, 190 (39%) involved unimodal touch input, and 66 (13%) used both modalities sequentially. Although only 13% of interactions involved sequential use of modalities, all participants used both modalities (individually or together) during at least three out of the six tasks in a session.

Interaction patterns also varied across participants for the same operation. For instance, observing the interactions for P13-P18 in Figure 4.8, we can notice that P17 and P18 primarily used speech (individually or sequentially with touch) for filtering. On the other hand, P13-P16 primarily used touch to filter nodes. Interaction patterns varied even for individual participants across tasks. For instance, while performing the first task, P1 issued a unimodal speech command to find connections. However, during the second task, to find connections, he used speech and touch sequentially.

Participants also verbally commented on their preference for multimodal input over unimodal input in their post-session interviews. Participants said that having multiple modes of input gave them more freedom to try different ways to perform a task. For instance, highlighting the use of speech and touch for different operations, P8 said "*The combination is certainly better. Voice is great when I was asking questions or finding something I couldn't see. Touch let me directly interact.*" Similarly, P2 said "*It (multimodal input) felt more natural. I really liked that I could choose what I wanted to do with my hands and what I wanted to say.*" Stating multimodal interaction was more natural, P10 also said "*Working with this second system felt more natural. If I wanted to filter by something I could just say that but when I'd see something interesting I could touch it without having to say something and wait for the system to process it.*"

The varied interaction patterns within and across participants coupled with the subjective comments highlight how the multimodal interface provided more freedom of expression, allowing participants to interact based on the task context or personal preferences.

**Complementary nature of modalities.** A popular hypothesis about multimodal interaction is that it allows users to offset the weaknesses of one modality with the strengths of another [36, 141]. Along these lines, when describing their experience with the multimodal system, 12 participants (P2, P4-6, P9-11, P13, P15-18) explicitly commented on the complementary nature of touch and speech and how it was a key advantage of multimodal input.

Participants found the ability to correct speech with touch very useful, with some participants even stating that the combination is vital to make effective use of speech. For example, P17 said "*I liked that I could correct with touch. Because it's not always going to be perfect right. Like the smart assistant on the phone sometimes gets the wrong thing but doesn't let me correct and just goes okay.*" Talking about cases when the system populated the right filtering attribute but did not detect the right value, P2 said "*the system would bring the correct dropdown even if it didn't get the value right and then I could simply*

*correct that.*" In addition to correcting speech recognition and ambiguity errors with touch, participants also appreciated that they could leverage touch to modify existing queries. For instance, referring to the query manipulation dropdowns in the speech input feedback row (Figure 4.6A), P18 said "*I liked that it allowed me to modify my command without having to say it again.*"

On the other hand, participants also used speech when they either forgot a gesture or were unable to perform an operation using touch. P13, for instance, said "*I used voice when I didn't know how to do it with touch.*" highlighting that speech can aid in overcoming memorability issues associated with touch gestures. Similarly, five participants (P4, P6, P13, P15, P18) used speech commands to navigate the view (i.e. zoom and pan) when they were in a dense region of the network and were unable to pinch or drag without touching the nodes on the canvas.

In addition to affirming the benefits of complementary interactions, such comments also highlight the value in further exploring elements like ambiguity widgets [62] to help users resolve challenges with speech and query manipulation widgets that help users modify existing utterances.

**Integrated interaction experience.** Another theme among the participants' comments regarding the advantages of multimodal interaction alluded to the notion of *fluidity* as characterized by Elmqvist et al. [52]. For instance, referring to the ability of being able to apply filters while interacting with nodes on the view, P16 said "*Generally I prefer touch but here speech was good because then I don't have to look through filters and I can just say it while moving points.*" Although he was initially skeptical about multimodal input, during his interview, P10 said "*It was somehow less complex even though more interactions were added.*" suggesting that the combination of modalities helped reduce the overall cognitive load. The comparatively fluid nature of multimodal input also led to participants perceiving themselves as being faster with the task even though the overall task completion times were comparable across the study interfaces. For instance, P1 said "*Having the combination was*

*a lot easier to work with. Instead of having to find nodes and then highlight connections, I could do it in one command and then continue to interact with the graph.*"

Motivated by such comments, we further reviewed the session videos to better understand what specifically about multimodal input evoked the feeling of fluidity. Based on our review, we attribute the fluidity of interaction in the multimodal interface to speech and touch facilitating *integrated interactions* [208] that are defined as "*interactions where a person's hands, tools, actions, interactions, visual response, and feedback are in situ where the data is visualized. That is, to effect an interaction, a person's attention is not drawn away from the visual representations of data in which they are interested.*" Examples of integrated interactions during the study included applying a filter using speech while dragging nodes—not having to take eyes off the nodes in focus, or the ability to find nodes using speech without having to divert attention in order to type on a virtual keyboard that occluded the underlying visualization, among others. While these are seemingly straightforward interactions in isolation, they illustrate that the modalities together allowed participants to stay in the flow of their analysis rather than divert their attention to other user interface elements.

*Effects of priming users with speech or touch*

One of our study goals also was to observe if participants interacted differently with the multimodal system when they had prior experience with one of the two modalities (**G2**). More specifically, we were curious if participants would continue to use the same modality and not use multimodal input? Would participants rely more heavily on the modality they first experienced? Would interaction patterns for these participants notably differ from those who only interact with the multimodal system? We were interested in these questions as the findings could challenge the need for multimodal input altogether. For instance, one possible outcome was that participants who interacted with the unimodal touch system (P1-P6) would continue to use only touch in the multimodal setting and similarly P7-P12 would

use only speech input in the multimodal setting. Such a finding would suggest that people resort to what they know, refraining from learning new ways to interact with a system. Alternatively, it could also imply that adding input modalities may have limited (or no) benefits when users know how to work with a system using a specific modality.

We observed that participants who had prior experience working with the unimodal system (P1-P12) interacted with the multimodal system comparably to participants (P13-P18) who worked only with the multimodal system. When we explicitly asked participants if their experience with the first system affected their behavior, participants said that they used both modalities subconsciously and did not think about it until we asked them to reflect on it. For instance, P12 said "*Now that I think of it, not consciously but I did use speech to mostly narrow down to a subset and then touch to do more detailed tasks.*" In fact, perhaps the single most important aspect that decided which modality would be used was the operation being performed. For instance, consider participants P1-P6 and the *Find Path* operation in Figure 4.8. In this case, when interacting with the multimodal interface, all participants switched to using only speech commands even though they had all previously performed the operation using touch. Furthermore, this interaction pattern of primarily using speech for finding paths is comparable to the participants in the other conditions (P7-P18), suggesting a general mapping between the modality and task. Combined with the comments from the previous section, these observations suggest that people naturally adapted to using a new modality that was more suited for an operation even if they were experienced at performing the same operation with a different mode of input.

*Operations and interaction patterns*

Figure 4.9 summarizes the number of participants who used speech-only ( 🗣 ), touch-only ( 👆 ), or combined speech and touch input ( 🗣 + 👆 ) for performing common network visualization operations (**G3**). Note that a single participant may have performed more than one type of interaction for an operation (e.g., as shown in Figure 4.8, for finding paths,

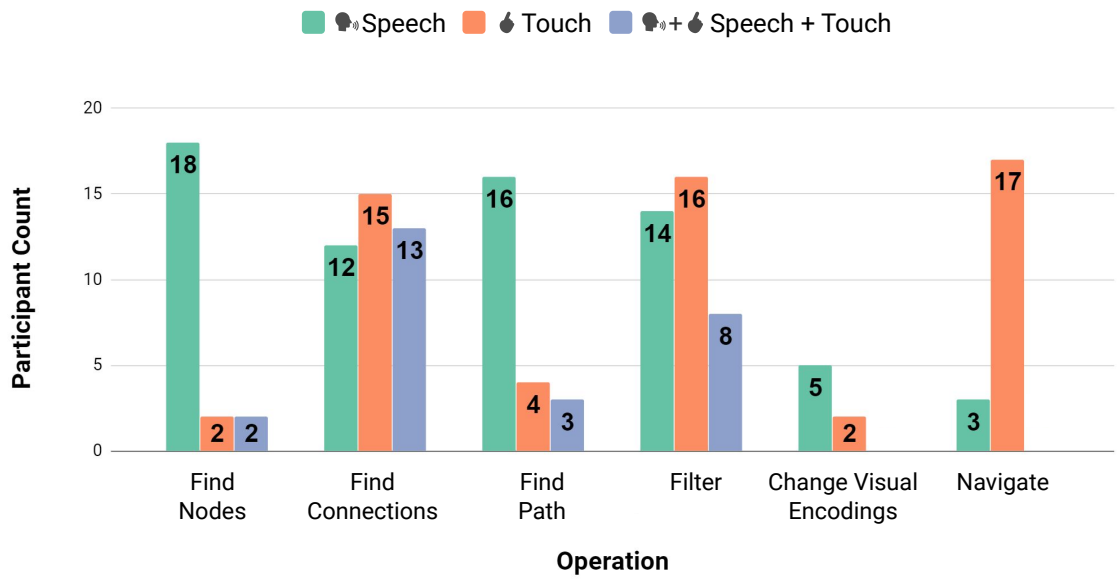



Figure 4.9: Number of participants using different modes of input in the multimodal interface for each type of operation. Navigation was primarily performed using touch, whereas finding nodes and paths was largely performed through speech. Other operations had more variety in input patterns.

P13 used both speech-only and touch-only interactions). Our goal here is not to suggest one "best" input modality or interaction for an operation but rather to highlight the variety in patterns so future system designers can make more informed interaction design decisions. For this analysis, we only considered the 489 interactions with the multimodal interface when listing the actions since the unimodal interfaces did not give participants the option to choose their preferred style of input.

At a first glance, both Figures 4.8 and 4.9 suggest that participants largely performed operations using a single modality, infrequently *combining* modalities. However, it is important to also note that participants switched between modalities for different operations, using both modalities individually or together at some point during all sessions. Affirming to the myths of multimodal interaction [144], this switching between modalities highlights that the value of the multimodal interaction does not only come from modalities being combined but also stems from the availability of different modalities to perform varied operations amidst a task.

## 4.6 Conclusion

Through ORKO's design and feedback from the two user studies, we present an example of how multimodal visualization interfaces can support fundamental visual analysis operations while facilitating fluid and expressive interactions **[RG2]**. Although we only worked with networks, as the first attempt at supporting *speech and touch* input for visualizations, the work described in this chapter set into motion a new line of multimodal interface research.

Specifically, designing and evaluating ORKO helped verify the fundamental hypothesis that people prefer multimodal visualization interfaces over their unimodal counterparts and more importantly, helped understand factors that contribute to this preference (freedom of expression, complementarity of modalities, support for integrated interactions). Furthermore, from a multimodal system design and implementation standpoint, this work also helped understand the types of utterances people may issue (e.g., sequential, standalone, follow-up) and how input from different modalities can be synchronized (e.g., order of input, time lag between inputs) to interpret multimodal interactions in the context of visualization tools.

# CHAPTER 5
# GENERALIZING MULTIMODAL INTERACTION DESIGN FOR INFORMATION VISUALIZATION TOOLS

With ORKO, we focused on network visualizations and correspondingly, the interactions that we designed involved only network-specific operations (e.g., finding nodes, finding paths). Expanding beyond networks and showing that multimodal interfaces can support general visual analysis scenarios, in this chapter, I detail a second project that investigates the idea of multimodal interactions with visualizations more broadly (i.e., considering a larger set of visualizations, operations, and modality combinations)[1].

Through a survey of existing visualization interfaces and interaction task taxonomies, we first identified core visual analysis operations and chart types to design multimodal interactions for. Following a set of design principles that focus on fluidity, expressivity, and consistency in interactions, we then developed a set of interactions incorporating pen, touch, and/or speech to support core categories of visual data analysis operations (e.g., map visual encodings, filter, details-on-demand). To generalize the process of designing multimodal interactions with visualization tools, we also compiled a set of core concepts (operations, parameters, targets, instruments) that can help design, describe, and compare multimodal interactions in future systems. Furthermore, to illustrate how these interactions can collectively support visual data exploration, we implemented them within a prototype tablet-based multimodal visualization interface, INCHORUS. Finally, based on a user study with INCHORUS, we demonstrated how the multimodal interface not only supported core visual analysis operations with different chart types, but also enabled an interaction experience that was fluid, expressive, and consistent **[RG2]**.

[1]The content of this chapter is based on work previously published in ACM CHI 2020 [183].

## 5.1 Systematic Design of Multimodal Interactions for Data Visualization Tools

We surveyed 18 visualization systems [15, 23, 50, 57, 83, 84, 89, 110, 158, 159, 160, 161, 170, 189, 191, 196, 208, 226] (including ORKO [189]) to identify tasks and visualizations to consider as part of our design. We included a system in our survey if it (1) involved interactions using one or more of pen, touch, or speech input with a single device and user and (2) focused on general visual data exploration and analysis (as opposed to systems that placed higher emphasis on externalizing users' thoughts [95, 155] or authoring illustrative visualizations [94, 108, 217]).

Through the survey, we identified five core categories of operations (Table 5.1) that were supported by most systems. Furthermore, given their frequent occurrence in the surveyed systems and prevalence in common visualization tools, we decided to focus on *histograms*, *bar charts* (including grouped and stacked bar charts), *line charts*, *scatterplots*, and *parallel coordinates plots* as the initial set of visualization types.

### 5.1.1 Conceptualizing Multimodal Interaction Design

To consistently design interactions, we needed a standardized nomenclature to describe and compare alternative interactions. Correspondingly, we reviewed the terminology and description of interactions in the surveyed visualization systems' papers. However, since most current systems were optimized for specific visualizations (e.g., networks [170], bar charts [50], scatterplots [159]) or form-factors (e.g., tablet [83], whiteboard [110]), there was no common language that let us consistently design and discuss possible interactions. Thus, based on our survey and a review of prior work in the more general space of post-WIMP (e.g., [16, 200]) and multimodal interfaces (e.g., [38, 124, 142, 184]), we identified a set of core concepts that could help us (and future system designers) systematically design and reason about interactions in the context of multimodal visualization systems.

We propose thinking of interactions with multimodal visualization systems in terms

Table 5.1: Proposed multimodal interactions for low-level operations during visual analysis. Operations categories are O1: Bind/unbind visual encodings, O2: Modify axes, O3: Filter, O4: Get details, and O5: Change chart type. Unless explicitly specified as an indirect instrument (II), all instruments are direct instruments (DI). An asterisk (*) indicates a parameter is optional. The rightmost column displays modalities (T: Touch, P: Pen, S: Speech) used in an interaction pattern.

| | Operation | Parameters (P) | Targets (T) | Instruments (DI or II) | Sample Keywords (K) | Interaction Patterns | Examples | Modalities | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | T | P | S |
| O1 | Bind attribute to an encoding | Attribute | X/Y axes, Color legend | Axis title region, Color legend title, Attribute pills II | | *I1:* Drag II to DI | Drag worldwide gross to X-axis title | ✓ | | |
| | | | | | | *I2:* Point on DI and tap II | Point on X-axis title and tap major genre | ✓ | | |
| | | | | | | *I3:* Point on DI and write P | Point on Y-axis title and write running time | ✓ | ✓ | |
| | | | | | | *I4:* Point on DI and speak P | Point on color legend title + "*Content Rating*" | ✓ | | ✓ |
| | | | | | | *I5:* Speak < T , P > | "*Show creative type on the x-axis*" | | | ✓ |
| | Bind multiple attributes to an encoding | Attributes | X/Y axes | Axis title region, Attribute pills II | *Add, include, group by, split by* | *I6:* Point on DI and speak < K , P > | Point on Y-axis title + "*Add budget and gross*" | ✓ | | ✓ |
| | | | | | | *I7:* Speak < T , K , P > | "*Group x-axis by content rating*" | | | ✓ |
| | | | | | | w/ modifier active: | w/ modifier active: | | | |
| | | | | | | *I8:* Drag II to DI | Drag major genre to X-axis title | ✓ | | |
| | | | | | | *I9:* Point on DI and tap II | Point on Y-axis title and tap IMDB Rating | ✓ | | |
| | | | | | | *I10:* Point on DI and write P | Point on X-axis title and write Running Time | ✓ | ✓ | |
| | Remove attributes from an encoding | Attributes | X/Y axes, Color legend | Axis title region, Color legend title | *Remove, unbind, clear, delete* | *I11:* Erase P* from DI | Erase X-axis title to remove attributes mapped to X-axis | | ✓ | |
| | | | | | | *I12:* Point on DI and speak < K , P* > | Point on color legend title + "*Clear*" | ✓ | | ✓ |
| | | | | | | *I13:* Speak < T , K , P* > | "*Remove budget from the y-axis*" | | | ✓ |
| O2 | Change data aggregation level | Aggregation level | X/Y axes | Axis title region | | *I14:* Point on DI and write P | Point on Y-axis title and write *sum* | ✓ | ✓ | |
| | | | | | | *I15:* Point on DI and speak P | Point on Y-axis title + "*max*" | ✓ | | ✓ |
| | | | | | | *I16:* Speak < T , P > | "*Show the average values on y*" | | | ✓ |
| | Sort | Sort order, Attribute | X/Y axes | Axis title region | *Order, sort, arrange, reorder* | *I17:* Swipe on DI, direction determines P | Swipe downwards on Y-axis title to sort in descending order by Y-axis attribute values | ✓ | | |
| | | | | | | *I18:* Point on DI and speak < K , P* > | Point on Y-axis title + "*Sort*" to sort in ascending order (default system value) by Y-axis attribute values | ✓ | | ✓ |
| | | | | | | *I19:* Speak < T , K , P* > | "*Arrange x-axis in decreasing order*" | | | ✓ |
| O3 | Filter marks | | Marks | Marks | *Filter, exclude others, remove, keep only* | *I20:* (Select +) Erase DI | Erase bar in a bar chart to remove a data category | | ✓ | |
| | | | | | | *I21:* w/ DI selected, speak < K > | Select points in a scatterplot + "*remove others*" to filter unselected points | | ✓ | ✓ |
| | Filter by criteria | Attribute values | Marks | Color legend rows | *Filter, exclude others, remove, keep only* | *I22:* Erase P from DI | Erase *Action* row from color legend to remove action movies | | ✓ | |
| | | | | | | *I23:* Speak < K , P > | "*Exclude horror movies*" | | | ✓ |
| O4 | Get mark details | | Marks | Marks | | *I24:* Long press DI | Long press on a bar in a bar chart to see its value | ✓ | ✓ | |
| | Get mark details by value | Attribute values | Marks | Axis scales | | *I25:* Drag along DI | Drag along X-axis scale in a horizontal line chart to see values for a specific timestamp | ✓ | | |
| O5 | Change chart type | Char type | Canvas | | | *I26:* Speak < P > | "*Switch to a line chart*" | | | ✓ |

of the following four concepts. People interact with visualization systems through a set of one or more low-level ***operations*** (e.g., binding an attribute to an encoding, sorting) to accomplish their high-level tasks (e.g., answering data-driven questions, creating specific visualizations). These operations typically require ***parameters*** (e.g., sorting order, attributes, encodings) and operate on one or more ***targets*** (e.g., selected marks, axis, canvas). Finally, operations are mediated through ***instruments*** in the interface (e.g., marks, axes scales). These instruments can be *direct* (i.e., when the target itself mediates an operation) or *indirect* (i.e., when an operation is performed on a target through a separate instrument).

With these general user interface concepts in mind, we investigated possible interactions for the operations identified through the survey. To ensure the resulting interactions were generalizable, we only considered basic elements (e.g., axes, marks, attribute pills) present in most visualization tools as our instruments. We then examined both interactions demonstrated in previous systems (e.g., writing an aggregation function name to change the data aggregation level [83], dragging along an axis to sort [50]) as well as novel interactions that were potentially more fluid and consistent (e.g., pointing on an axis with a finger and speaking an attribute name to specify mappings, using the pen's eraser to filter).

Table 5.1 lists the ten low-level operations and the corresponding set of interactions (*I1-I26*) derived after a series of iterations, along with examples and input modalities. We initially considered *selection* and *zoom/pan* as two additional core operations. However, since these are low-level interface actions and sometimes a precursor to other operations (e.g., filtering a set of marks may require selecting them first), we decided not to include them as standalone operations. In our interaction set, selections can be performed in four ways: 1) tapping with pen/finger directly on a mark, 2) tapping with pen/finger on a legend item to select marks by categories [72], 3) dragging the pen on an axis scale to select marks based on data values, and 4) drawing a free-form lasso with the pen on the chart area. Additionally, zoom and pan are supported through the standard two-finger pinch and single

finger drag gestures on the chart area, respectively.

### 5.1.2 Design Principles

In this section, we describe five underlying principles we had when designing our multimodal interactions. We compiled these principles based on the surveyed papers as well as design guidelines from prior work advocating for post-WIMP visualization interfaces [52, 107, 154].

***DP1.*** *Maintain interaction consistency across visualizations*

To support consistency, we prioritize globally functional interactions (i.e., ones that work across visualization types) over locally optimal interactions (i.e., ones that are specific to a type of visualization). A pattern from Table 5.1 exemplifying this principle is *I15: Drag along axis scales* to see mark details. Previous systems have inconsistently used this interaction to sort bars in a bar chart [50] and select points in a scatterplot [159]. However, since these are both locally optimal interactions, we use dragging along an axis scale to display mark details which is a common operation across visualizations.

We note that some operations are specific to certain visualization types. For example, sorting an axis is meaningful to bar charts and parallel coordinate plots but not to scatterplots and line charts. Thus, *I17:swiping on the axes* only works for the appropriate visualizations and has no effect in others. We also reserve the swipe interaction for sorting and do not employ it for a different operation elsewhere.

***DP2.*** *Minimize indirection in interactions*

Aligned with the guidelines for fluid interaction [52], we try to enable interactions with direct instruments (e.g., marks, axes), avoiding external controls and indirect instruments that are separated from the view. For instance, to filter marks, people can use *I20: erase marks* directly instead of adjusting external widgets like sliders or dropdown menus. Or

to see the details of a mark, one can use *I24: long press on marks* instead of indirectly requesting for details through voice.

Another implication of this design principle is that if an operation is inherently indirect, we offload it to speech since it is also, by nature, indirect. Examples of such indirect operations include changing the visualization type and filtering based on attributes that are not encoded in the current view (e.g., filtering points by *imdb rating* in a scatterplot of *production budget* by *worldwide gross*).

***DP3.*** *Leverage simple and familiar gestures*

Simple gestures that are familiar to users are easier to learn and subjectively preferred for interacting with pen and touch-based visualization systems [50, 83, 161, 208]. To maintain simplicity and promote familiarity, we avoid devising specialized gestures for individual operations. In fact, as illustrated by the patterns in Table 5.1, all our pen and touch interactions only involve common gestures including tap, point (hold), swipe, and drag. Particularly for cases where one gesture could be mapped to multiple operations, instead of introducing an alternative gesture for one of the operations, we apply a division of labor tactic [76] and offload the interaction to a different modality. For example, due to their ubiquity across devices and applications, we reserve touch-based pinch and drag gestures for zoom and pan, respectively. However, dragging on the chart area is also an intuitive way to perform selection (e.g., by drawing lassos [159]), which is another important action during visual analysis [221]. To resolve this conflict, we leverage a second modality and allow people to draw selection lassos by dragging on the chart area using the pen.

***DP4.*** *Avoid explicit interaction modes*

Interaction modes enable an interface to support a wider range of operations. However, constantly switching between modes (e.g., inking vs. gesture) for the same type of input (e.g., pen) can be disruptive to the users' workflow and are known to be a common source of

errors [111, 153]. To avoid explicit interaction modes, we assign semantically meaningful actions to different modalities (e.g., touch for pan/zoom, pen for selection) and leverage a combination of modalities to support advanced variations of simpler actions (e.g., using bimanual pen and touch input to compound selections).

***DP5.*** *Strive for synergy not equivalence*

A common myth about multimodal interaction is that all modes of input can support all operations [144]. Instead of designing specialized interactions (e.g., highly customized and complex gestures or widgets) to ensure equivalence, we support equivalence between modalities only if the equivalence is inherently meaningful. For instance, we allow binding attributes to encodings using all three modalities (*I1-I10* in Table 5.1). On the other hand, because there is no direct interaction (**DP2**) to filter marks based on an attribute that is not encoded in the view using pen or touch, we only allow this via speech (e.g., saying "*Remove movies with an imdb rating under 8*" when the system shows a scatterplot of *budget* and *gross*).

Furthermore, we also leverage *complementarity-based interactions* [124], where different chunks of information are provided by different modalities and subsequently merged together to perform an operation. In addition to help accomplish **DP2** and **DP3**, complementarity can also facilitate faster interactions and reduce the complexity of speech commands, ultimately improving both the user and system performance [19, 124, 203]. For instance, with touch alone, binding multiple attributes to an axis requires multiple interactions with control panel widgets such as dropdown menus [50]. Alternatively, during *I6:pointing on the axis and speaking*, the axis (target) is implicitly determined by touch whereas speech allows specifying multiple attributes (parameters) as part of the same action. Similarly, to support negative filters, instead of providing additional keep-only button or menu item in touch-only systems (e.g., [50, 159]), a system with *I21:select-and-speak* can let people select points by drawing a lasso with a pen and saying "*exclude others*." In

this case, the target (marks) is specified through the pen while speech provides the operation (via "*exclude*") and further modifies the target (via "*others*").

Note that the concepts and interactions in Table 5.1 are by no means an exhaustive or definitive set. They are only one sample set of interactions we designed with **DP1-5** and basic elements of visualization systems in mind. In fact, depending on a system's interface, some of these interactions may not even be applicable. For instance, if a system does not explicitly list attributes as interactive widgets, the *I1:drag-and-drop* and *I2:point-and-tap* interactions involving attribute pills to bind attributes to encodings cannot be used. However, the *I3:point-and-write* and *I4:point-and-speak* interactions for the same operation remain valid since they rely on the X/Y axes of the chart itself.

## 5.2 INCHORUS: Visual Data Exploration through Multimodal Interaction on Tablet Devices

To demonstrate how the proposed interactions collectively support visual data exploration, we employed them in a tablet-based visualization system prototype, INCHORUS. Our choice to focus on tablets was driven by two characteristics of existing DM-only visualization systems on tablets.

First, the majority of prior research about data visualization on tablets [15, 50, 84, 158, 159] have focused on a single visualization type, optimizing interactions for that chart type. This local optimization could result in a *globally inconsistent interaction experience* when multiple types of visualizations are included as part of one system. For example, prior systems have used the gesture of dragging a finger along the axis to sort a bar chart [50] and select points in a scatterplot [159]. However, when both bar charts and scatterplots are supported by the same system, the gesture of dragging along an axis causes a conflict, resulting in inconsistent functionality across visualizations [161]. Resolving such inconsistencies often requires system designers to introduce specialized gestures such as holding on

the axis of a bar chart to enter a transient "sort mode" in which one can swipe to sort [161]. Such subtle differences in gestures can be difficult to remember and may lead to errors while performing tasks, however.

Second, when depending only on pen and/or touch, systems face *increased reliance on menus and widgets* as the number and complexity of operations grow. For example, to filter, users have to select visual marks and tap a delete/keep-only button or adjust sliders and dropdown menus in control panels [83, 159, 161]. Such indirect interactions with interface elements external to the objects of interest (e.g., marks) can divert the users' attention which may prove disruptive to their workflow [50]. Additionally, given the space constraints of tablets, control panels can occlude the visualization and limit the screen space available for the visualization itself.

We show how the proposed interactions in Table 5.1 help overcome these challenges by complementing the directness and precision of pen and touch with the freedom of expression afforded by speech.

### 5.2.1 Interface Overview

Figure 5.1 illustrates INCHORUS's user interface its different components. As stated earlier, to design interactions that were generalizable, we focused on a minimalistic interface with basic elements (e.g., axes, marks, legend, attribute pills) present in most visualization tools. Additionally, similar to previous pen and touch systems (e.g., [160, 217]), we also added a modifier button (Figure 5.1C) that serves two purposes: 1) it allows utilizing bimanual input, which can help avoid explicit mode switches during pen- or touch-only interactions (**DP4**), and 2) it serves as a "record" button to input voice commands. INCHORUS uses a "push-to-talk" technique: it records speech while a finger is on the modifier button, the X/Y axis title regions, or the color legend title, and executes the recognized command once the finger is lifted.

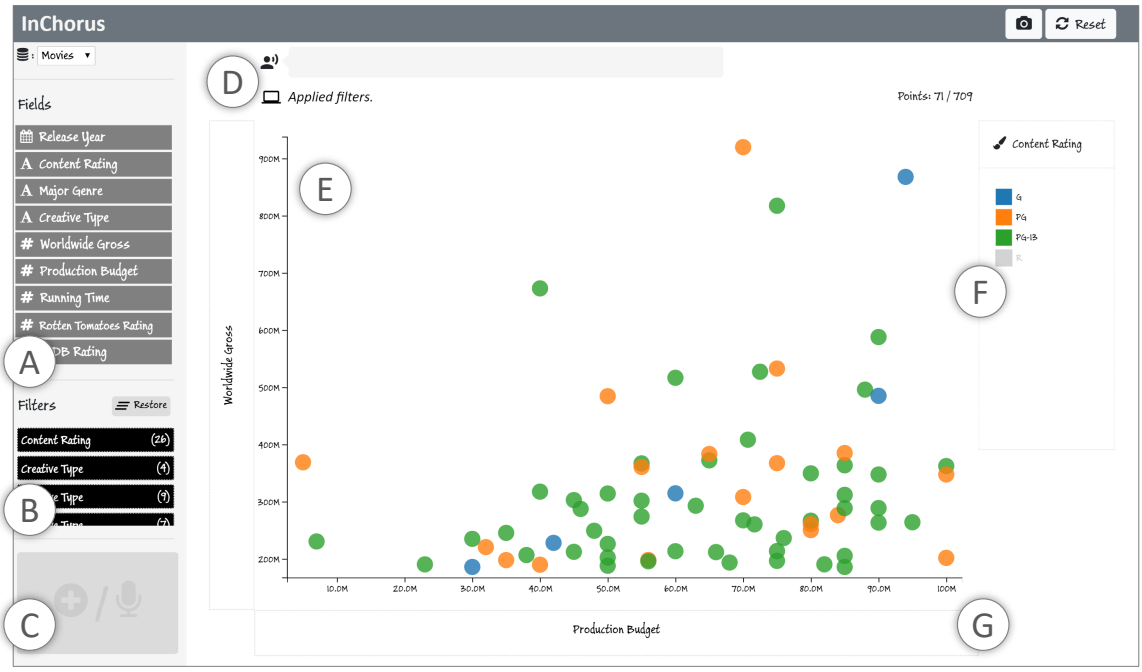



Figure 5.1: INCHORUS interface components. (A) Attribute pills, (B) Active filters, (C) Modifier button, (D) Speech command display and system feedback row, (E) Chart canvas with marks (in this case, circles), (F) Color legend area, and (G) Axis scale and title area.


Figure 5.2: Tapping an attribute while pointing on the x-axis title region binds the data attribute to the x-axis.

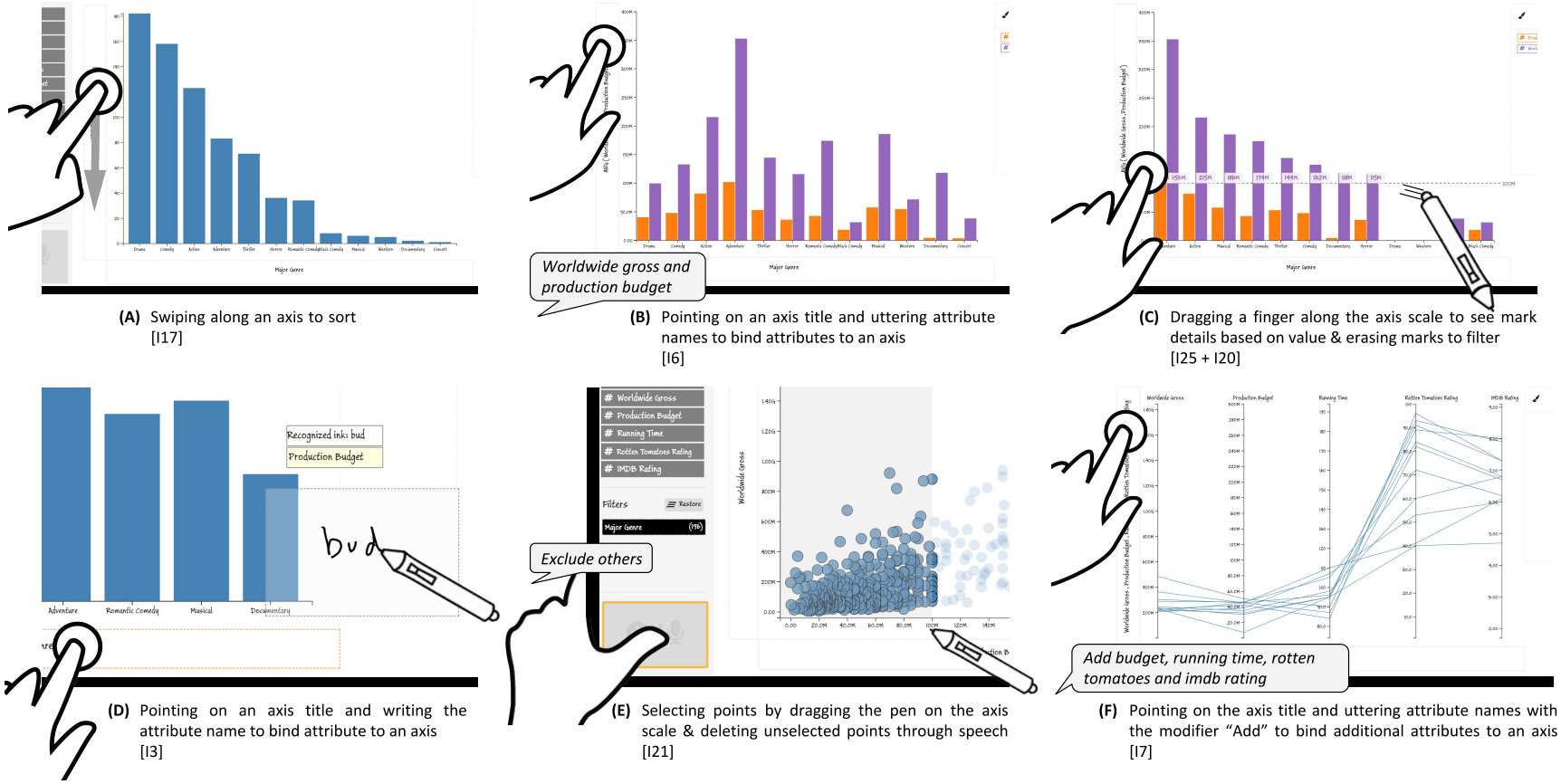
### 5.2.2 Interacting with INCHORUS

Consider the following scenario as a walk-through of the visual data exploration experience when working with InChorus. Imagine that Joe, an analyst at a movie production house, wants to identify movie characteristics his company should focus on for their next investment. To investigate previously released movies, Joe loads a dataset of 709 movies into InChorus. The dataset contains nine attributes for each movie, including *Release Year* (📅 temporal), *Worldwide Gross* (**#** quantitative), and *Major Genre* (**A** categorical), among others; all nine attributes are shown on the left panel (Figure 5.1A).

**Identifying key genres.** To get an overview of values for each attribute, Joe *taps* on individual attributes in the side panel *while pointing* (i.e., holding down a finger) on the X-axis title region (Figure 5.2). As he taps through the attributes, InChorus displays univariate summary visualizations (histograms, bar charts, and line charts) based on the attribute's data type.

To see the popularity of different genres, Joe binds the *Major Genre* attribute to the X-axis, creating a bar chart. Joe decides to sort the bars by the number of movies so he can identify more popular genres faster. As he *swipes downwards* on the Y-axis, InChorus sorts the genres in descending order by count (Figure 5.3A). To get a sense of the typical return on investment for different genres, Joe adds the gross values to the view by *pointing* on the Y-axis and *saying* "*Worldwide gross and production budget*." This updates the view to a grouped bar chart displaying the average gross and budget for different genres (Figure 5.3B). Now, to look at the highest grossing genres instead of the most popular ones, Joe again wants to sort the view. However, because two attributes are mapped to the Y-axis, instead of swiping, Joe now *points* on the Y-axis and *says* "*Sort by worldwide gross in descending order*" to clearly express his intent.

To see values corresponding to the bars, Joe *drags* his finger along the Y-axis scale. As he *drags* his finger along the Y-axis scale, InChorus displays a horizontal ruler highlighting the value corresponding to his finger's position and shows details of bars that intersect with the ruler. Inspecting the chart, Joe decides to only focus on high grossing genres. He




**(A)** Swiping along an axis to sort [I17]

**(B)** Pointing on an axis title and uttering attribute names to bind attributes to an axis [I6]

**(C)** Dragging a finger along the axis scale to see mark details based on value & erasing marks to filter [I25 + I20]

**(D)** Pointing on an axis title and writing the attribute name to bind attribute to an axis [I3]

**(E)** Selecting points by dragging the pen on the axis scale & deleting unselected points through speech [I21]

**(F)** Pointing on the axis title and uttering attribute names with the modifier "Add" to bind additional attributes to an axis [I7]

Figure 5.3: Using INCHORUS to explore a movies dataset. Sub-figure captions describe the interactions being performed [along with the corresponding interaction pattern labels from Table 5.1].

uses the axis value-based ruler as a cut-off point and *erases* bars corresponding to genres with an average *Worldwide Gross* under 100M, filtering them from the view (Figure 5.3C).

**Shortlisting profitable movies.** With genres shortlisted, Joe now wants to compare the budget and gross for individual movies using a scatterplot. To do this, he first *erases* the *Production Budget* from the Y-axis title to remove it. He then *points* on the X-axis title region and starts writing "*budget*" in the ink pad, selecting *Production Budget* from the recommended list of attributes (Figure 5.3D). This replaces the *Major Genre* attribute on the X-axis with the *Production Budget*, creating a scatterplot. Since he works for a relatively small production house, he decides to focus on lower budget movies. He *drags the pen* along the X-axis scale to select movies with a budget under 100M and *says* "*exclude others*" (Figure 5.3E) to remove the unselected movies. To further focus on movies with high a return on investment, he also removes movies with a gross of under 200M.

With the filtered scatterplot, Joe starts examining other attributes. To understand what types the movies were, Joe maps the *Creative Type* attribute to the color of the points by *saying* "*Color by creative type*." Noticing that *Contemporary Fiction*, *Kids Fiction*, and *Science Fiction* are most popular, he filters out the other movie types by *erasing* them from the legend. Similarly, mapping the *content rating* to the color of points, Joe removes *R*-rated movies since his company is more interested in movies catering to a universal audience. This filtering results in the view shown in Figure 5.1.

Inspecting the scatterplot, Joe notices a set of low budget movies that have made a profit of over 5x. He selects these movies at the bottom left corner of the view as well as the three highest grossing movies at the top right corner of the view by *holding the modifier button* and *drawing* two free-form lassos around the points using the pen. Joe then filters out other movies from the chart by *saying* "*remove others*."

**Comparing shortlisted movies.** Finally, to analyze what characteristics made the shortlisted movies so successful at the box office, Joe wants to inspect and compare the shortlisted movies with respect to the relevant attributes. Joe first removes the *Production*

*Budget* from the X-axis by *erasing* it. He then *points* on the Y-axis (currently showing only the *Worldwide Gross*) and *says* "*Add budget, running time, rotten tomatoes and imdb rating*." INCHORUS, in response, adds the additional attributes to the Y-axis, creating a parallel coordinates plot. Joe further investigates the shortlisted movies by selecting different value ranges on the parallel axes and identifies a final list of five movies to present as examples of profitable and reliable investments to the management team.

### 5.2.3 Affordances and Feedback

To make users aware of possible actions, INCHORUS presents different types of affordances. For instance, when the user points on an axis, INCHORUS highlights attributes that can be mapped to that axis, renders an ink pad to indicate that written input can be provided, and flashes the command display box and microphone red to indicate that speech recognition is active (Figure 5.2). Note that the attribute pills are contextually highlighted based on the active view and the attribute type. For example, in Figure 5.2, *Major Genre* stays dark gray because it is already mapped to the X-axis. Alternatively, if one pointed on the Y-axis with a categorical attribute shown on the X-axis, the system would only highlight numerical attributes in the panel since we currently do not support visualizations simultaneously showing categorical attributes on both axes.

INCHORUS also provides constant feedback based on user actions. In addition to visual feedback for direct pen/touch actions (e.g., fading unselected points in Figure 5.3E, adding an orange stroke when pointing on an axis title as shown in Figure 5.2), the system also displays three types of textual feedback messages above the chart area (Figure 5.1D): 1) Success: when the system successfully executes operations in response to user actions (example messages include *Coloring by Major Genre*; *Sorted bars by Worldwide Gross in descending order*), 2) Void action: when users performs a valid operation but the operation has no effect on the view (example messages include *No points meet that filtering criteria*; *Bars are already sorted in descending order by IMDB Rating*), and 3) Error: when users

perform invalid actions or the system is unable to interpret a speech command (example messages include *The pen cannot be used in the panel area. Please use touch.*; *Unable to process that command. Please try a different one*).

By providing contextually-relevant affordances before an action and complementing them with feedback after actions, INCHORUS helps users both know what actions are available as well as interpret the system's reactions to their actions.

### 5.2.4 Implementation

INCHORUS is implemented in JavaScript as a web-based application. All visualizations are rendered using D3.js [20]. Pen and touch inputs are collected as standard JavaScript events and processed by custom event handlers. INCHORUS uses the HTML5 speech recognition API [209] for translating speech-to-text. To improve recognition accuracy, the speech recognizer is trained with the operation-specific keywords (Table 5.1) and attributes and values in the loaded dataset.

We implemented a custom JavaScript-based lexical parser to interpret the recorded NL commands. The lexicon consists of the attributes and values in the dataset as well as manually defined operation-specific keywords. To identify operations, targets, and parameters, the system compares the tokenized input string to the lexicon. If it is unable to detect a target using the NL command alone, the system employs multimodal fusion and infers the target through the invoking instrument (e.g., axes) and the active view (e.g., selected points).

## 5.3 User Study

We used INCHORUS as a test bed to assess the general usability of the proposed interactions and gather subjective feedback. In particular, since multimodal interaction with visualizations through pen, touch, and speech was a novel concept, we wanted to assess its practical viability and see whether people actually adapt to using this style of interaction and are

able to perform common visual analysis tasks.

### 5.3.1 Participants and Setup

We recruited 12 participants (P1-P12; six females, five males, and one "undisclosed"), ages 27-55, via email through mailing lists of a large technology company. All participants rated themselves as being fluent English speakers and had a working knowledge of data visualizations (i.e., understood basic visualization types and elements such as axes, legends) but not necessarily specific visualization systems (e.g., Tableau, Microsoft Power BI). In terms of prior experience with input modalities, all participants said they use touch-based systems on a daily basis. Two participants said they use a pen once in a few weeks, five said they had used it in the past but do not use it regularly, and five said they had no experience working with a pen. Eight participants said they use voice-based systems on a daily basis, three said they use it on a weekly basis, and one said she only occasionally uses voice-based systems.

Sessions were conducted in-person in a quiet conference room. Participants interacted with InChorus on Google's Chrome browser on a 12.3" Microsoft Surface Pro set to a resolution of 2736 x 1824. Participants were encouraged to position the device in a way that was most comfortable for them. A 24" monitor was used to display the study instructions and tasks as a slide show. Participants received a $50 gift card as a compensation for their time. All sessions were audio and video recorded.

### 5.3.2 Procedure and Tasks

Each study session had four phases including a training phase, two task phases, and debriefing. The study protocol and tasks were iteratively refined through 12 pilot sessions. We included two types of tasks to emulate two significantly different visual analysis scenarios. Specifically, the first task phase emulates scenarios where users know the operations they want to perform and have to communicate that intent to the system through interactions. On the other hand, the second task phase emulates scenarios where users first need to think

about the task they want to accomplish (e.g., think about the attributes they want to use and the type of chart they want to create), translate that task into system operations (e.g., binding attributes to specific encodings), and finally perform those operations through the supported interactions. Each session lasted between 71-124 minutes (average: 86 min).

**Introduction & Training.** After they provided consent and filled out a background questionnaire, participants were introduced to the various system operations along with possible interactions for each operation using a dataset of 303 cars with eight attributes (e.g., *Horsepower*, *Acceleration*, *Origin*) for each car. During this phase, as training, participants were free to practice the interactions until they felt confident performing them. This phase lasted approximately between 37-60 minutes (average: 46 min).

**Task Phase 1: Replication and Value Identification.** In this phase, participants were given four tasks using the IMDB movies dataset introduced as part of the usage scenario earlier. Each task consisted of a set of one to four sub-tasks that displayed a visual state for participants to replicate or values they had to identify. For example, one of the tasks had four sub-tasks requiring participants to 1) recreate a given grouped bar chart, 2) filter out two categories of values, 3) switch to a multi-series line chart, and 4) identify series values for a specific year. This first task phase took approximately 8-26 minutes (average: 13 min) to complete.

**Task Phase 2: Fact Verification.** Participants were given a dataset of 500 US colleges with nine attributes for each college including a number of numerical (e.g., *Cost*, *Admission Rate*) and categorical attributes (e.g., *Control Type*, *Region*). Following the jeopardy-style evaluation [62] for visualization NLIs, we gave participants five statements (e.g., *There are more public schools in Southwest than the Great Lakes*) that they had to mark as true or false based on their exploration of the data. In addition to stating the answer, participants also had to verbally justify their responses and take screenshots of visualizations they used. This phase lasted approximately between 7-28 minutes (average: 14 min).

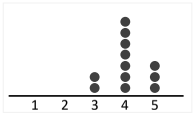
**Debrief.** At the end of the session, we had a debriefing phase that included a post-

A

| Operation | Interaction Patterns | Total (with %) | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 | P11 | P12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bind attribute to an encoding | Drag II to DI | 150 (40%) | 24 | 3 | 26 | 33 | | | 29 | 1 | 10 | | 12 | 12 |
| | Point on DI and tap II | 177 (47%) | 2 | 27 | | 1 | 46 | 29 | 3 | 21 | 9 | 20 | 13 | 6 |
| | Point on DI and write P | 6 (2%) | 2 | | | | | | | 3 | 1 | | | |
| | Point on DI and speak P | 42 (11%) | 4 | 1 | 8 | 3 | | | | 3 | 5 | 10 | | 8 |
| | Speak < T , P > | 1 (0%) | | | | | | | 1 | | | | | |
| Bind multiple attributes to an encoding | Point on DI and speak < K , P > | 21 (40%) | 3 | | | 3 | | 3 | | 5 | 2 | 4 | | 1 |
| | Speak < T , K , P > | | | | | | | | | | | | | |
| | Drag II to DI | 11 (21%) | | | 3 | 2 | | | 5 | | | | 1 | |
| | Point on DI and tap II | 20 (39%) | | 5 | | | 2 | | | 2 | 2 | 5 | 2 | 2 |
| | Point on DI and write P | | | | | | | | | | | | | |
| Remove attributes from an encoding | Erase ( P from) DI | 71 (93%) | 2 | 1 | 3 | 10 | 6 | 9 | 7 | 9 | 4 | 14 | 4 | 2 |
| | Point on DI and speak <( K ) , P > | 5 (7%) | 1 | 1 | 2 | | | | | | | | | 1 |
| | Speak < T , K ,( P ) > | | | | | | | | | | | | | |
| Change data aggregation level | Point on DI and write P | 13 (50%) | 1 | | 1 | 1 | 2 | 2 | 1 | 2 | 1 | | 2 | |
| | Point on DI and speak P | 13 (50%) | 1 | 2 | 1 | 2 | | | 1 | 2 | | 2 | | 2 |
| | Speak < T , P > | | | | | | | | | | | | | |
| Sort | Swipe on DI | 119 (94%) | 8 | 6 | 19 | 9 | 4 | 19 | 15 | 6 | 10 | 15 | 8 | |
| | Point on DI and speak < K , P > | 8 (6%) | | | 1 | | | | | | | | | 7 |
| | Speak < T , K , P > | | | | | | | | | | | | | |
| Filter marks | (Select +) Erase DI | 20 (17%) | | 4 | | 2 | | 2 | 8 | | 3 | | | 1 |
| | w/ DI selected, speak < K > | 100 (83%) | 15 | 10 | 11 | 7 | 10 | 6 | 2 | 7 | 10 | 7 | 7 | 8 |
| Filter by criteria | Erase P from DI | 29 (73%) | | | 3 | 1 | 1 | 6 | 4 | 5 | | 4 | 2 | 3 |
| | Speak < K , P > | 11 (27%) | 1 | | 1 | 6 | 1 | | 1 | 1 | | | | |
| Get mark details | Long press DI | 114 (100%) | 2 | 11 | 12 | 12 | 12 | 17 | | 17 | 1 | 13 | 7 | 10 |
| Get mark details by value | Drag along DI | 49 (100%) | 2 | 1 | 4 | 7 | 4 | 1 | 9 | 1 | 12 | 1 | 5 | 2 |
| Change chart type | Speak < P > | 20 (100%) | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 8 | 1 | 2 | 1 | 1 |

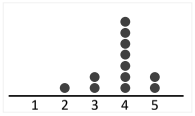
B

**Q1.** I think that I would like to use this system frequently.



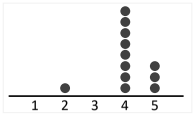

**Mean:** 4.08

**Q2.** I thought the system was easy to use.



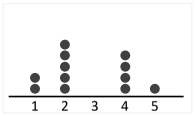

**Mean:** 3.83

**Q3.** I found the various functions in this system were well integrated.



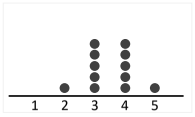

**Mean:** 4.08

**Q4.** I found it difficult to switch between modalities or use them simultaneously.



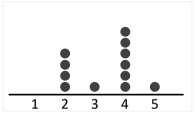

**Mean:** 2.75

**Q5.** I felt very confident using the system.




**Mean:** 3.50

**Q6.** The tool let me interact the way I naturally wanted to.




**Mean:** 3.33

Figure 5.4: Summary of study results. (A) Operations executed during the study along with the frequency of interaction patterns which were used to perform those operations. Bar widths are normalized for individual operations to facilitate comparison of alternative interaction patterns and (B) Responses to post-session likert-scale questions about the interaction experience.

session questionnaire consisting of likert-scale questions regarding their interaction experience and an interview. During the interview, we asked participants general questions about their overall experience, asking them to list interactions they particularly liked/disliked, as well as targeted questions based on our observations during the session.

### 5.3.3 Results

All participants successfully replicated charts or identified values for the four tasks in the first phase. For the second task phase, nine participants correctly verified all five statements whereas three (P1, P5, P10) correctly verified three out of the five statements. The average completion time for individual tasks was 3 minutes for the first phase and 2:30 minutes for the second.

*Interaction Summary*

To track participants' interactions, we reviewed the session videos to count the number of times participants attempted operations in Table 5.1 along with which interaction patterns they used. We identified a total of 1197 attempts, out of which 1000 (83%) executed successfully, 67 (6%) were invalid operations (e.g., mapping an attribute to color with no attribute mapped to the X/Y axes), 15 (1%) used unsupported speech commands (e.g., invoking an axis-value based ruler and saying "*Remove points under this*") or pen/touch gestures (e.g., dragging an attribute from the color legend to the X axis), and 115 (10%) involved erroneous interactions (e.g., speech recognition errors, conflicting pen and touch). These errors included dragging an attribute pill outside the screen or dropping it outside the axes/legend (36), speech recognition errors (32), forgetting to trigger recording of voice commands before speaking (16), conflicting pen and touch (e.g., trying to drag an attribute pill with the pen) (15), dragging an attribute while pointing on an axis (6), forgetting to select from the list of recognized items while writing (5), and issuing incomplete speech-only commands (e.g., saying "*Remove under 1200*" without specifying an attribute name)

(5).

For invalid operations, unsupported and erroneous interactions, participants often reattempted their initial interaction one or more times before switching to a different interaction pattern or operation. Hence, to avoid double counting interaction patterns by including these reattempts, we only summarize the 1000 valid and successful interactions in Figure 5.4A.

*Subjective Feedback*

Figure 5.4B summarizes participants' responses to the post-session questionnaire. In general, participants were positive about the overall experience stating they found the system functionalities well integrated ($M$ = 4.08 out of 5) and that they would want to use the system frequently ($M$ = 4.08). However, all participants noted that there was a learning curve especially during the training phase. That said, participants felt the interactions were intuitive and just needed some getting used to as they had no prior experience with multimodal interfaces. For instance, P6 compared her initial reaction to using a new type of keyboard and said "*...it was a cohesive system. It's just kind of getting the muscle memory down. It's like when you switch to someone else's keyboard you know where everything is but you just can't type.*" However, after the training, not only did she successfully complete all tasks but as highlighted by the counts in Figure 5.4A, did so using a variety of patterns including unimodal interactions, bimanual interactions, and multimodal interactions combining pen/touch with speech.

While seven participants said they did not find it difficult to switch between or combine modalities, five said it was confusing, in particular to differentiate between pen and touch. For instance, P9 said "*Distinguishing and switching between voice versus not-voice was not difficult but pen versus finger was a tougher one. I don't think I make as much of a distinction between pen and finger and only use the pen to be more specific, it's a finer point as opposed to being a different tool.*" We believe this confusion may have been

exacerbated by the fact that the three participants giving this feedback had never used a pen before and the remaining two had used it minimally in the past. Nonetheless, this confusion fuels the open challenge with designing bimanual pen and touch interaction [57, 76], deeming further investigation.

Lastly, on average, participants were neutral about how "natural" it felt to interact with the system ($M$ = 3.33), with four participants stating they felt the system did not support their natural workflow. During the interviews, we noted that more so than the interaction patterns, this stems from the fact that some participants preferred not to manually specify mappings to explore the data. For instance, P5 said "*I'm an excel person. If I had different templates I would have selected different types of graphs and seen what the data looks like and then chose the scatter graph.*"

## 5.4 Discussion

### 5.4.1 Potential of "Restricted" Natural Language Interfaces

A majority of the work on visualization NLIs has focused on developing interpretation techniques for complex and underspecified commands [62, 77, 172, 173]. While this is the holy grail of NLIs in general, responding to high-level questions and underspecified commands is challenging and highly error prone due to issues such as ambiguity and preserving context across commands. Our work sheds light on a more modest but less error prone class of NLIs for visualization. Specifically, with InChorus, we allow users to issue short keyword-based commands that can be used in conjunction with another modality (**DP5**) or individually to perform low-level operations. We found that these simple commands allowed participants to perform required operations while improving the overall speech recognition and interpretation accuracy: out of a total of 274 utterances, we only had 32 (11%) recognition errors and 7 interpretation errors (2%). This speech recognition error rate of an average of 3% per session was noticeably lower than that for ORKO (avg. of 16% per session). With such results in mind, in line with recent work advocating

for "restricted" NLIs in complex domains [133], perhaps an opportunity lies in exploring "restricted" NLIs for visualization that build upon simple lexicons and smoothen the transition from current direct manipulation and WIMP-based systems to NLIs. Although they start simple, over time, such systems could evolve to support more complex commands, incrementally exposing users to more advanced system functionality [61, 184].

### 5.4.2 Synergy between Input Modalities

Our goal was to enable an overall consistent and fluid interaction experience during visual analysis, accommodating users' individual preferences. To this end, instead of aiming to achieve equivalence between modalities, we synergistically combined three input modalities (i.e., pen, touch, and speech), leveraging their unique advantages (**DP5**). The distribution of interaction frequencies in Figure 5.4A illustrates that the proposed multimodal interactions accommodated varied user preferences while allowing all participants to perform common visual analysis tasks. While some participants (e.g., P1, P8, P9) were open to trying a wider range of interactions and using all modalities and others (e.g., P5, P6, P11) preferred resorting to touch as much as possible, all participants adapted themselves to using multimodal input and, when needed, successfully used different modalities (individually and in combination) to complete tasks. This was particularly encouraging given that some participants had no experience with some modalities (e.g., five out of 12 participants had never used a pen). P7 aptly summarized his overall experience of interacting with InChorus stating "*It [InChorus] feels completely integrated like this is one thing it's not like this is the pen stuff that I'm doing and now I got to sort through the finger things I can do or the things with voice commands it was like I'm going to interact with this system however best suits me and I'm able to do that nine times out of ten*."

## 5.5 Conclusion

We presented a set of design principles and concepts (operations, parameters, targets, instruments) that can help design and compare multimodal interactions for visualization systems. Following the listed principles and concepts, we developed 26 interactions involving pen, touch, and speech input that consistently support five core visual analysis operations (bind visual mappings, modify axes, filter, request details-on-demand, change chart type) across five popular chart types (bar charts, line charts, scatterplots, histograms, parallel coordinate plots) and their variants (e.g., grouped and stacked bar charts). Incorporating these interactions into a tablet-based multimodal visualization interface, InChorus, we demonstrate how speech-based multimodal interfaces can help overcome design challenges (e.g., supporting direct and consistent interactions across chart types) with pen/touch-only interfaces. Finally, through a user study where participants reenacted common visual analysis scenarios with InChorus, we also exemplified how the proposed multimodal interactions support visual data exploration.

Collectively, the systems and their evaluations in Chapters 4 and 5 illustrate that multimodal interfaces can support the same analytic capabilities as current visualization tools but enable a more fluid and expressive interaction experience that accommodates varying user interaction patterns and preferences **[RG2]**. Besides supporting my overarching thesis, the discussions in these chapters pertaining to system design challenges and user interaction behavior also lay the groundwork for systematically designing and evaluating future NL- and DM-based multimodal visualization interfaces.

# CHAPTER 6

# INTERWEAVING MULTIMODAL INTERACTION WITH FLEXIBLE UNIT VISUALIZATIONS TO ENABLE FREE-FORM DATA EXPLORATION

Through the research described so far, I have shown how multimodal visualization interfaces like ORKO and INCHORUS support a higher degree of expressivity (i.e., they give users more freedom to choose their preferred style of interaction and combine modalities in different ways). What users can accomplish with this freedom of expression, however, is constrained by the level of formalism of the underlying representations. For instance, when interacting with a scatterplot in a tool like INCHORUS, one can issue a voice command to color points by an attribute, use pen and touch in innovative ways to inspect points, and freely draw lasso regions to select points of interest using the pen. However, the colors and positions of all the points in the scatterplot remain tightly bound to the specified data attributes. Thus, even though touch and speech are available as input modalities, users cannot perform naturalistic actions like dragging the selected points of interest to a specific region on the view (as one might with physical objects laid out on a table) or issue a voice command to color the selected points differently (while still preserving the color coding for the remainder of the view). In contrast, even some of the earliest examples of NL- and DM-based multimodal interactions such as Bolt's "put-that-there" [19] allowed users to manipulate graphical objects in a free-form manner without binding them to actions they performed in the past (in this case, mapping visual encodings to data attributes).

This difference in the flexibility of actions between examples from the HCI literature and visualization systems suggested that perhaps we have missed out on exploring the true potential of multimodal interfaces by confining ourselves to well-specified representations and visual analysis operations. Thus, with this third project, we sought to explore how one can bring the fluid and synergistic interactions illustrated in classic examples of multimodal

graphics applications to visualization tools and support more free-form data exploration. To investigate this idea, we developed an approach that interweaves DM- and NL-based multimodal interaction with *flexible unit visualizations* that allow people to specify both systematically-bound views (e.g., scatterplots, unit column charts) and customized views reflecting their mental model of the data space.

In this chapter, I describe this interweaving approach and operationalize it within a pen, touch, and speech-based multimodal visualization interface, DATABREEZE[1]. I describe the iterative design process we followed to create DATABREEZE and highlight specific aspects of the system interface and interactions that empower synergistic use of the different modalities. I also report findings from a preliminary user study with DATABREEZE, discussing how the expressivity of multimodal interactions in concert with the flexibility of the underlying representation enabled novel, free-form data exploration workflows **[RG3]**.

## 6.1 Proposed Approach: Coupling Flexible Unit Visualizations and Multimodal Interaction

By representing individual data cases as unique visual marks, unit visualizations allow people to specify views with different levels of customization. For instance, a 2D scatterplot is an example of a *systematically bound* view, where the position of each visual mark is strictly bound to two data attributes (e.g., Figure 6.1A). Alternatively, by explicitly changing the properties (e.g., position, color) of marks, one can create a *manually customized* view, where not all points in the view are bound to the same set of data attributes (e.g., Figure 6.1B). To allow users to create both systematically bound and manually customized views, we propose the notion of **flexible unit visualizations**.

Similar to flexible linked axes [32], flexible unit visualizations could be especially powerful during more open-ended data exploration. For example, users could apply well-known unit visualization layouts (e.g., scatterplots, unit column charts) to correlate and compare

[1]The content of this chapter is based on work previously published in IEEE TVCG [186].

values. Once they identify a set of points of their interest or have a specific exploration criteria in mind, users could then explicitly move points out from the scatterplot into separate groups, coloring these groups to create a customized view that best fits their mental model (e.g., Figure 6.1B). Furthermore, the ability to spatially organize data and create virtual workspaces can also aid externalization during the sensemaking process, especially in the initial, exploratory phases [5, 6].

Such flexible data exploration is difficult with existing unit visualization tools that support interaction only through DM and control panels [49, 54, 121, 158, 159, 164]. For example, SandDance [49] only allows systematic specification: the changes to the view need to be specified through menus, not allowing users to manipulate individual data points to adjust the view. Alternatively, while other systems such as ForceSPIRE [54] and VisExemplar [164] support direct interaction with individual data points, changes resulting from this interaction are propagated to all data points, ultimately resulting in a systematically bound view. For example, juxtaposing two points in ForceSPIRE recomputes the force-directed layout and repositioning points in VisExemplar creates a new scatterplot.

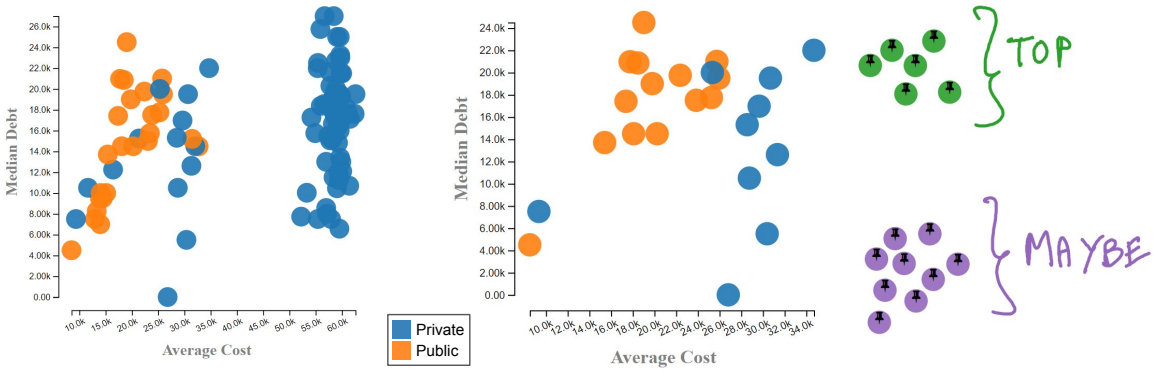


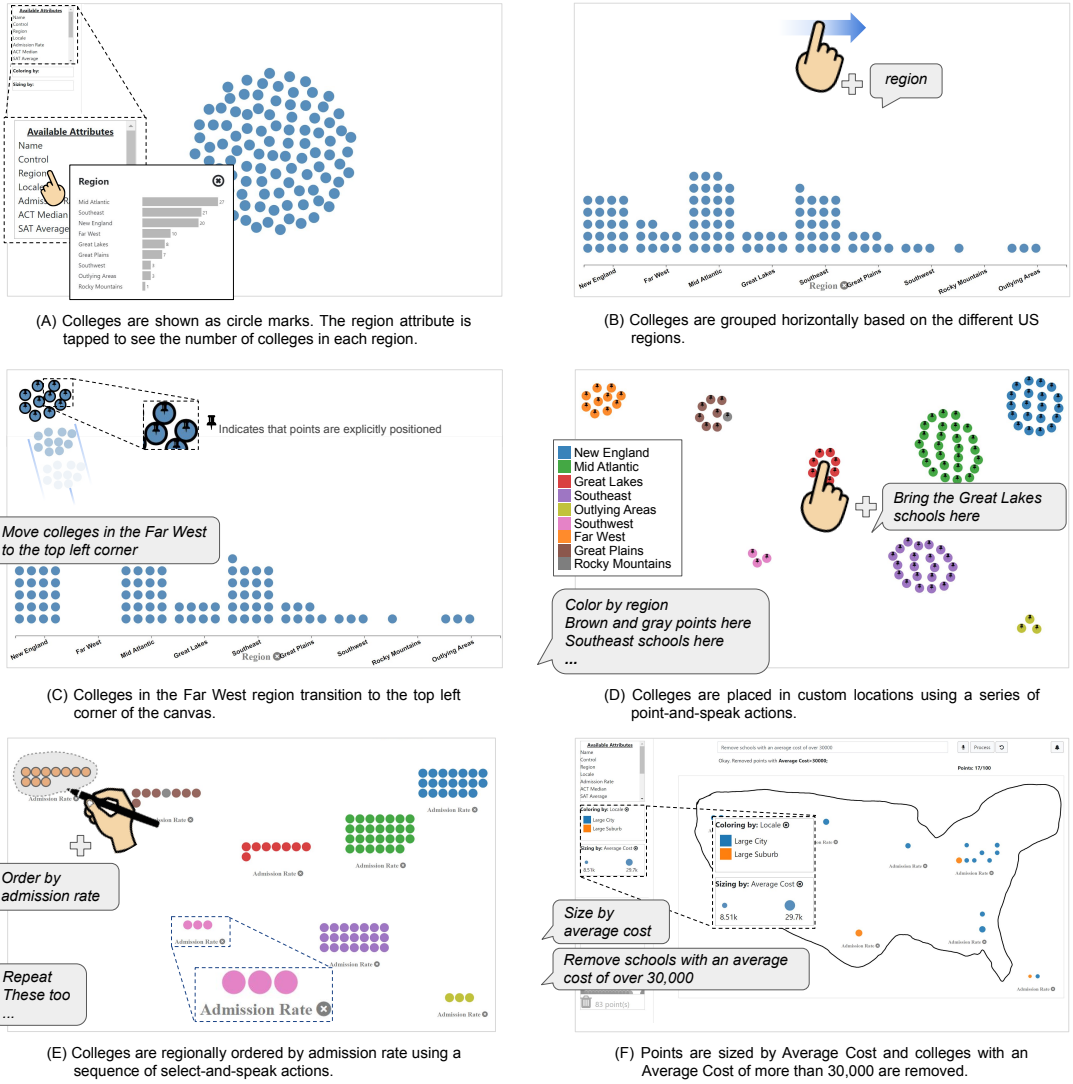

Figure 6.1: Flexibility afforded by unit visualizations. Both visualizations are displaying a U.S. colleges dataset. (A) A standard systematically bound scatterplot showing the relationship between *Average Cost* and *Median Debt*. (B) A manually customized view having a scatterplot (which is still bound to *Average Cost* and *Median Debt*) and two groups of preferred and potential colleges created by explicitly moving and coloring points from the initial scatterplot.

In general, relying solely on DM and control panels for interaction with a unit visualization can be problematic: the user interface can be overly complex when incorporating all of the desired actions and user interaction can be tedious for creating manually customized views (e.g., moving points that are not spatially clustered). On the other hand, the complementary strengths of DM and NL make a combination of the two a potentially effective solution for this challenge. The synergistic use of DM and NL when interacting with flexible unit visualizations could allow users to create both systematically bound and manually customized views, also enabling seamless switching between the views. Specifically, the precision and control afforded by DM can allow people to make fine-tuned, customized changes, whereas the ability of NL to augment systematic actions can help them overcome the repetition that accompanies DM.

### 6.1.1 Motivating Scenario: Identifying Preferences among US Colleges

To illustrate how multimodal interaction with flexible unit visualizations can facilitate visual data exploration, we describe a usage scenario. Imagine Sarah, a parent who is identifying colleges her daughter might want to apply to. Sarah downloads a dataset of the top 100 schools in the U.S. from a popular college ranking website. The dataset contains 14 attributes for each college including both categorical (e.g., *Region*, *Locale*) and quantitative (e.g., *SAT Average*, *Average Cost*) attributes. For consistency, we use this dataset in our examples throughout the paper.

**Getting an overview from systematically bound views.** The system initially shows all points clustered at the center of the screen. To learn more about the available attributes, Sarah taps on the different attributes in the attribute summary panel (Figure 6.2A). Looking at the *Region* attribute, Sarah decides to categorize colleges by their regions. To do this, she swipes from left-to-right on the canvas and says "*Region*." Inferring the specification of an axis through the swipe gesture and identifying the attribute via speech, the system creates a column chart grouping colleges by their regions (Figure 6.2B). Although this chart gives




(A) Colleges are shown as circle marks. The region attribute is tapped to see the number of colleges in each region.

(B) Colleges are grouped horizontally based on the different US regions.

(C) Colleges in the Far West region transition to the top left corner of the canvas.

(D) Colleges are placed in custom locations using a series of point-and-speak actions.

(E) Colleges are regionally ordered by admission rate using a sequence of select-and-speak actions.

(F) Points are sized by Average Cost and colleges with an Average Cost of more than 30,000 are removed.

Figure 6.2: Scenes illustrating the usage scenario of exploring colleges in the U.S. Sub-figure captions summarize the system states

.

her a good overview, Sarah finds it difficult to compare the different regions because the placement of regions does not visually correspond to their geographic locations.

**Contextualizing the data space with customized views.** Sarah decides to adjust the view so the points are positioned similar to how they look on a map of the U.S. To do this, she starts by saying "*Move the colleges in the Far West to the top left corner.*" This moves colleges in the *Far West* region to the top left corner of the canvas (Figure 6.2C). The 📌 icon indicates that Sarah explicitly moved the colleges and their positions are no longer bound to

the X-axis. To keep track of the different regions, Sarah says "*Color by region*"—applying a global color mapping to all points. She then continues to reposition other points through a series of speech-only (e.g., "*Move the New England schools to the top right corner*") and multimodal utterances (e.g., + "*Bring the Great Lakes schools here*," + "*Brown and gray points here*"). This results in a view that combines custom positioning of the colleges with the systematic color mapping based on the *Region* attribute (Figure 6.2D).

**Adding localized mappings for detailed exploration.** Curious about the competitiveness of schools in the different regions, Sarah decides to organize the schools further by their *Admission Rate*s. She starts from the Far West schools: after selecting the schools by drawing a lasso around them, Sarah says "*Order by admission rate.*" The system, in response, re-orders the selected Far West schools by their admission rates (in an ascending order). Because the position of the points is now determined by an attribute value, the system removes the icon from the re-ordered points. Sarah repeats this select-and-order sequence for other regions by selecting groups of points and saying commands like "*repeat*" and "*these too.*" Inferring from her previous commands, the system re-orders the colleges by their admission rates, leading to the view shown in Figure 6.2E.

**Adding annotations and global mappings to switch back to overview-level exploration.** To externalize her mental mapping of the data space on the canvas, Sarah draws a rough outline around the points using the brush tool to make the view look like a map of the U.S. As they might provide better career opportunities in the future, Sarah is more interested in schools in larger localities. To see the types of locations the schools are in, she says "*Color by locale*" and scans the updated chart. Noticing a lot of schools are in remote locations or small towns, Sarah filters the view by saying "*Remove schools that are not in large cities or large suburbs.*" This removes 47 points from the canvas. Next, to get a sense of the cost to attend each college, Sarah issues the command "*Size by average cost*" and notices that except for some colleges in the Great Lakes and Rocky Mountains, most colleges are expensive and potentially beyond their budget. To filter colleges further, she

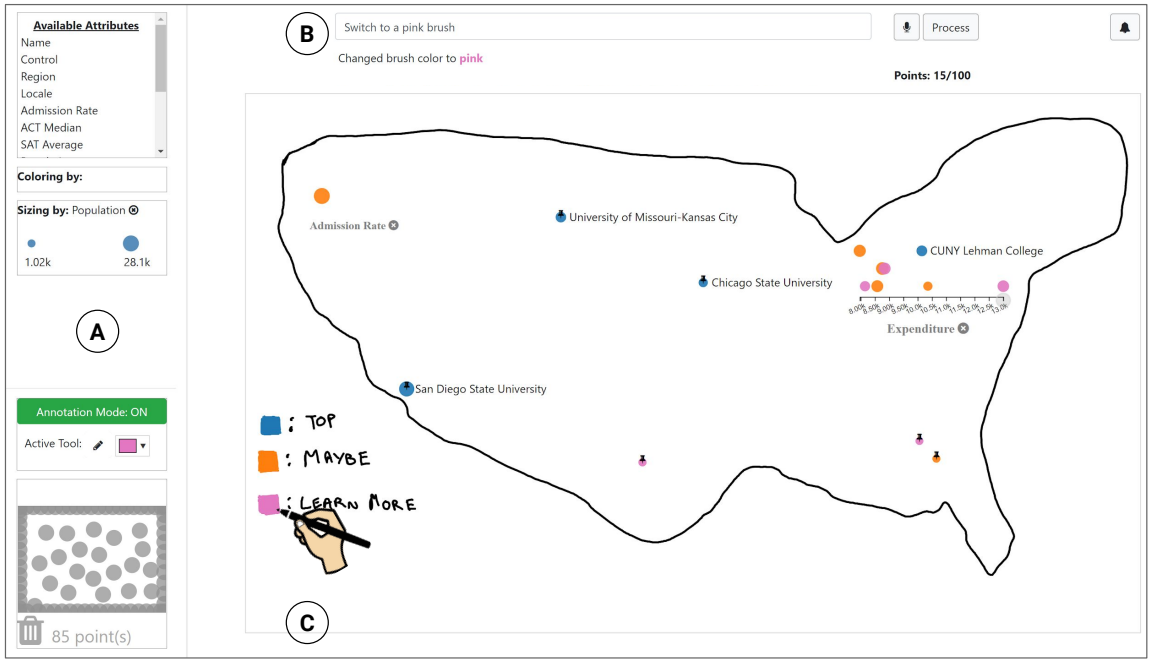



Figure 6.3: The system's interface as Sarah concludes her exploration: (A) Side panel with (from top to bottom) the attribute summary container, legends for global coloring and sizing attributes, annotation options, and a virtual bin for filtered points; (B) Speech input and feedback row; (C) Main canvas. Here, Sarah is using the brush tool to draw a custom color legend.

says "*Remove schools with an average cost of over 30,000*," leaving only 17 colleges on the canvas (Figure 6.2F).

**Combining global and local operations for drill-down.** Sarah taps on points individually to see their details and explores the colleges from different perspectives by iteratively assigning different attributes to the position, size, and color of points. To enable better comparison of schools within a geographic region, Sarah also creates local views by mapping attributes to the positions of subsets of points (Figure 6.3), inspecting the individual groups. Based on this inspection, Sarah decides that she does not want to consider the two colleges in the 'Outlying Areas' and removes them.

**Externalizing custom mappings.** Inspecting the remaining 15 schools, Sarah identifies four schools that are her top picks, six schools that she would strongly consider, and five schools that she needs to read more about. To externalize this mental ranking, Sarah selects

the four schools that are her top choices and says "*Add labels and color them blue*" to show their names and change their color. Sarah then selects the six colleges that she would strongly consider and says "*Color these orange*," and selects the remaining five schools and says "*Pink.*" Sarah again activates the brush tool and draws a legend to note her color choices (Figure 6.3). Concluding her exploration, Sarah sends this view via an email to herself to discuss it with her family members.

### 6.1.2 Design Process and Goals

To create a system that supports the illustrated style of visual data exploration, we followed an iterative design process that helped us identify and refine a set of design goals we needed to accomplish. Below, we describe this design process and resulting design goals.

We first define a few key terms we use throughout this paper. We use the term *command* to refer to any type of NL utterance such as a query, comment, or question. By an *operation*, we refer to actions like selection, changing encodings, coloring points, etc. Operations typically require *parameters* (e.g., attributes, color names, data values) and operate on one or more *targets* (e.g., all points on the canvas, selected points, points meeting a specific data criteria).

#### *Design Process*

As a test bed, we initially implemented a basic version of a pen-, touch-, and speech-based multimodal unit visualization tool. The tool supported a minimal set of operations (e.g., assign X/Y axes, change color and size, filter) and interactions (e.g., dragging to move points, drawing a lasso for selection, speech commands for individual operations). We implemented the system on an 84" Microsoft Surface Hub (Figure 6.4) to support scalability (in terms of number of points along with the ability to interact with individual points), as well as to provide the freedom to spatially organize the view.

We iterated on the tool's design and implementation across six design sessions (each

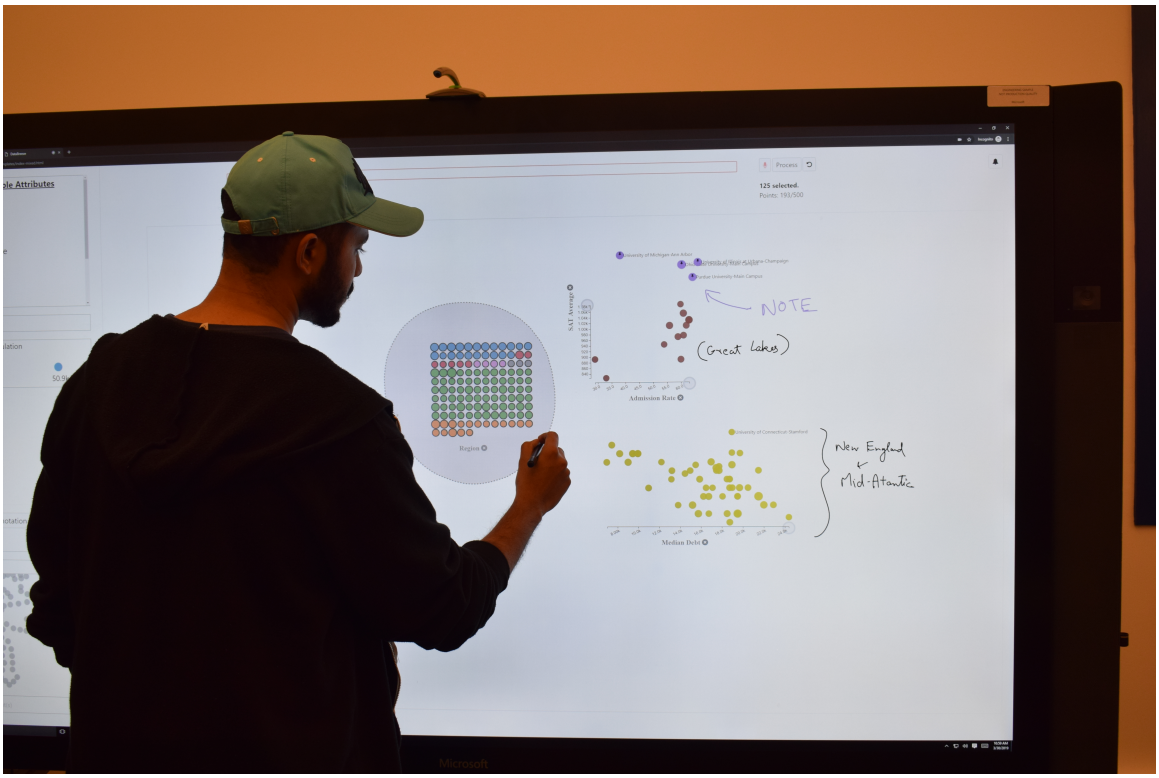



Figure 6.4: DATABREEZE running on an 84" Microsoft Surface Hub with an external microphone placed on top of the display to record speech input.

between 30-90 minutes) among ourselves and other graduate students. In these sessions, we investigated interactions during open-ended tasks (e.g., shortlist a set of startups to invest in, shortlist a set of colleges for your child or younger sibling). These design sessions allowed us to critically reflect on the design, get early feedback on the interactions (e.g., pen/touch gestures, grammar of spoken commands), and identify operations that we needed to support. Specifically for the pen/touch gestures, due to a lack of consensus during the design sessions, we additionally conducted four informal elicitation sessions with graduate students, observing how they performed actions such as selecting and moving points, invoking and interacting with context menus, etc.

*Design Goals*

We considered several factors for developing a multimodal system supporting flexible unit visualizations (e.g., How should the system integrate input from multiple modalities?

Should changes explicitly made to a subset of points be propagated to all points in the view?). Below we list the goals that we initially had in mind at the start of the project (DG1-4) as well as the ones we incrementally derived based on the observations during the design sessions (DG5-8). While these goals are primarily applicable to multimodal interfaces supporting flexible unit visualizations, some of them (DG2-4, DG6, DG7) are also generally applicable to DM- and NL-based multimodal visualization systems.

DG1. **Support both systematic binding and manual customization.** The premise of this work is that allowing people to manually customize systematically bound visualizations can aid data exploration, offering high flexibility. To enable this manual customization, the system should allow users to specify visualizations similar to current tools (e.g., assigning X/Y-axes attributes to create a scatterplot), while still supporting data item- or subset-level manipulation (e.g., dragging points out of a scatterplot to create a customized group, changing the color of specific points).

DG2. **Support various multimodal input patterns.** Prior research on multimodal interfaces has shown that although a system supports multiple modalities, people may choose to interact using a single modality and not combine inputs [144]. Furthermore, even when using multiple modalities, people may not use them simultaneously and instead combine them sequentially (e.g., select a set of points with touch, pause, and then issue a spoken command) [141, 144]. Hence, the system should support unimodal input as well as both sequential and simultaneous integration of modalities.

DG3. **Leverage simple pen/touch gestures.** While pen/touch input can be highly expressive, complex gestures involving multiple fingers or bimanual interaction can be difficult to learn and discover [215]. These challenges are further amplified with the addition of a third modality in the form of speech. Therefore, the system should leverage simple and familiar pen/touch interactions that are easy to learn while still supporting the required set of operations.

**DG4. Provide instruction and feedback for speech input.** Lack of instruction (knowing what can be said) and feedback (understanding what the system did in response to a command) are well-known challenges with NL interaction [22, 177, 219, 220]. This challenge is amplified in multimodal interfaces where the linguistic structure of commands may differ when users interact multimodally [144]. Hence, the system should assist discovery and learning of the supported range of speech commands. Furthermore, the system should make users aware of the actions it took in response to speech commands—giving users the option to revert or correct them.

**DG5. Support both global and local changes.** We observed that participants wanted to perform operations at both a global (e.g., creating a scatterplot using all data points) and a local level (e.g., selecting a subset of points within a scatterplot to form an ordered group). Therefore, to support incremental data exploration and smoother transitions between systematically bound and customized views (DG1), the system should let users perform operations on all points on the canvas (global operations) or on a subset of data items (local operations). However, local changes may conflict with previously applied global mappings (e.g., moving points may break a globally specified position mapping). To overcome this, we initially removed the global mapping in case of conflicting local changes (e.g., removing the global X-axis scale if a set of points are explicitly moved). During the design sessions, however, we observed that participants preferred that visual elements of the previously applied global changes be preserved even when conflicting local changes are made. We found that this helped participants contextualize changes and continue their exploration. Hence, the system should try to preserve context for local changes and provide visual cues to differentiate between local and global mappings. For instance, in Figure 6.2C, the *Region* scale is shown on the global X-axis even though a subset of the points are moved out. Furthermore, the 📌 icons on the moved points indicate that the points are not bound to the global view.

**DG6. Support equivalence between pen and touch.** Following Hinckley et al.'s guideline of "*pen writes, touch manipulates,*" [76] we initially applied a division of labor tactic separating the roles of pen and touch: we let people draw lassos and select points using the pen, while moving the points with a finger. During the design sessions, however, we observed that participants frequently confused the role of pen and touch, often trying to use the two interchangeably (e.g., drag points with a pen, select points with a finger). This adversely affected the system's usability, suggesting that the benefits of greater expressiveness through separated roles was not worth the resulting confusion it caused. Thus, when semantically meaningful differences are missing between pen and touch interaction, the system should support equivalent and consistent operations between the two.

**DG7. Support implicit and explicit triggering of speech.** While we supported unimodal, sequential, and simultaneous integration of modalities in our initial prototypes (DG2), similar to ORKO, users had to explicitly trigger speech input using a "listen" button or a wake-word. However, we observed that this impeded multimodal interaction often resulting in participants saying a command after performing a gesture only to realize that the system was not listening. Therefore, to facilitate more seamless multimodal interaction, the system should provide both explicit and implicit speech activation techniques.

**DG8. Support externalization of custom mappings.** During the design sessions, participants created custom mappings (DG1) fitting their mental models by making local changes to the view (DG5). To let people externalize their custom mappings (e.g., adding labels for "virtual bins" or drawing custom legends as in Figures 6.1B and 6.3), the system should support basic inking features.

## 6.2 DATABREEZE

With the design goals listed above in mind, we developed DATABREEZE—a multimodal system that facilitates visual data exploration by supporting constructing and interacting

Table 6.1: Examples of supported operations and their corresponding speech and multimodal commands in DATABREEZE

| Operations | Sample Speech & Multimodal Commands |
|---|---|
| **Assign X/Y-axes** | *Sort vertically by Admission Rate;*  *+ SAT Average; Align horizontally by debt;* |
| **Filter** | *Remove schools in the Far West; Remove all points except the blue ones;*  *+ Remove;* |
| **Color/Size by attribute** | *Color by region;*  *+ Size these by expenditure; Color by locale and then size by average cost;* |
| **Order by attribute** |  *+ Order by admission rate; Rearrange schools in the Southeast by their population;* |
| **Move** | *Put the public schools on the right;*  *+ Bring the private schools here;*  *+ Green here;* |
| **Others** |  *+ Color red; Highlight Stanford;*  *+ Summarize; Add labels to all public schools;* |

with flexible unit visualizations. Table 6.1 illustrates operations currently supported in DATABREEZE that were derived and refined based on the aforementioned design sessions.

### 6.2.1 Pen & Touch Interaction

DATABREEZE supports three familiar gestures (tap, long press, and drag) (DG3) that can be performed on the canvas or on a data point (Table 6.2). To support externalization of custom mappings (DG8), DATABREEZE also supports drawing using a brush tool. If the brush tool is active, dragging the pen on the canvas renders ink strokes. Otherwise, pen and touch can be used interchangeably (DG6). Akin to previous pen- and touch-based visualization tools [94, 217], DATABREEZE employs radial context menus (Figure 6.5) with which people can perform operations.

### 6.2.2 Speech Interaction

DATABREEZE allows the use of speech unimodally or as part of multimodal interactions to perform the supported operations (DG2). Table 6.1 highlights some examples of supported NL-only and multimodal NL commands.

*Triggering Speech Input (DG7)*

DATABREEZE starts listening or recording user utterances in response to one of five user actions: 1) tapping the microphone icon (🎙), 2) double-tapping on the canvas (similar to

Table 6.2: Pen and touch interactions in DATABREEZE. Except when the brush tool is active, pen and touch can be used interchangeably.

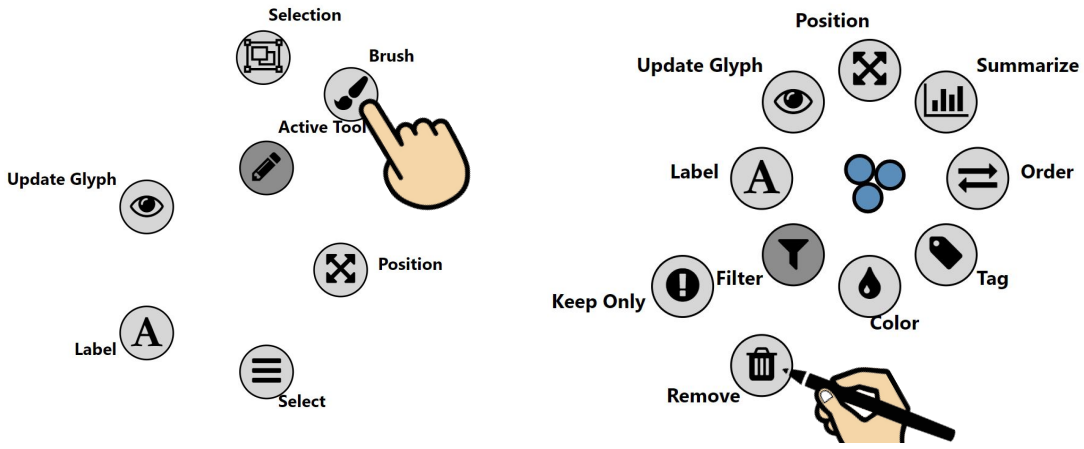
| Gesture | Target | Touch | Pen |
|---|---|---|---|
| Tap | Canvas | Clears selections | |
| | Point | Shows tooltip with label | |
| Long press (>1 sec.) | Canvas | Select + Context menu for global operations | |
| | Point | Select + Context menu for local operations | |
| Drag | Canvas | Draws a selection lasso or initiates X/Y axis | (w/ brush tool) Draws ink strokes |
| | Point | Moves point(s) | |

knocking on a door), 3) long pressing on the canvas or a data point with a finger or pen, 4) selecting one or more points by drawing a lasso, or 5) swiping horizontally or vertically on the canvas to specify an axis. The first two triggering techniques are explicit and allow users to initiate speech input on-demand. The latter three are more implicit triggering techniques to support smoother multimodal interaction where the system starts listening based on the user's pen or touch actions. Whenever the system is listening, the microphone icon and the input box flash red (🎙).

*Interpreting Speech Commands*

**General command interpretation strategy.** We use a combination of a template- and lexicon-based parser to interpret speech. DATABREEZE identifies the operations, targets, and parameters of the spoken command by comparing the input to predefined command phrasing templates (e.g., *Size by [attribute]*) compiled from the initial design sessions. If the input does not match a template, the system tokenizes the command string and compares the tokens to the system lexicon to infer the operations, targets, and parameters. The system lexicon contains keywords/phrases mapping to the different operations (e.g., 'order,' 'color,' 'remove') and parameter values (e.g., attribute names and values, color names, canvas regions like 'top' and 'right').

Consider the example command "*Remove all private schools with an average cost of*




(A) Context menu for global operations (B) Context menu for local operations

Figure 6.5: Context menus for pen/touch interaction. (A) A global menu is invoked by long pressing on the canvas background. The user is switching to the brush tool. (B) A local menu is invoked by long pressing on selected data points. The user is filtering out the selected points.

*more than 30,000*" that does not match a predefined phrasing template. To interpret this command, the system first removes all stopwords from the input command. Next, the system derives all N-grams from the string and compares the N-grams to the available lexicon using the cosine similarity and the Wu-Palmer similarity score [216]. This comparison results in the system matching the N-gram "*private schools*" to the attribute-value pair *Control=Private* and "*average cost more than 30,000*" to the attribute-value pair *Average Cost>30,000*. Furthermore, using the word "*Remove*", the system also infers that the user is referring to the *filter* operation. Combining the identified operation and attribute-value pairs, the system removes all points having *Control=Private* and *Average Cost>30,000*.

**Handling follow-up commands.** In addition to fully-specified speech commands, DATABREEZE also supports follow-up commands. We use conversational centering [68] to infer follow-up commands, extending the model beyond attribute-focused operations (e.g., filtering and changing encodings to support additional data item-level operations introduced in DATABREEZE (e.g., updating point colors, moving points, specifying local axes). At a high-level, when it executes a command, the system creates a context object that contains

references to the operations, parameters, and targets associated with the command. If the subsequent command contains new operations, parameters, or targets, this context object is refreshed. If not, the system tries to identify the missing information using the context object and updates it accordingly. Figure 6.6 shows a sample follow-up command sequence employing this strategy.

### 6.2.3 Multimodal Interaction

DATABREEZE supports three types of multimodal commands where operations, parameters, and targets are derived using a combination of the pen/touch and speech input.

**Point-and-speak.** Figure 6.2D shows an example of a multimodal point-and-speak command where the user points on the canvas and says "*Bring the Great Lakes schools here*." In this case, the command contains references to the operation (*Move* operation identified using the word "*Bring*") and target (points with *Region=Great Lakes*). However, the parameter for the move operation (i.e., canvas location to move points to) is provided via touch and is deictically referenced using the word "*here*." Thus, by considering both the input command and how it was triggered, the system moves the target points to the requested location.

**Select-and-speak.** With select-and-speak commands, users can select a set of points and issue a speech command to perform a local operation on those points. An example of this is shown in Figure 6.2E where the user selects a set of points and says "*Order by admission rate*." In this case, the system identifies the operation (*Order*) and parameter (*Admission Rate*) using the command, inferring the target based on the preceding selection that triggered speech input.

**Swipe-and-speak.** With swipe-and-speak commands, users can swipe horizontally or vertically on the canvas and say an attribute name to position points by a specific attribute. An example of this is shown in Figure 6.2B where the user swipes from left-to-right and says "*region*." By detecting the swipe gesture and the attribute *Region*, the system infers


Figure 6.6: An interaction sequence illustrating follow-up commands in DATABREEZE. The first command orders *Mid-Atlantic* schools by the *Control* type (*Public* vs. *Private*). The subsequent command implicitly refers to the *Mid-Atlantic* schools and the *Order* operation from the initial command. The second follow-up command again implicitly refers to *Mid-Atlantic* schools but specifies a different operation (*Assigning X-axis*).

that the user wants to arrange points horizontally by *Region* and creates a unit column chart.

*Handling Ambiguity and Failure (DG4)*

To highlight ambiguity in commands, DATABREEZE presents ambiguity widgets [62]. They appear in the feedback row and allow users to refine ambiguous values in the input command using pen/touch. With the range of explicit, follow-up, multimodal commands and their possible phrasing variations (DG2), interpretation errors are practically bound to occur during NL interaction. When input commands are unintelligible (e.g., "*Apply a legion shelter*" instead of "*Apply a region filter*") due to speech recognition errors or beyond the scope of supported commands (e.g., "*Plot colleges geographically*"), DATABREEZE notifies users about this failure, asking them to try a different command.

A key difference between DATABREEZE and existing visualization NLIs lies in how the system handles partially complete commands. Current systems either only notify users about failure [77, 189], apply system defaults [77, 189], or list all possible operations or values to choose from [189]. Instead, utilizing this failure as a teaching opportunity, DATABREEZE performs an additional processing step and checks the command for partial phrasings or keywords that might map to potential operations. If it finds a match, the system generates an explanation along with an exemplary command that could help the user learn the correct phrasing. An example of this is shown in Figure 6.7C where the system suggests

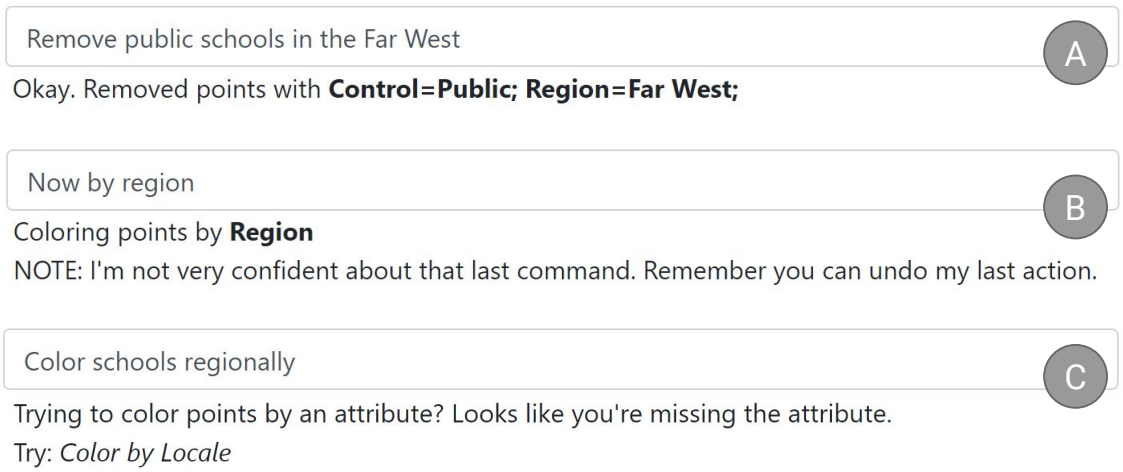



Figure 6.7: Feedback messages shown after (A) successfully executing a command, (B) executing a follow-up command, and (C) partially interpreting a command.

the example command *Color by Locale* when it is unable to interpret the user command "*Color schools regionally*." In this case, DATABREEZE was able to determine the operation using the keywords '*Color by*' but was unable to detect the attribute to color by (and thus randomly chose *Locale* as an example: if there are multiple operations a partial command may map to, the system selects one at random). By notifying users about its actions and providing suggestions and explanations when commands fail, DATABREEZE attempts to reduce the "black-box" effect of NL interaction.

*Command Feedback and Discovery (DG4)*

When DATABREEZE processes a spoken command, it updates the feedback row to summarize the actions it performed in response to the command. If the user command is successfully processed, the system states the operation along with target or parameter values (e.g., Figure 6.7A). However, for follow-up commands, because the system infers user intent based on previous commands, there is a higher chance of error. To highlight this, in addition to the action performed, the system feedback reminds users about the undo feature that allows reverting the most recent action (Figure 6.7B).

To preemptively make users aware of possible speech commands and phrasings, DATABREEZE

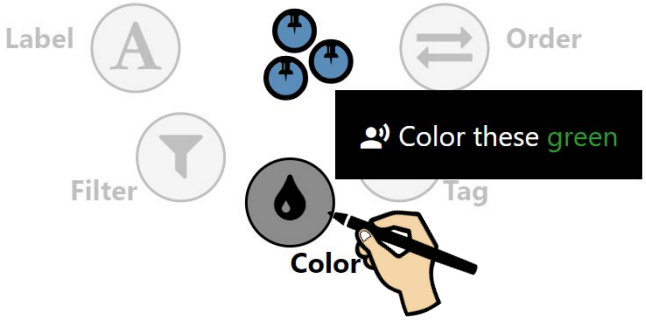



Figure 6.8: A sample deictic speech command for coloring selected points is shown on long pressing a context menu option.

employs an adaptive command discovery approach [61, 184], suggesting commands through tooltips when users long press on context menu options (Figure 6.8). DATABREEZE also suggests commands post-hoc once an operation has been performed using pen/touch. To deliver these suggestions unobtrusively, the system displays them above the canvas in the feedback row. For example, if the user removes labels for all points on the canvas using the context menu, the feedback row displays the message '💡 To remove all labels, you could also 🗣 *Clear all labels*.' Users can turn off the system suggestions by tapping the 🔔 icon at the top right corner (Figure 6.3).

### 6.2.4 System Implementation and Architecture Overview

DATABREEZE is implemented as a web-based application and supports data files with numerical and categorical attributes in the CSV format. The visualization is rendered using D3.js [20]. All pen/touch inputs are collected as standard JavaScript events and processed by custom event handlers. DATABREEZE uses the HTML5 webkit speech recognition for translating speech-to-text. At the start of each session, we train the recognizer with the data attributes and values from the input dataset as well as a list of system keywords (e.g., remove, summarize). While it still detects arbitrary speech, this training helps improve the recognition accuracy for the most relevant keywords in user commands. Translated commands are processed using a custom interpreter implemented in JavaScript.

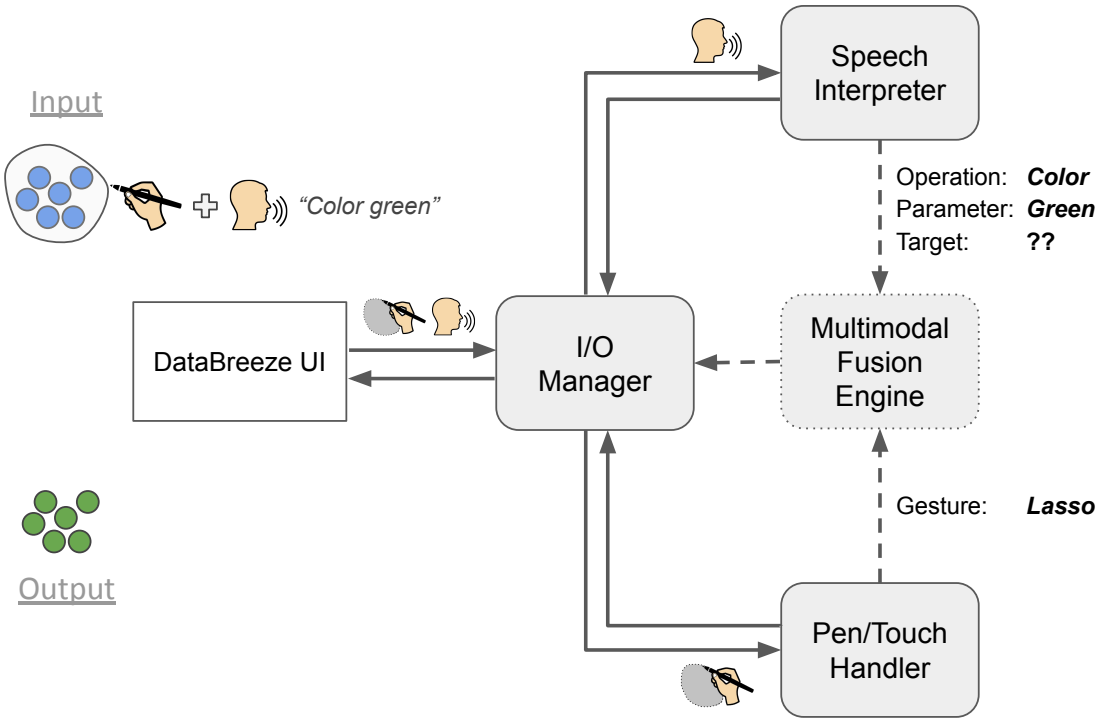



Figure 6.9: DATABREEZE System architecture highlighting the flow of information between different components for the exemplary interaction of coloring points by selecting them and saying "*Color green*."

Figure 6.9 gives an overview of the system architecture. At a high-level, all user input is collected by the *I/O Manager* which then propagates it to the *Speech Interpreter* or the *Pen/Touch Handler* depending on the input type. These components individually process the input and send the response with the required changes back to the I/O Manager which updates the DATABREEZE interface. For multimodal commands where these components are individually unable to process the input (e.g., the speech interpreter does not detect the required parameters to execute an operation), the extracted input details are passed to the *Multimodal Fusion Engine*. This component then combines the information from both input streams to determine the required system action and passes it back as a response to the I/O Manager.

To combine different input streams, the fusion engine uses a predefined set of mappings between operations (e.g., coloring, assigning axes), their relevant parameters (e.g., color

names, data attributes), and targets (e.g., data points). For the example shown in Figure 6.9, the user selects a set of nodes  with the pen (which implicitly triggers speech) and says "*Color green*." Using the keywords in the input command, the *Speech Interpreter* identifies that the operation being referred to is *Color* and the color to be used is *green* (the operation parameter). However, because the command does not specify any criteria to select points (missing target for the *Color* operation), the *Fusion Engine* also considers the input from the *Pen/Touch Handler*. Detecting that a selection (lasso) was performed, DATABREEZE infers the target points by considering the active selection state of points on the canvas. Upon detecting the selected points, the *Fusion Engine* combines this information with the output from the *Speech Interpreter* and updates the color of the selected points.

Compared to systems like ORKO that perform the fusion temporally, the fusion engine in DATABREEZE operates based on the semantics of the system state (e.g., selections, active filters) and the intended operation (e.g., creating axes, moving). This allows using speech and pen/touch simultaneously or sequentially (starting with either modality). For example, one can point  to a location on the canvas while speaking a command, or point and lift the finger to think about a command and then say it, both resulting in the same action. DATABREEZE also preserves gestures as part of the context objects across multimodal commands. For instance, if one swiped  and said "*Region*," DATABREEZE will set the axis to *Region*. Now if the user issues "*Locale*" as the next command, the fusion engine preserves the swipe gesture in memory and repeats the axis specification operation but with the new *Locale* attribute.

### 6.3 Preliminary User Study

We conducted a preliminary user study to gauge people's reactions to DATABREEZE, and more specifically to observe: (1) if and when people switch between systematically bound and manually customized views during data exploration and (2) the use of different modalities during data exploration with flexible unit visualizations.

### 6.3.1 Participants and Experimental Setup

We recruited six participants (P1-P6; four females, two males), aged 23 to 28. All participants were university students and indicated their field of study as computer science (P1), HCI (P2, P3, P5), visual art (P4), and cognitive science (P6). In terms of prior visualization experience, two participants (P1, P2) stated they had minimal experience working with visualization tools, three (P3, P4, P5) said they had worked with visualization tools on multiple occasions but not on a regular basis, and one participant (P6) said she was a frequent Tableau user. All participants were native English speakers and rated themselves as being moderately to highly comfortable working with touch-, pen-, and speech-based systems (e.g., on their iPads or Siri/Alexa). Participants interacted with DATABREEZE on Google's Chrome browser on an 84" Microsoft Surface Hub set to a 3840x2160 resolution with an external microphone to capture voice commands (as illustrated in Figure 6.4). All sessions were audio and video recorded.

### 6.3.2 Procedure

Participants were first given a brief introduction to DATABREEZE including the interface components, the interactions they could perform with pen/touch, and how they could invoke speech (5 min). To avoid biasing participants towards interacting in a particular way, we neither showed examples of speech commands nor gave them an exhaustive list of the available operations. Next, as "training," we directed participants to freely interact with DATABREEZE so they would be comfortable with the different interactions (10 min). We used a dataset about cars for the introduction and training phase. Participants were free to ask any questions they had regarding the interactions or system behavior during the training phase.

Participants then performed an open-ended task with the colleges dataset (with 500 U.S. colleges) in which they were asked to explore the data to produce a list of colleges for their younger siblings to apply to (15-30 min). Participants were free to leverage their

external knowledge of U.S. colleges as the shortlisting criteria. We did not ask participants to think-aloud because it could result in unintended recognition (due to the implicit speech triggering techniques) and also interrupt their workflow. We did, however, ask participants to take screenshots whenever they felt they identified a view that they would like to share with their family members (e.g., for discussion). Furthermore, we told participants to let us know whenever the system did not behave as they expected.

The study session ended with a debriefing in which we asked participants to provide feedback on their experience working with the system (5-15 min). We asked a set of standard questions across participants, but also seeded these interviews with our observations during the session (e.g., always using a specific modality for an operation). Overall, sessions lasted between 40-60 minutes and participants were compensated with a $20 Amazon Gift Card for their time.

### 6.3.3 Results and Observations

All participants completed the task, identifying at least ten colleges. Four participants identified multiple groups of colleges based on different criteria (P1, P3, & P5: two groups, P6: three groups). After identifying each group, these participants took a screenshot and reset the tool to start over. This resulted in 11 shortlisted groups of colleges across the six participants. Below, we highlight key observations from the study, focusing on the participants' data exploration patterns afforded by the combination of flexible unit visualizations and multimodal interaction.

#### *Visual Data Exploration Patterns*

We observed three high-level patterns that participants employed while exploring data with DATABREEZE.

The most common pattern was participants starting with a systematically bound view (**SB**), switching to a manually customized view (**MC**), but later switching back-and-forth

between the two views one or more times (**SB**→**MC**↔**SB**). We observed this pattern during five (out of 11) shortlists identified across four sessions (P1, P2, P4, P6). For example, P4 started with a scatterplot of *Average Cost* and *Median Earnings*. As he inspected colleges, he switched to a customized view where he removed points from the scatterplot and spatially categorized them into two virtual bins of "General" and "Field Specific" schools (**SB**→**MC**). Once he had narrowed down on a subset of colleges and refined his virtual bins by creating smaller, local scatterplots within the bins. He then specified a global X-axis based on *Region* (**MC**→**SB**), directly manipulating the resulting view to order points in a custom manner within each region (**SB**→**MC**). This seamless transition between systematically bound and manually customized views was enabled by DATABREEZE's support for global and local operations (**DG5**) as well as multimodal interaction (e.g., selecting a set of points and using swipe-and-speak to define a local axis, or issuing the same command without selecting points to create a global axis).

The second common pattern involved participants starting with a systematically bound view, then switching to a manually customized view and resorting to customized views until the end of their exploration (**SB**→**MC**). We observed this pattern during four shortlists across four sessions (P1, P3, P5, P6). For example, P5 created a scatterplot visualizing *Admission Rate* and *Population*, coloring points by *Locale*. From this scatterplot, she selected a set of colleges with lower *Admission Rate*s and ordered them by *Average Cost*. She continued to work with this subset of points creating new scatterplots with other attributes. However, while she worked with these points, she preserved her original scatterplot and would go back to it to explore a different set of points. This illustrates an interesting example similar to the one in Figure 6.1B, where the initial systematically bound view and global mappings become a platform to facilitate more localized exploration.

The last (i.e., least common) pattern comprised of participants exploring data exclusively using systematically bound views (**SB**). This pattern was employed during two shortlists, once each by P3 and P6. They created a scatterplot and iteratively refined it by filter-

ing points or modifying visual encodings until they identified a group of points that were interesting to them. This exploration strategy strongly aligns with Shneiderman's information seeking mantra [179] and is largely supported by shelf configuration and control panel-based visualization tools today.

*Multimodal Interaction Usage and Feedback*

Participants performed a total of 164 key operations (e.g., assign X/Y axes, change color, filter) across all sessions. As shown in Table 6.3, 37 (22%) of these interactions were performed using speech alone (e.g., "*Color by region*"), 43 (27%) were performed using only pen/touch and context menus (e.g., selecting points and using the context menu to remove them), and 84 (51%) involved a combination of pen/touch and speech input (e.g., select-and-speak, swipe-and-speak). Besides the 121 speech commands that were correctly recognized (37 speech-only + 84 multimodal), there were seven misrecognized commands (0-3 commands per session). Overall, regardless of their exploration strategy or pattern, all participants leveraged both touch/pen and speech (either unimodally or multimodally) to interact with the system.

Participants preferred speech to perform operations globally (e.g., color by attribute) but context menus to perform operations locally (e.g., coloring specific points red). When we asked participants about their choice to use context menus over select-and-speak (e.g.,  + "*Remove*"), participants said they felt a stronger sense of control with menus especially because local operations involved deletions or making fine-tuned changes that were better suited for dynamic querying [178] afforded by menus (e.g., changing color of points using a color picker).

In general, participant feedback suggested that their choice of interaction was primarily based on reducing the time to perform an operation. For instance, regardless of whether they were creating an axis for a subset of points (local operation) or for all points on the canvas (global operation), all participants used swipe-and-speak (e.g.,  + "*Admission*

Table 6.3: Summary of participants' interactions with DATABREEZE. Cells show the number of occurrences of an interaction (rows) for each participant (columns). Cells are colored column-wise from white (no interaction) to dark blue (most frequent interaction) for each participant.

| | P1 | P2 | P3 | P4 | P5 | P6 |
|---|---|---|---|---|---|---|
| Context Menu | 8 | 12 | 4 | | 12 | 7 |
| Speech | 7 | 7 | 6 | 2 | 7 | 8 |
| Swipe-and-Speak | 12 | 7 | 8 | 8 | 11 | 12 |
| Point-and-Speak | | 3 | 1 | | 4 | |
| Select-and-Speak | | | | 10 | 6 | 2 |
| **Total** | 27 | 29 | 19 | 20 | 40 | 29 |

*Rate*") attributing this to being both natural and fast. P4, for instance, said "*especially because the system was already listening to me when I swiped, it made more sense to say the attribute name than go to a menu and choose an attribute*." This comment also highlights the importance of subtle aspects such as triggering techniques for speech commands (**DG7**) during the design of multimodal systems.

## 6.4 Discussion and Future Work

The iterative design process and the preliminary user study helped us identify a number of key takeaways, raising additional design questions and highlighting avenues for future work. We discuss these points below.

### 6.4.1 Interweaving Interaction and Flexible Representation Techniques to Design Novel Tools and Experiences

DATABREEZE's flexibility stems from the fact that it interweaves multimodal interaction combining DM and NL with a particular visual representation (flexible unit visualizations) that provides suitable affordances for that style of interaction. For instance, talking about the ability to drag points out of a scatterplot during her interview, P5 said "*I really like that*

*flexibility. Being able to drag and drop points where you want mimics physical interaction like you would with documents on our table.*" This exemplifies how, in this case, the *directness of manipulation* [80] afforded by pen and touch naturally lent itself to participants expecting flexibility not supported in a common visual representation (dragging points in a scatterplot).

Another participant (P4) with a design background compared his experience using DATABREEZE to that of creating a mood board with a physical art board. In this context, he stated "*The system was great for that [exploring data similar to using an art board]. I could just quickly drag and pull things to create groups and categories that made sense in my head.*" Later, referring to his frequent usage of operations like coloring or filtering specific points through select-and-speech actions, he said, "*sometimes voice can be more of a novelty than a tool but in this case, it felt definitely like a tool. Where, if I had to otherwise keep going to some buttons and pressing them I would probably have used them less.*" Once again, such comments collectively suggest that the combination of the affordances of the visual representation and input/interaction techniques resulted in DATABREEZE supporting a workflow that is closer to how people interact with objects in the real-world (in P4's case, with an art board).

Although DATABREEZE is just one example of how this may be accomplished, it illustrates the potential of novel tools and experiences that can emerge by *interweaving interaction techniques and flexible representations*, adjusting each as necessary to enable a seamless user experience. While the two themes of interaction and representation have individually received considerable attention in visualization research today, far fewer examples have explored new tools and experiences that emerge from their synergistic integration (e.g., [32, 40, 104, 158]). With interactive devices and interfaces supporting alternative forms of input becoming a more common platform for visualizations, a compelling research opportunity lies in exploring visualization tools and user experiences stemming from the combination of naturalistic input/interaction techniques and more flexible visual

representations.

### 6.4.2 Leveraging Complementarity-based Multimodal Interaction

A key feature of DATABREEZE differentiating it from previous NL-based multimodal visualization systems is its increased support for *complementarity-based* multimodal interaction [124], where individual modalities are used to acquire different chunks of information which are then merged to enable a more sophisticated operation. For example, in a swipe-and-speak action, the operation of specifying the location and direction of an axis is specified through a pen/touch gesture whereas the designation of which data attribute to place on that axis is done through speech. Participants frequently performed such interactions and commented on them favorably. This was also reflected by the high number (84/164) of multimodal interactions during the user study. In fact, even participants who were initially hesitant about multimodal interaction (P3, P4, P6) quickly adapted to such interactions. For instance, P6 said "*Gestures typically in my mind aren't combined with voice commands [...] but once I got used to it, it was great and saved a lot of time.*" Referring to the swipe-and-speak action, P5 specifically noted that she found complementarity-based multimodal interaction to be most effective when there was a strong direct mapping between an operation and the input modality. As also highlighted earlier (**DG7**), an inherent consideration for supporting complementarity-based multimodal interaction was implicitly triggering speech input at the right time so that participants can more seamlessly integrate their actions across modalities. Along these lines, five out of the six participants (except P2) commented favorably on the implicit triggering techniques stating it made interacting with the system both fast and more natural.

This feedback suggests that if employed for the right operations, complementarity-based multimodal interaction can be a valuable feature in visualization systems by supporting a more fluid, integrated interaction experience [208], in turn, helping people preserve their workflow. Furthermore, spoken commands in complementarity-based interactions

are typically shorter and more focused (e.g., including only parameter values like attribute names or operation specific keywords like color, size, etc.). In addition to being easier for users to speak, from a system standpoint, these commands are generally easier to interpret [203]. With these potential user- and system-centered benefits in mind, an open area for future exploration lies in identifying operations and tasks that are best suited for complementarity-based multimodal interaction, as well as the appropriate interface support to mediate such interactions.

### 6.4.3 Improving System Feedback and Error Recovery

An important aspect of the interface design was to provide appropriate feedback in response to spoken commands (**DG4**). Correspondingly, we reserved an exclusive region in the interface for feedback directly under the speech input box (Figure 6.3B). Although the system presented different types of feedback, even suggesting corrections when possible, during the user study, participants often failed to notice the feedback. This was particularly problematic when there were command phrasing related errors. Since participants did not see the feedback, they ignored the system's phrasing suggestions and instead hyperarticulated their initial commands [135], resulting in the same error. Hence, an open area for improvement in DATABREEZE is to examine alternative feedback techniques that are more noticeable yet unobtrusive to the user's workflow.

A related point to feedback is error recovery. Similar to other speech-based mutlimodal systems (e.g., [199, 204]), DATABREEZE allows users to undo the most recent voice command. Going forward, it is important to implement a more complete undo stack, tackling associated challenges in doing so (e.g., managing scope [1], handling errors in undo command utterances [69]). With the flexibility of creating custom views and making global versus local changes, giving users the ability to backtrack multiple steps would further enhance the overall usability and user experience.

## 6.5 Conclusion

In this chapter, I described an approach interweaving DM- and NL-based multimodal interaction with flexible unit visualizations. Through the design and implementation of a prototype system, DATABREEZE, I highlight the design considerations and challenges in operationalizing this idea. Finally, based on observations from a preliminary user study with DATABREEZE, I illustrated how the proposed interweaving approach enables a free-form data exploration experience **[RG3]**. In addition to supporting this dissertation's overarching thesis, the findings from this chapter also shed light on novel tools and analytic workflows that can be supported by unifying interaction and representation techniques, adjusting each as necessary to enable a seamless user experience.

# CHAPTER 7

# HELPING DEVELOPERS PROTOTYPE NATURAL LANGUAGE-BASED VISUALIZATION SYSTEMS

The interaction flexibility in ORKO, INCHORUS, and DATABREEZE stems from having NL as an added input modality. These systems along with the growing suite of NLIs for visualization in both academic research (e.g., [62, 77, 172, 225]) and commercial software (e.g., [129, 195]) exhibit the promising potential of NL interaction with data visualization systems. However, creating NL-based visualization systems remains a challenging task. Besides implementing a graphical user interface (GUI) and rendering views, visualization system developers must also implement a natural language processing (NLP) module to interpret queries. While there is a collection of toolkits to support GUI and visualization design (e.g., D3.js [20], Vega-Lite [169]), developers currently have to implement custom modules for query interpretation. Unfortunately, for developers without experience with NLP tools and techniques, implementing this pipeline is non-trivial, requiring them to spend significant time and effort in learning and implementing different NLP techniques.

To mitigate this development challenge and aid prototyping of NL-based visualization systems, we developed the *Natural Language-Driven Data Visualization* (*NL4DV*) toolkit **[RG4]**. In this chapter, I detail NL4DV's implementation and design goals, discussing how the toolkit formalizes the inferred information into a JSON-based analytic specification that can be programmatically parsed. Through example applications, I showcase how this formalization helps: 1) develop NL- and DM-based multimodal visualization interfaces, 2) implement new NLIs for data visualization, and 3) support NL-based visualization specification in data science programming environments.

## 7.1 Challenges in Interpreting Natural Language Queries for Data Visualization

While NLIs provide flexibility in posing data-related questions, inherent characteristics of NL such as ambiguity and underspecification make implementing NLIs for data visualization a challenging task. Consider the spectrum of queries in Figure 7.1 issued to create visualizations in the context of an IMDb movies dataset with different attributes including the # Worldwide Gross, A Genre, and Release Year, among others (for consistency, we use this movies dataset for examples throughout this chapter). The query *"Create a histogram showing distribution of IMDB ratings"* (Figure 7.1a) explicitly refers to a data attribute (*IMDB Rating*), a low-level analytic task (*Distribution*), and requests a specific visualization type (*Histogram*). This is an ideal interpretation scenario from a system standpoint since the query explicitly lists all components required to generate a visualization.

On the other hand, the second query *"Show average gross across genres for the science fiction and fantasy movies"* (Figure 7.1b) does not explicitly state the visualization type or the attribute *Creative Type*. Instead, it explicitly references the attributes *Worldwide*

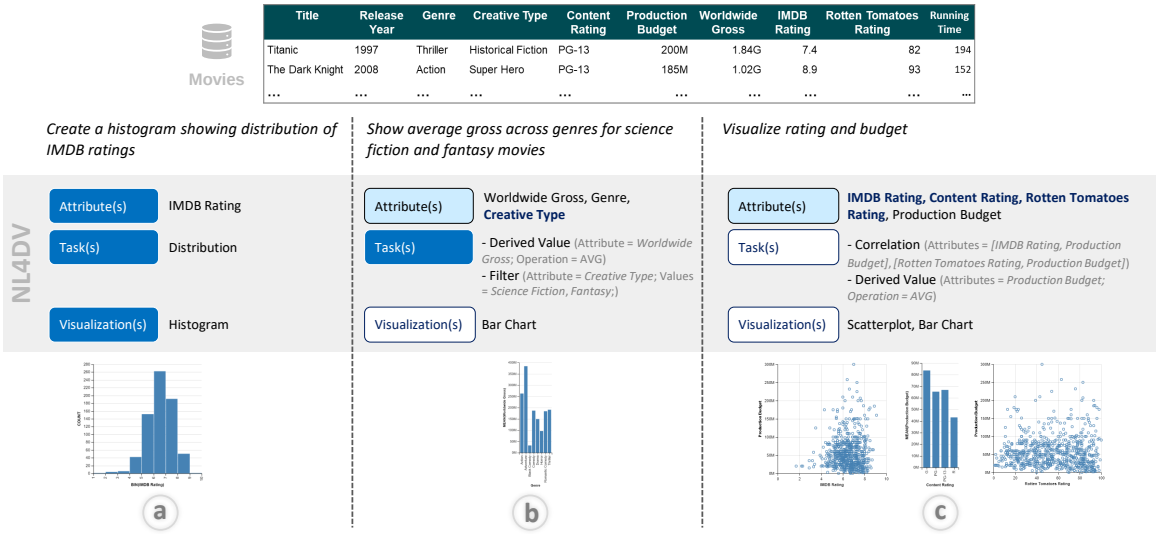



Figure 7.1: Examples illustrating the flexibility of natural language queries for specifying data visualizations. NL4DV processes all three query variations, inferring explicit, partially explicit or ambiguous, and implicit references to attributes, tasks, and visualizations. The corresponding visualizations suggested by NL4DV in response to the individual queries are also shown.

*Gross* and *Genre* through 'gross' and 'genres', and implicitly refers to the *Creative Type* through the values 'science fiction' and 'fantasy'. Furthermore, by specifying data values for the *Creative Type* attribute and the word 'average,' the query also mentions two intended analytic tasks: *Filtering* and computing *Derived Values*, respectively. This second query is more challenging since it requires the system to implicitly infer one of the attributes and then determine the visualization type based on the identified attributes and tasks.

Finally, the third query *"Visualize rating and budget"* (Figure 7.1c) is even more challenging to interpret since it neither explicitly states the desired visualization type nor the intended analytic task. Furthermore, while it explicitly references one attribute (*Production Budget* through 'budget'), the reference to the second attribute is ambiguous ('rating' can map to *IMDB Rating*, *Content Rating*, or *Rotten Tomatoes Rating*).

To accommodate such query variations, visualization NLIs employ sophisticated NLP techniques (e.g., dependency parsing, semantic word matching) to identify relevant information from the query and build upon visualization concepts (e.g., analytic tasks) and design principles (e.g., choosing graphical encodings based on attribute types) to generate appropriate visualizations. For instance, given the query in Figure 7.1b, after detecting the data attributes and analytic tasks, a visualization NLI should select a visualization (e.g., bar chart) that is well-suited to support the task of displaying *Derived Values* (average) for *Worldwide Gross* (a **#** quantitative attribute) across different *Genre*s (a **A** nominal attribute). Similarly, in the scenario in Figure 7.1c, a NLI must first detect ambiguities in the input query attributes, determine the visualizations suited to present those attribute combinations (e.g., *scatterplot* for two quantitative attributes), and ultimately infer the analytic tasks based on those attributes and visualizations (e.g., a scatterplot may imply the user is interested in finding *correlations*).

## 7.2 NL4DV Overview and Scope

Figure 7.2 presents an overview of a typical pipeline for implementing NLIs that generate visualizations in response to NL queries. At a high-level, once an input query is collected through a *User Interface*, a *Query Processor* infers relevant information such as data attributes and analytic tasks from the input query. This information is then passed to a *Visualization Recommendation Engine* which generates a list of visualizations specifications relevant to the input query. These specifications are finally rendered through a library (e.g., D3 [20]) of the developer's choice.

Our primary objective with NL4DV is to help visualization developers prototype NLIs. In terms of the goals described in Chapter 3, NL4DV is primarily geared to support *visualization generation* but it can also be used to support some simple interactions with an active chart (e.g., filtering, sorting). With this scope in mind, NL4DV provides a high-level API for processing NL queries to extract relevant information like attributes and tasks. Furthermore, NL4DV also includes a built-in visualization recommendation engine that internally leverages the extracted information to return an ordered list of Vega-Lite specifications relevant to the input query. Developers can choose to directly render the Vega-Lite specifications to create views (e.g., using Vega-Embed [202]) or use the attributes and tasks inferred by NL4DV to make custom changes to their system's interface.

### 7.2.1 Design Goals

Four key design goals drove the development of NL4DV. We compiled these goals based on a review of design goals and system implementations of ORKO, INCHORUS, DATABREEZE, prior visualization NLIs [62, 77, 93, 172, 193, 225], and recent toolkits for supporting visualization development on new platforms and modalities (e.g, [171, 180]).

**DG1. Minimize NLP learning curve.** NL4DV's primary target users are developers without a background or experience in working with NLP techniques. Correspondingly, it was

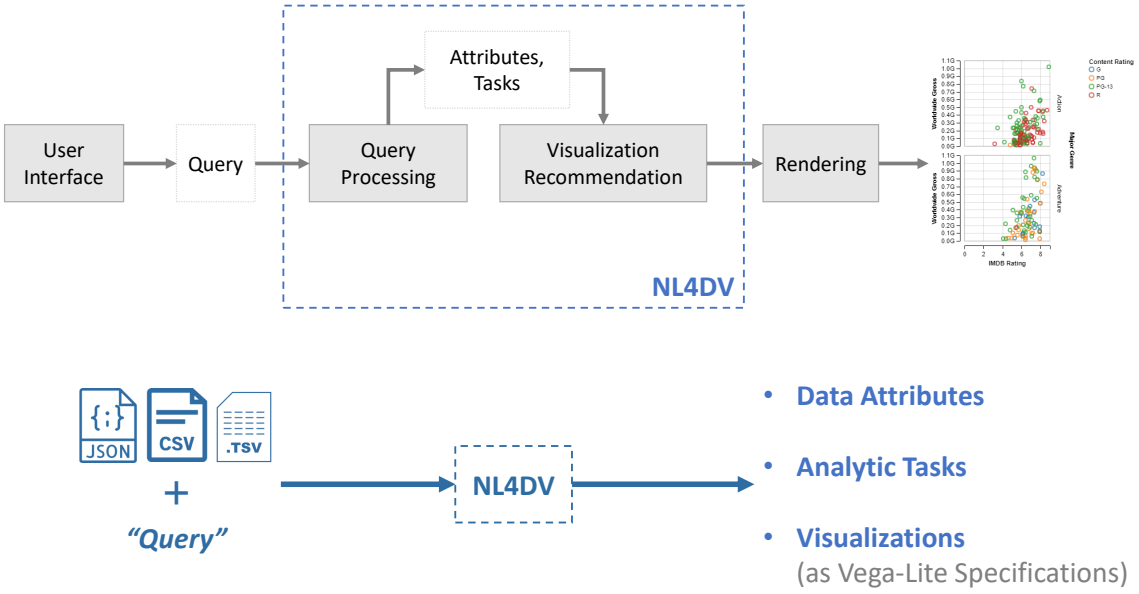



Figure 7.2: (Top) A high-level overview of steps involved in generating visualizations based on NL queries. NL4DV encapsulates the query processing and visualization recommendation components, providing abstract functions to support their functionality. (Bottom) Once initialized with a dataset, NL4DV parses input NL queries and returns relevant information (in terms of data attributes and analytic tasks) and an ordered list of Vega-Lite specifications.

important to make the learning curve as flat as possible. In other words, we wanted to enable developers to use the output of NL4DV without having to spend time learning about the mechanics of how information is extracted from NL queries. In terms of toolkit design, this consideration translated to providing high-level functions for interpreting NL queries and designing a response structure that was optimized for visualization system development by emphasizing visualization-related information such as analytic tasks (e.g., filter, correlation) and data attributes and values.

**DG2. Generate modularized output and support integration with existing system components.** By default, NL4DV recommends Vega-Lite specifications in response to NL queries. However, a developer may prefer rendering visualizations using a different library such as D3 or may want to use a custom visualization recommendation engine (e.g., [113, 132, 211]), only leveraging NL4DV to identify attributes and/or tasks in the input query.

Supporting this goal required us to ensure that NL4DV's output was modularized (allowing developers to choose if they wanted attributes, tasks, and/or visualizations) and that developers do not have to notably modify their existing system architecture to use NL4DV. In terms of toolkit design, in addition to using a standardized grammar for visualizations (in our case, Vega-Lite), these considerations translated to devising a formalized representation of data attributes and analytic tasks that developers can programmatically parse to link NL4DV's output to other components in their system.

**DG3. Highlight inference type and ambiguity.** NL is often underspecified and ambiguous. In other words, input queries may only include partial references to data attributes or may implicitly refer to intended tasks and visualizations [198] (e.g., Figure 7.1b, c). Besides addressing these challenges from an interpretation standpoint, it was also important to make developers aware of the resulting uncertainty in NL4DV's output so they can choose to use/discard the output and provide appropriate visual cues (e.g., ambiguity widgets [62]) in their systems' interface. In terms of toolkit design, this translated to structuring NL4DV's output so it indicates whether information is inferred through an explicit (e.g., query substring matches an attribute name) or implicit (e.g., query refers to a data attribute through the attribute's values) reference, and highlights potential ambiguities in its response (e.g., two or more attributes map to the same word in an input query as in Figure 7.1c).

**DG4. Support adding aliases and overriding toolkit defaults.** Visualization systems are frequently used to analyze domain-specific datasets (e.g., sales, medical records, sports). Given a domain, it is common for data attributes to have abbreviations or aliases (e.g., "GDP" for *Gross domestic product*, "investment" for *Capital*), or values that are unique to the dataset (e.g., the letter "A" can refer to a value in a course grade dataset, but would be considered as a stopword and ignored by most NLP algorithms by default). In terms of toolkit design, these dataset-specific considerations translated to providing developers with helper functions to specify aliases or special word lists that NL4DV should consider/exclude for a given dataset.

## 7.3 NL4DV Design and Implementation

Listing 1 shows the basic Python code for using NL4DV. Given a query string, with a single function call **`analyze_query(query)`**, NL4DV infers attributes, tasks, and visualizations, returning them as a JSON object (**DG1**). Specifically, NL4DV's response object has an **`attributeMap`** composed of the inferred dataset attributes, a **`taskMap`** composed of the inferred analytic tasks, and a **`visList`**, a list of visualization specifications relevant to the input query. By providing attributes, tasks, and visualizations as separate keys in the response object, NL4DV allows developers to selectively extract and use parts of its output (**DG2**).

### 7.3.1 Data Interpretation

Once initialized with a dataset (Listing 1, line 2), NL4DV iterates through the underlying data item values to infer metadata including the attribute types (**#** ***Q***uantitative, **A** ***N***ominal, ***O***rdinal, ***T***emporal) along with the range and domain of values for each attribute. This attribute metadata is used when interpreting queries to infer appropriate analytic tasks and generate relevant visualization specifications.

Since NL4DV uses data values to infer attribute types, it may make erroneous interpre-

```
1   from nl4dv import NL4DV
2   nl4dv_instance = NL4DV(data_url="movies.csv")
3   response = nl4dv_instance.analyze_query("Show the relationship
    ↪  between budget and rating for Action and Adventure movies
    ↪  that grossed over 100M")
4   print(response)
```

```
{
    "attributeMap": { ... },
    "taskMap": { ... },
    "visList": [ ... ]
}
```

Listing 1: Python code illustrating NL4DV's basic usage involving initializing NL4DV with a dataset (line 2) and analyzing a query string (line 3). The high-level structure of NL4DV's response is also shown.

tations. For example, a dataset may have the attribute *Day* with values in the range $[1, 31]$. Detecting a range of integer values, by default, NL4DV will infer *Day* as a quantitative attribute instead of temporal. This misinterpretation can lead to NL4DV making poor design choices when selecting visualizations based on the inferred attributes (e.g., a quantitative attribute may result in a histogram instead of a line chart). To overcome such issues caused by data quality or dataset semantics, NL4DV allows developers to verify the inferred metadata using `get_metadata()`. This function returns a hash map of attributes along with their inferred metadata. If they notice errors, developers can use other helper functions (e.g., `set_attribute_type(attribute, type)`) to override the default interpretation (**DG4**).

### 7.3.2 Query Interpretation

To generate visualizations in response to a query, visualization NLIs need to identify informative phrases in the query that map to relevant concepts like data attributes and values, analytic tasks, and visualization types, among others. Figure 7.3 shows the query in Listing 1 "*Show the relationship between budget and rating for Action and Adventure movies that grossed over 100M*" with such annotated phrases (we use this query as a running example throughout this section to describe NL4DV's query interpretation strategy). To identify relevant phrases and generate the `attributeMap`, `taskMap`, and `visList`, NL4DV performs four steps: 1) *query parsing*, 2) *attribute inference*, 3) *task inference*, and 4) *vi-*

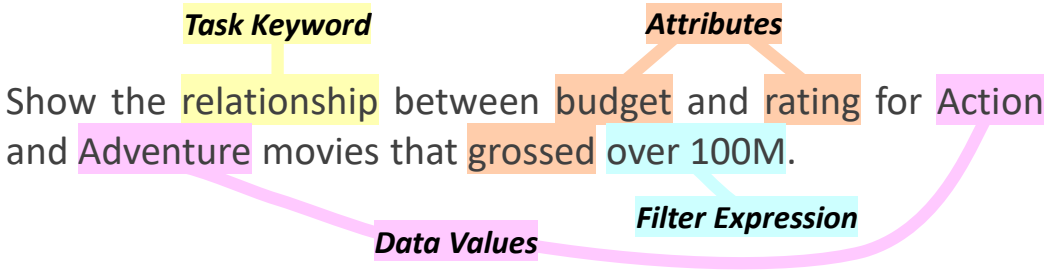



Figure 7.3: An illustration of query phrases that NL4DV identifies while interpreting NL queries.

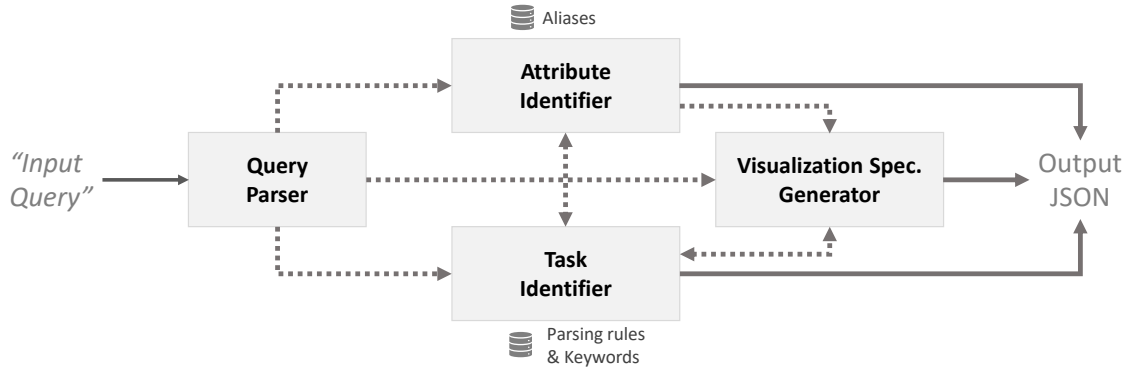



Figure 7.4: NL4DV's architecture. The arrows indicate the flow of information between different modules.

*sualization specification generation*. Figure 7.4 gives an overview of NL4DV's underlying architecture. Below we describe the individual query interpretation steps (task inference is split into two steps to aid explanation) and summarize the pipeline in Figure 7.5.

*Query Parsing*

The query parser runs a series of NLP functions on the input string to extract details that can be used to detect relevant phrases. In this step, NL4DV first preprocesses the query to convert any special symbols or characters into dataset-relevant values (e.g., converting *100M* to the number *100000000*). Next, the toolkit identifies the POS tags for each token (e.g., *NN*: Noun, *JJ*: Adjective, *CC*: Coordinating Conjunction) using Stanford's CoreNLP [122]. Furthermore, to understand the relationship between different phrases in the query, NL4DV uses CoreNLP's dependency parser to create a dependency tree. Then, with the exception of conjunctive/disjunctive terms (e.g., 'and', 'or') and some prepositions (e.g., 'between', 'over') and adverbs (e.g., 'except', 'not'), NL4DV trims the input query by removing all stop words and performs stemming (e.g., '*grossed*' $\rightarrow$ '*gross*'). Lastly, the toolkit generates all *N-grams* from the trimmed query string. The output from the query parser (POS-tags, dependency tree, N-grams) is shown in Figure 7.5a and is used internally by NL4DV during the remaining stages of query interpretation.

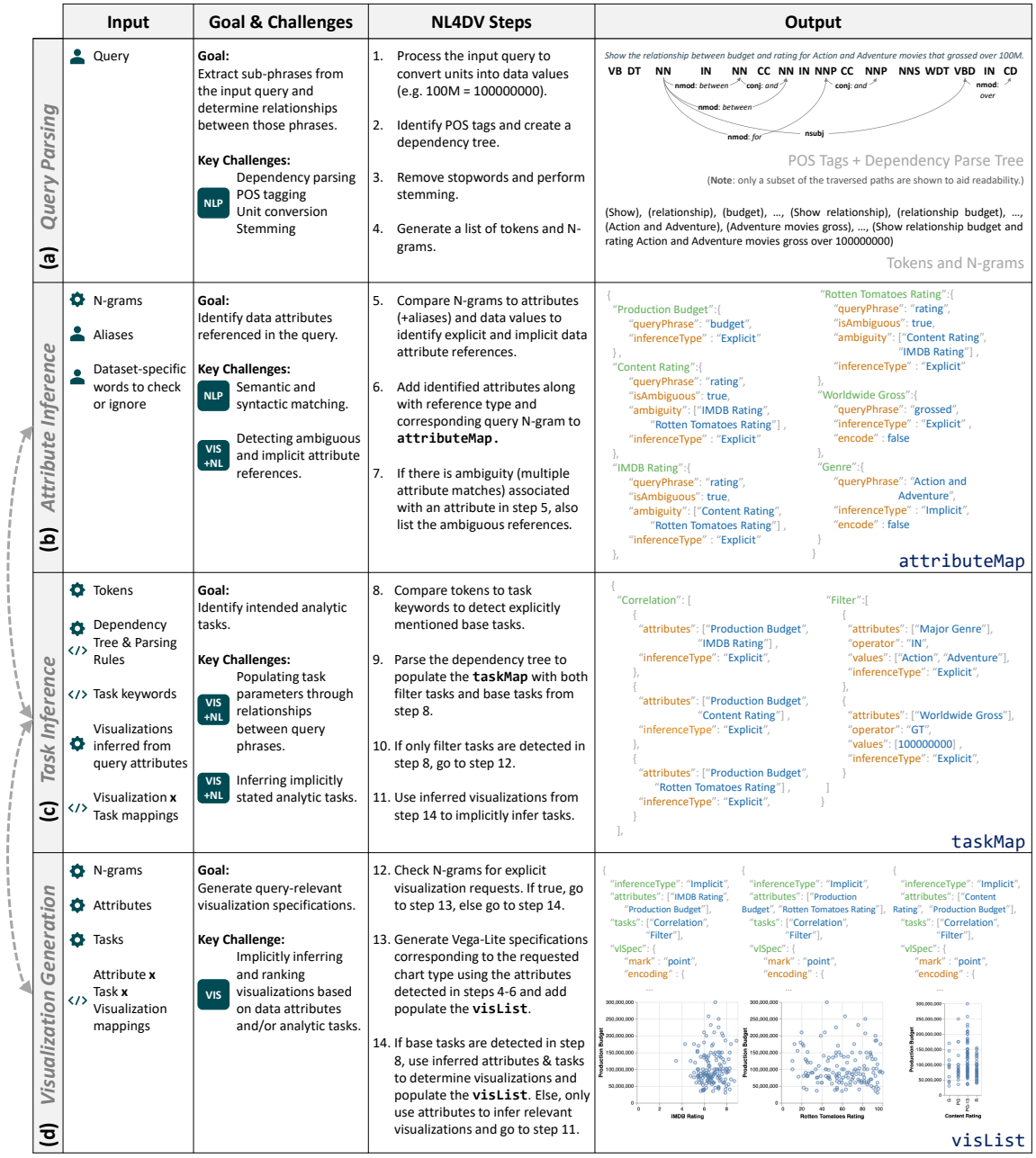



Figure 7.5: Summary of NL4DV's query interpretation pipeline. Information flows sequentially across stages unless explicitly indicated by bi-directional arrows. Input to different stages is provided externally by developers, generated by other stages of NL4DV's pipleline, or preconfigured into NL4DV. The figure also highlights the key goal and implementation challenges for each stage (NLP: general NLP challenge, VIS+NL: challenge specific to visualization NLIs, VIS: visualization design/recommendation challenge). NL4DV internally tackles these challenges, providing visualization developers with a high-level API for query interpretation.

*Attribute Inference*

After parsing the input query, NL4DV looks for data attributes that are mentioned both *explicitly* (e.g., through direct references to attribute names) and *implicitly* (e.g., through references to an attribute's values). Developers can also configure aliases (e.g., 'Investment' for *Production Budget*) to support dataset- and domain-specific attribute references (**DG4**). To do so, developers can provide a JSON object consisting of attributes (as keys) and lists of aliases (as values). This object can be passed through the optional parameters **alias_map** or **alias_map_url** when initializing NL4DV (Listing 1, line 2) or using the helper function `set_alias_map(alias_map, url="")`.

To infer attributes, NL4DV iterates through the N-grams generated by the query parser, checking for both syntactic (e.g., misspelled words) and semantic (e.g., synonyms) similarity between N-grams and a lexicon composed of data attributes, aliases, and values. To check for syntactic similarity, NL4DV computes the cosine similarity $Sim_{\text{cos}}(i, j)$ between a N-gram *i* and a tokenized lexical entity *j*. The possible values for $Sim_{\text{cos}}(i, j)$ range from [0,1] with 1 indicating that strings are equivalent. For semantic similarity, the toolkit checks for the Wu-Palmer similarity score [216] $Sim_{\text{wp}}(i, j)$ between a N-gram *i* and a tokenized lexical entry *j*. This score returns the distance between stemmed versions of *p* and *a* in the WordNet graph [130], and is a value in the range $(0, 1]$, with higher values implying greater similarity. If $Sim_{\text{cos}}(i, j)$ or $Sim_{\text{wup}}(i, j) \geq 0.8$, NL4DV maps the N-gram *i* to the attribute corresponding to *j*, also adding the attribute as a key in the `attributeMap`.

As shown in Figure 7.5b, the `attributeMap` is structured such that besides the attributes themselves, for each attribute, developers can also identify: (1) query substrings that led to an attribute being detected (`queryPhrase`) along with (2) the type of reference (`inferenceType`), and (3) ambiguous matches (`ambiguity`). For instance, given the query in Figure 7.3, NL4DV detects the attributes Production Budget (based on 'budget'), Content Rating, IMDB Rating, Rotten Tomatoes Rating (`ambiguity` caused by the word 'rating'), Worldwide Gross (based on 'grossed'), and Genre (based on the values 'Action'

and 'Adventure'). Furthermore, since *Genre* is referenced by its values, it is marked as Implicit whereas the other attributes are marked as Explicit (**DG3**).

*Explicit Task Inference*

In addition to data attributes, NL4DV also checks the N-grams for references to analytic tasks. NL4DV currently identifies five low-level analytic tasks [3] including four base tasks: *Correlation*, *Distribution*, *Derived Value*, *Trend*, and a fifth *Filter* task. We separate base tasks from filter since base tasks are used to determine appropriate visualizations (e.g., correlation maps to a scatterplot) whereas filters are applied across different types of visualizations. We focus on these five tasks as a starting set since they are commonly supported in prior NLIs [62, 89, 173, 193, 225] and are most relevant to NL4DV's primary goal of supporting visualization specification through NL (as opposed to interacting with a given chart [172] or question answering [93]).

While filters may be detected via data values (e.g., 'Action', 'Comedy'), to detect base tasks, NL4DV compares the query tokens to a predefined list of task keywords (e.g., 'correlate', 'relationship', etc. for the *Correlation* task, 'range', 'spread', etc. for the *Distribution* task, 'average', 'sum', etc. for *Derived Value*). We defined these keywords based on descriptions and examples from prior visualization NLIs [62, 77, 172, 193, 225] as well as ∼200 questions collected by Amar et al. [3] when formulating their analytic task taxonomy.

Merely detecting references to attributes, values, and tasks is insufficient to infer user intent, however. To model relationships between query phrases and populate task details, NL4DV leverages the POS-tags and the dependency tree generated by the query parser. Specifically, using the token type and dependency type (e.g., `nmod`, `conj`, `nsubj`) and distance, NL4DV identifies mappings between attributes, values, and tasks. These mappings are then used to model the **`taskMap`**.

The **`taskMap`** contains analytic tasks as keys. Tasks are broken down as a list of objects that include an `inferenceType` field to indicate if a task was stated explicitly (e.g.,

through keywords) or derived implicitly (e.g., if a query requests for a line chart, a *trend* task may be implied) and parameters to apply when executing a task. These include the `attributes` a task maps to, the `operator` to be used (e.g., GT, EQ, AVG, SUM), and `values`. If there are ambiguities in task parameters (e.g., the word 'fiction' may refer to the values 'Science Fiction,' 'Contemporary Fiction,' 'Historical Fiction'), NL4DV adds additional fields (e.g., `isValueAmbiguous`=true) to highlight them (**DG3**). In addition to the tasks themselves, this structuring of the **`taskMap`** allows developers to detect: (1) the parameters needed to execute a task (`attributes`, `operator`, `values`), (2) operator- and value-level ambiguities (e.g., `isValueAmbiguous`), and (3) if the task was stated explicitly or implicitly (`inferenceType`).

Consider the **`taskMap`** (Figure 7.5c) for the query in Figure 7.3. Using the dependency tree in Figure 7.5a, NL4DV infers that the word 'relationship' maps to the Correlation task and links to the tokens 'budget' and 'rating' which are in-turn linked by the conjunction term 'and.' Next, referring back to the **`attributeMap`**, NL4DV maps the words 'budget' and 'rating' to their respective data attributes, adding three objects corresponding to correlations between the `attributes` [Production Budget, IMDB Rating], [Production Budget, Content Rating], and [Production Budget, Rotten Tomatoes Rating] to the *correlation* task. Leveraging the tokens 'Action' and 'Adventure', NL4DV also infers that the query refers to a Filter task on the `attribute` Genre, where the `values` are *in the list* (IN) [Action, Adventure]. Lastly, using the dependencies between tokens in the phrase '*gross over 100M*,' NL4DV adds an object with the `attribute` Worldwide Gross, the *greater than* (GT) `operator`, and 100000000 in the `values` field. While populating *filter* tasks, NL4DV also updates the corresponding attributes in the **`attributeMap`** with the key `encode`=False (Figure 7.5b). This helps developers detect that an attribute is used for filtering and is not visually encoded in the recommended charts.

*Visualization Generation*

NL4DV uses Vega-Lite as the underlying visualization grammar. The toolkit currently supports the Vega-Lite `marks`: bar, tick, line, area, point, arc, boxplot, text and `encodings`: x, y, color, size, column, row, theta to visualize up to three attributes at a time. This combination of marks and encodings allows NL4DV to support a range of common visualization types including *histograms*, *strip plots*, *bar charts* (including stacked and grouped bar charts), *line* and *area charts*, *pie charts*, *scatterplots*, *box plots*, and *heatmaps*. To determine visualizations relevant to the input query, NL4DV checks the query for explicit requests for visualization types (e.g., Figure 7.1a) or implicitly infers visualizations through attributes and tasks (e.g., Figures 7.1b, 7.1c, and 7.3).

Explicit visualization requests are identified by comparing query N-grams to a predefined list of visualization keywords (e.g., 'scatterplot', 'histogram', 'bar chart'). For instance, the query in Figure 7.1a specifies the visualization type through the token 'histogram,' leading to NL4DV setting bar as the `mark` type and binned IMDB Rating as the `x encoding` in the underlying Vega-Lite specification.

To implicitly determine visualizations, NL4DV uses a combination of the attributes and tasks inferred from the query. NL4DV starts by listing all possible visualizations using the detected attributes by applying well-known mappings between attributes and visualizations (Table 7.1). These mappings are preconfigured within NL4DV based on heuristics used in prior systems like Show Me [119] and Voyager [212, 213]. As stated earlier, when generating visualizations from attributes, NL4DV does not visually encode the attributes used as filters. Instead, filter attributes are added as a `filter transform` in Vega-Lite. Doing so helps avoid a combinatorial explosion of attributes when a query includes multiple filters (e.g., including the filter attributes for the query in Figure 7.3 would require generating visualizations that encode four attributes instead of two).

Besides attributes, if tasks are explicitly stated in the query, NL4DV uses them as an additional metric to modify, prune, and/or rank the generated visualizations. Consider the

Table 7.1: Attribute (+encodings), visualization, and task mappings preconfigured in NL4DV. Attributes in curly brackets {are optional}. Note that these defaults can be overridden via explicit queries. For instance, "*Show average gross across genres as a scatterplot*" will create a scatterplot instead of a bar chart with *Genre* on the `x`- and AVG(*Worldwide Gross*) on the `y`-axis. For unsupported attribute combinations and tasks, NL4DV resorts to a table-like view created using Vega-Lite's *text* `mark`.

| **Attributes (`x`, `y`, `color`/`size`/`row`/`column`)** | **Visualizations** | **Task** |
|---|---|---|
| Q x Q x {N, O, Q, T} | Scatterplot | Correlation |
| N, O x Q x {N, O, Q, T} | Bar Chart | Derived Value |
| Q, N, O x {N, O, Q, T} x {Q} | Strip Plot, Histogram, Bar Chart, Heatmap | Distribution |
| T x {Q} x {N, O} | Line Chart | Trend |

query in Figure 7.3. Similar to the query in Figure 7.1c, if only attributes were used to determine the charts, NL4DV would output two scatterplots (for *Q*x*Q*) and one bar chart (for *N*x*Q*). However, since the query contains the token 'relationship,' which maps to a *Correlation* task, NL4DV enforces a scatterplot as the chart type, setting the `mark` in the Vega-Lite specifications to point. These Task x Visualization mappings (Table 7.1) are again heuristically set within NL4DV based on prior visualization systems [28, 64, 132] and studies [96, 162]. Furthermore, since correlations are more apparent in *Q*x*Q* charts, NL4DV also ranks the two *Q*x*Q* charts higher, returning the three visualization specifications shown in Figure 7.5d.

NL4DV complies the inferred visualizations into a **`visList`** (Figure 7.5d). Each object in this list is composed of a vlSpec containing the Vega-Lite specification for a chart, an inferenceType field to highlight if a visualization was requested explicitly or implicitly inferred by NL4DV, and a list of attributes and tasks that a visualization maps to. Developers can use the **`visList`** to directly render visualizations in their systems (via the vlSpec). Alternatively, ignoring the **`visList`**, developers can also extract only attributes and tasks using the **`attributeMap`** and **`taskMap`**, and feed them as input to other visualization recommendation engines of their choice [113, 211] (**DG2**).

*Implicit Task Inference*

When the input query lacks explicit keywords referring to analytic tasks, NL4DV first checks if the query requests for a specific visualization type. If so, the toolkit uses mappings between Visualizations x Tasks in Table 7.1 to infer tasks (e.g., *distribution* for a histogram, *trend* for a line chart, *correlation* for a scatterplot).

Alternatively, if the query only mentions attributes, NL4DV first lists possible visualizations based on those attributes. Then, using the inferred visualizations, the toolkit implicitly infers tasks (again leveraging the Visualization x Task mappings in Table 7.1). Consider the example in Figure 7.1c. In this case, the tasks *Correlation* and *Derived Value* are inferred based on the two scatterplots and one bar chart generated using the attribute combinations $Q$x$Q$ and $N$x$Q$, respectively. In such cases where the tasks are implicitly inferred through visualizations, NL4DV also sets their `inferenceType` in the **`taskMap`** to Implicit.

## 7.4 Example Applications

In this section we describe how NL4DV's output can be used to develop both NL-only and multimodal NL- and DM-based visualization systems.

### 7.4.1 Augmenting a DM-based Visualization Tool with NL

Consider TOUCHPLOT (Figure 7.6-top), a tablet-based scatterplot visualization system we developed. We modeled TOUCHPLOT after the interface and capabilities of the multi-touch scatterplot visualization system, Tangere [159]. Specifically, users can select points and zoom/pan by interacting directly with the chart canvas, bind attributes to the position, color, and size encodings using dropdown menus on axes and legends, or apply filters using a side panel. TOUCHPLOT is implemented using HTML and JavaScript, and D3 is used for creating the visualization.

With INCHORUS, we showed how complementing touch interactions with speech can support a more fluid interaction experience during visual analysis on tablets. For example, while touch can support fine-grained interactions with marks, speech can allow specifying filters without having to open and interact with the side panel, saving screen space and preserving the user workflow. To explore such fluid interactions, we developed MMPLOT (Figure 7.6-bottom), a modified version of TOUCHPLOT that supports multimodal touch and speech input. In addition to touch interactions, MMPLOT allows issuing speech commands to specify charts (e.g., "*Correlate age and salary by country*") and filter points (e.g., "*Remove players over the age of 30*").


Figure 7.6: (Top) TOUCHPLOT interface supporting interaction through touch and control panels. (Bottom) MMPLOT interface supporting multimodal interactions. Here, the user has specified a new scatterplot and applied a filter through a single query "*Show a scatter plot of age and salary for players under the age of 30.*"

```
1       $.post("/analyzeQuery", {"query": query})
2         .done(function (responseString) {
3           let nl4dvResponse = JSON.parse(responseString);
4           let taskMap = nl4dvResponse['taskMap'],
5               visList = nl4dvResponse['visList'];
6           if("filter" in taskMap){ // query includes a filter
7             for(let taskObj of taskMap['filter']){
8               for(let attr of taskObj['attributes']){
9                 // use the attribute type, 'operator', and 'values' to apply requested
                  ↪ filters
10              }
11            }
12          }
13          if(visList.length>0){ // query specifies a new chart
14            let newVisSpec = visList[0]['vlSpec'];
15            // check if newVisSpec is a scatterplot configuration supported by the system
                ↪ and modify the attribute-encoding mappings
16          }
17          // invoke the D3 code to update the view
18        });
```

Listing 2: JavaScript code to parse NL4DV's output for supporting speech and multimodal interactions in MMPLOT (Figure 7.6-bottom).

To support these interactions, we record speech input and convert it to a string using the Web Speech API [209]. This query string is then passed to the server, where we make a call to NL4DV's **`analyze_query(query)`**. By parsing NL4DV's response in JavaScript, TOUCHPLOT is modified to support the required speech interactions (Listing 2). In particular, we parse the **`taskMap`** to detect and apply any filters requested as part of the query (lines 6-12). Next, we check if the input query specifies a new scatterplot that can be rendered by the system and adjust the view mappings accordingly (lines 13-16). This sequential parsing of **`taskMap`** and **`visList`** allows using speech to apply filters, specify new scatterplots, or do both with a single query (Figure 7.6). Unlike previous examples, since this application uses D3 (as opposed to Vega-Lite) to create the visualization, when parsing NL4DV's output, we perform an added step of invoking the D3 code required to update the view (line 17).

### 7.4.2 Recreating Ambiguity Widgets in DataTone

Consider a second example where we use NL4DV to replicate features of the DataTone system by Gao et al. [62]. Given a NL query, DataTone identifies ambiguities in the query

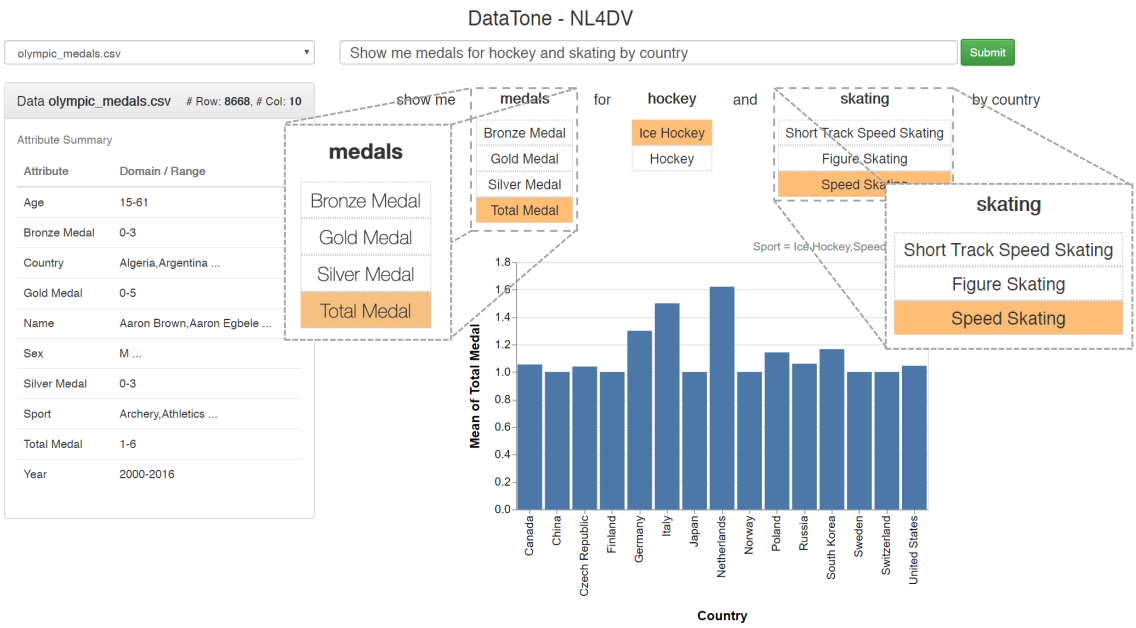



Figure 7.7: A sample interface illustrating how NL4DV can be used to replicate DataTone's [62] ambiguity widgets.

```
$.post("/analyzeQuery", {"query": query})
  .done(function (responseString) {
    let nl4dvResponse = JSON.parse(responseString);
    let attributeMap = nl4dvResponse['attributeMap'],
    taskMap = nl4dvResponse['taskMap'];
    for(let attr in attributeMap){
      if(attributeMap[attr]['isAmbiguous']){
        // add attr and attributeMap[attr]['ambiguity'] to attribute-level
        ↪  ambiguity widget corresponding to the attributeMap[attr]['queryPhrase']
      }
    }
    ...
  });
```

Listing 3: JavaScript code to parse NL4DV's output and generate attribute-level ambiguity widgets (highlighted in Figure 7.7-left). A similar logic is used to iterate over the **`taskMap`** when creating value-level ambiguity widgets (highlighted in Figure 7.7-right) for filtering.

and surfaces them via "ambiguity widgets" (e.g., dropdown menus) that users can interact with (through DM) to clarify their intent.

Figure 7.7 shows a DataTone-like interface implemented using NL4DV. This system is also implemented as a Flask web-application using HTML and JavaSript on the client-side. The example in Figure 7.7 illustrates the result of executing the query "*Show me*

*medals for hockey and skating by country*" against an Olympics medal winners dataset[1]. Here, 'medals' is an ambiguous reference to four data attributes corresponding to the three medal types (e.g., *Gold Medals*) and the *Total Medals*. Similarly, 'hockey' and 'skating' are value-level ambiguities corresponding to the *Sport* attribute (e.g., 'hockey'=[*Ice Hockey*, *Hockey*]).

Similar to the Vega-Lite editor application, the server-side code only involves initializing NL4DV with the active dataset and making a call to the **`analyze_query(query)`** function to process user queries. As detailed in Listing 3, on the client-side, to highlight attribute- and value-level ambiguities in the query, we parse the **`attributeMap`** and **`taskMap`** returned by NL4DV in JavaScript, checking the `isAmbiguous` fields. Vega-Embed is once again used to render the `vlSpecs` returned as part of NL4DV's **`visList`**. Note that we only focus on data ambiguity widgets in this example, not displaying design ambiguity widgets (e.g., dropdown menus for switching between visualization types). To generate design ambiguity widgets, however, developers can parse the **`visList`**, converting the Vega-Lite `marks` and `encodings` into dropdown menu options.

### 7.4.3 Using NL4DV in Jupyter Notebook

The above examples showcase how NL4DV can be used to create visualization systems supporting both NL and DM input. However, NL4DV can also be used to support direct NL-based visualization specification without the need for added interactions. In particular, since NL4DV generates Vega-Lite specifications, in environments that support rendering Vega-Lite charts, the toolkit can be used to directly specify visualizations through NL in Python. NL4DV provides a wrapper function **`render_vis(query)`** that automatically renders the first visualization in the **`visList`**. By rendering visualizations in response to NL queries in environments like Jupyter Notebook, NL4DV enables novice Python data scientists and programmers to conduct visual analysis without needing to learn about vi-

[1]This is the same query used to illustrate the concept of ambiguity widgets in the DataTone paper [62].

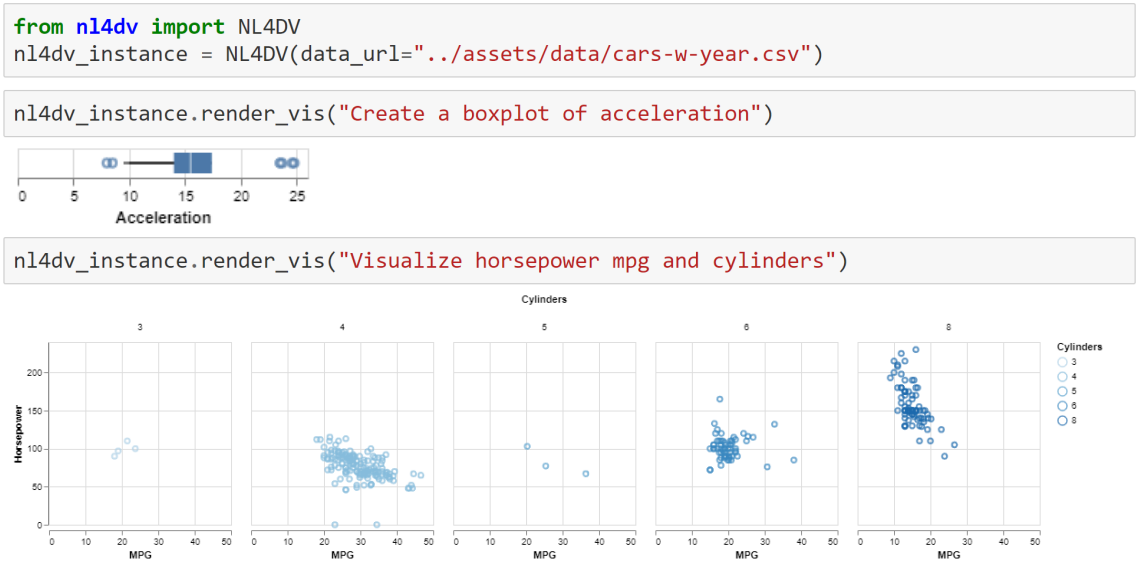



Figure 7.8: NL4DV being used to specify visualizations through NL in Python within a Jupyter Notebook.

sualization design or Python visualization packages (e.g., Matplotlib, Plotly). Figure 7.8 shows an instance of a Jupyter Notebook demonstrating the use of NL4DV to create visualizations for a cars dataset. For the first query "*Create a boxplot of acceleration*," detecting an explicit visualization request, NL4DV renders a *box plot* showing values for the attribute *Acceleration*. For the second query "*Visualize horsepower mpg and cylinders*", NL4DV implicitly selects a scatterplot as the appropriate visualization using the inferred attributes (# *Horsepower*, # *MPG*, ≡ *Cylinders*).

## 7.5 Limitations and Future Work

*Towards a Grammar of Visualization Tasks*

A fundamental difference between NLIs and other visualization systems is that NL allows people to more freely express their intended analytic tasks. However, while there exist formalized grammars for specifying visualizations [169, 210], tasks remain a more logical and subjective concept with implementations tailored to individual systems. Through NL4DV's `taskMap` (Figure 7.5c), we present one plausible formal representation of vi-

sual analytic tasks in terms of `attributes`, `operator`, and `values`. This is only a preliminary representation, however, and is designed with common low-level analytic tasks [3] in mind. To this end, analogous to the grammar of graphics, an interesting direction for research is to extend this initial grammar or formalize a new grammar of analytic tasks for visualizations. Such a grammar could promote consistent description and implementation of tasks across visualization systems, also enabling more structured development of task-based visualization recommendation systems (e.g., [132, 162]).

*Supporting Other Visualizations and Tasks*

In this initial version of NL4DV, we focus on supporting NL-based specification of basic visualizations such as bar charts, line charts, and scatterplots, among others in the context of tabular datasets. Going forward, we will extend the toolkit to support additional visualization types including maps and networks. Besides visualizations, we will also extend the set of analytic tasks inferred by NL4DV to support other tasks such as comparisons and finding outliers, enabling more robust querying and visualization specification. While NL4DV currently allows overriding its default attribute type interpretations, we are also looking into incorporating recent semantic data type detection models (e.g., [78, 227]) within NL4DV directly. Expanding beyond visualization specification, another area for development is adding support for other capabilities described in Chapter 3 such as question answering and formatting visualizations through NL.

With some custom logic as shown in the first two example applications, NL4DV's current output can be used to develop simple multimodal visualization interfaces. However, the supported range of multimodal interactions are minimal and primarily involve parallel or sequential integration [124] (as opposed to the synergistic interactions shown in systems like InChorus and DataBreeze). Thus, going forward it is important to extend NL4DV to support easier integration of data and events from different input modalities (e.g., through an added parameter that tracks the state of the active visualization and processes queries

based on that state).

*Evaluation via a Longitudinal User Study*

We have tested NL4DV by querying it against tabular datasets containing between 300-6000 rows and up to 15 attributes. We currently showcase NL4DV's capabilities through sample queries (additional queries are provided as supplementary material) and by illustrating applications developed using NL4DV. However, effectively assessing the toolkit's value deems a more longitudinal study incorporating feedback from both visualization and NLP developers. Such a study will help assess the practical usability of the toolkit (both from an API design and scalability standpoint), highlight potential issues, and understand the breadth of possible applications.

## 7.6 Conclusion

In this chapter, I described the NL4DV toolkit. NL4DV takes as input a dataset and a NL query and returns an inferred set of data attributes/values, analytic tasks, and visualization specifications relevant to the query. Through multiple examples, I illustrate how the toolkit can be used in different contexts including developing multimodal visualization interfaces and supporting NL-based visualization specification. As an open-source toolkit[2], NL4DV promotes and enables future research on NL-based visualization systems by reducing development viscosity and lowering skill barriers to accommodate designers/developers who lack experience in NLP tools and techniques **[RG4]**.

[2] https://nl4dv.github.io/nl4dv/

# CHAPTER 8
# REFLECTION AND FUTURE WORK

Since I embarked upon this research in 2016, there have been notable developments in the space of NL-based visualization tools including the emergence of multiple research prototypes [77, 89, 93, 100, 225] and commercial tools such as Tableau's Ask Data [173, 195] and ThoughtSpot [197]. These advancements highlight the growing popularity of NL interaction in visualization tools and reiterate the value of the results and insights from this dissertation. To provide an overview of the current state of research on NL-based visualization tools, I apply the query goal-oriented framework introduced in Chapter 3 and chronologically summarize relevant systems (including the ones I developed) in Table 8.1.

Table 8.1 highlights that while some of the recent systems (e.g., Evizeon [77], ORKO) have continued to focus on supporting visualization-specific interactions, there is an increased focused on the theme of generating or specifying visualizations through NL. Similarly, recent systems have begun to target individual tasks such as formatting visualizations (Vis-Annotator [100]) and low-level question answering (VisQA [93]) that previously received little to no consideration.

## 8.1 Reflection

In this section, I reflect upon my work over the past four years, discussing how my goals and perspective evolved over time, the challenges I faced during system evaluations, and the importance of complementing the expressiveness of multimodal interfaces with system intelligence.

Table 8.1: Updated version of Chapter 3, Table 3.1. A **dark pink** stroke indicates that the listed systems were published/developed after our initial survey in February 2017. **Highlighted system names** are developed as part of this dissertation's research.

| | | 2001 | 2011 | 2015 | 2016 | | 2017 | | 2018 | 2019 | 2020 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | InfoStill | Articulate | DataTone | Eviza | Articulate2 | Evizeon | Orko | Valetto | FlowSense | InChorus | VisQA | Vis-Annotator | DataBreeze | NL4DV |
| **Visualization-oriented** | Generating Visualizations | ☐ | ✎ | ■ | | ✎ | ■ | | ■ | ■ | ■ | | | ☐ | ■ |
| | Visualization Specific Interactions | ☐ | ☐ | ☐ | ■ | ✎ | ■ | ■ | | ☐ | ■ | | | ■ | ☐ |
| | Formatting Visualizations | | | | * | | | | | | | | ■ | ✎ | |
| **Data-oriented** | Low-level Analytic Operations/Questions | | ☐ | | ☐ | * | ☐ | | ☐ | | | ■ | | | |
| | High-level Questions | | | | * | | | | | | | | | | |
| **System control-oriented** | | ☐ | | | | ☐ | | ☐ | | ■ | | | | ✎ | |

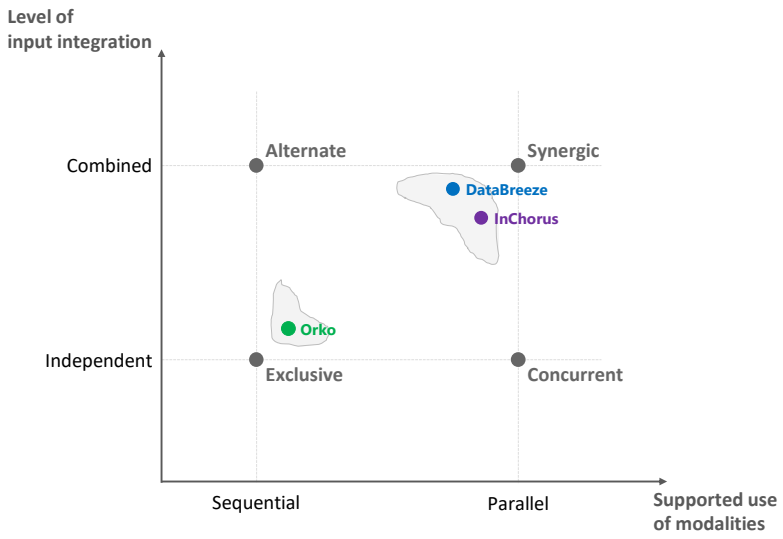



Figure 8.1: A design space of multimodal interactions as described by Gourdol et al. [65] along with an approximation of where the three multimodal visualization interfaces presented in this dissertation lie within that design space.

### 8.1.1 Towards Synergic Multimodal Visualization Interfaces

Following the approach of well-known multimodal interface frameworks (e.g., [65, 124]), one can model a design space of multimodal interactions as shown in Figure 8.1. Specifically, Figure 8.1 categorizes multimodal systems along two dimensions: i) their supported use of modalities and ii) the level of input integration that the system supports. The figure also shows where the three systems described in this dissertation lie within this design space. ORKO extensively supports *exclusive* interactions (modalities used independently) while minimally supporting *alternate* (modalities combined sequentially) and *concrurrent* (modalities used in parallel for different operations) interactions. On the other hand, INCHORUS and DATABREEZE support a higher degree of *synergic* interactions (modalities combined in parallel) while also supporting other styles of multimodal interactions when semantically appropriate.

When I began this research, I approached it from a comparative lens and sought to understand the trade-offs between DM-only and multimodal interfaces that supported both NL and DM. Looking back, this mindset is also reflected in ORKO's design (through the

focus on supporting equivalence between touch and speech) and evaluation (particularly the second study comparing unimodal and multimodal interaction). However, as I gathered more knowledge about why people prefer multimodal interaction (Chapter 4, Section 4.5.3) and experienced how synergic multimodal interfaces can enable new scenarios and benefits (Chapters 5 and 6), I realized that solely using a comparative lens was a confining design approach. While I highlight the value of synergic interactions in describing INCHORUS and DATABREEZE, I restate it here since this was one of the key takeaways for me as a designer of multimodal visualization systems:

> *As system designers, we should not overemphasize designs that support all operations with all modalities. Instead, we must strive to find the most suited operation-to-modality mappings and synergistically combine modalities when an operation logically affords it.*

There are some caveats to this recommendation, however. First, it is made under the assumption that the system is developed for a setting where all modalities are available at all times (e.g., all users of a system like INCHORUS need to have access to a pen in addition to the implicitly supported touch and speech input). Second, it assumes that all modalities can be operated equally by all users (i.e., it does not account for users with certain types of disabilities). The second point pertaining to people with disabilities, in particular, is a major factor not considered in the scope of my research but is a vital design consideration for multimodal visualization interfaces going forward [106, 117].

### 8.1.2 Evaluation Challenges for NL-based Visualization Tools

Due to the availability of NL as an input modality, I constantly encountered certain challenges when conducting system evaluations. One of these challenges was that of determining the *training procedure*. Consider the following anecdote from my own work. During pilot sessions for the first study with ORKO (Chapter 4, Section 4.4), I initially told participants what operations (e.g., finding paths, filtering) the tool supports and demonstrated

sample commands for the same (e.g., we used the command "*Find a path between Rooney and Ronaldo*" to find the shortest path between two nodes). Although I explicitly made participants aware that these were only sample utterances, during the task phase, participants continued using the same phrasing to find paths between nodes as they thought it was the only pattern understood by the system. We also considered not having any training but found this to be impractical for a research prototype since participants had limited knowledge about what the tool could be used for and its basic operation. To overcome this, we eventually followed a "minimal" training protocol that included giving participants an overview of the system interface and listing supported operations but not executing sample NL commands as part of the training.

Another challenge during system evaluations was *framing* the study tasks. For standard visualization tools, a common practice is to give participants a series of questions or operations (e.g., "Which state had the highest sales in 2019?", "Highlight countries with a population under 100M") that they need to answer/perform using the tool. With NL-based systems, however, participants may simply parrot back such questions/tasks as a command to the system. To address this, as described in Chapters 4-6, I used a combination of jeopardy-style facts [62], target replication tasks, and open-ended exploration scenarios as part of my user studies. Nonetheless, as I also discuss in a separate article [190], these strategies have their unique risks and benefits that must be considered carefully depending on the study goal. For instance, jeopardy-style facts engage participants and mimic realistic data analysis scenarios but are challenging to phrase and rely on familiarity with a dataset. On the other hand, target replication tasks have minimal risk of phrasing bias but may not mimic realistic data analysis scenarios. Similarly, open-ended tasks are easy to devise but make it challenging to get feedback on specific system functionality. To summarize:

> *Determining the appropriate training procedure and choosing the right task framing strategy are two key considerations when evaluating NL-based visualization interfaces.*

### 8.1.3 Complementing Input Expressiveness with System Intelligence

In this dissertation, I primarily focused on investigating how multimodal interfaces offer users a higher level of expressivity to specify their intents. However, human-computer interaction, by definition, is bi-directional and involves a dialogue between the users and systems [56, 136]. To this end, besides reacting to user input, all three systems I described earlier incorporate some added level of system intelligence to aid user interaction. For example, ORKO incorporates elements of proactive system behavior by rearranging the summary charts in real-time based on user interactions (Figure 4.2E). Another example of an assistive feature is the contextual suggestion of example commands to aid discoverability of speech in DATABREEZE (Chapter 6, Figures 6.7 and 6.8). Although these were relatively modest additions to the system functionality, participants perceived these as intelligent behavior and stated that the features helped them become familiar with the tool's language and even get answers to questions they were thinking of posing next. For instance, referring to the real-time reordering of the summary charts in ORKO, one participant from the second study (Chapter 4, Section 4.5) said "*I really liked the charts that came up on the right. They always seemed to be relevant to what I was thinking of at the time.*" hinting that the summary charts preempted his follow-up queries. Similarly, commenting on the contextual command suggestions in DATABREEZE, one of the study participants said "*The tooltips were really helpful in the initial practice phase and helped me incrementally understand what I could say.*" Based on such comments, an important aspect to consider when designing multimodal visualization interfaces is implementing proactive features to assist user interactions. More specifically:

> *Complementing multimodal input with proactive system behavior can increase awareness of system actions and promote contextual learning of possible interactions.*

## 8.2 Future Work

Below I highlight some themes for future work that can build upon the ideas presented in this dissertation.

### 8.2.1 Unifying Multimodal and Mixed-Initiative Interaction for Fluid and Guided Data Exploration

During open-ended tasks like data exploration, users are often unfamiliar with the dataset and questions they want to investigate. In such scenarios, systems can leverage their computational capabilities to mine both the underlying data and the user interactions to guide users in exploring their data through in-situ recommendations of "next steps." Although there is a long history of research on mixed-initiative recommendations within visualization tools (e.g., [64, 92, 212]), multimodal visualization systems that incorporate NL present unique opportunities to extend this line of work in a couple of ways.

First, many existing systems track user interactions during data exploration to model user interest. However, current DM-based tools primarily support interactions only with data attributes and values. Alternatively, with NL, people can issue queries referring to data attributes, values, tasks, and/or visualizations, all as part of a single query, or combine NL with DM interactions (e.g., selections) on the active view. This added expressivity offered by NL can allow systems to better capture user interest and generate recommendations not only pertaining to attributes and visualizations, but also analytic tasks. Secondly, current tools largely leverage visualization thumbnails and faceted views to present recommendations (e.g., [92, 213]). Here, thinking about how NL can be used as part of system output (as opposed to just user input) opens research dimensions minimally explored by prior work (e.g., [45, 112]). For instance, given a visualization, systems can compute statistical measures from the underlying data and present salient facts (as textual statements) that can be observed using the visualization. These "data facts" can, in turn, be used to

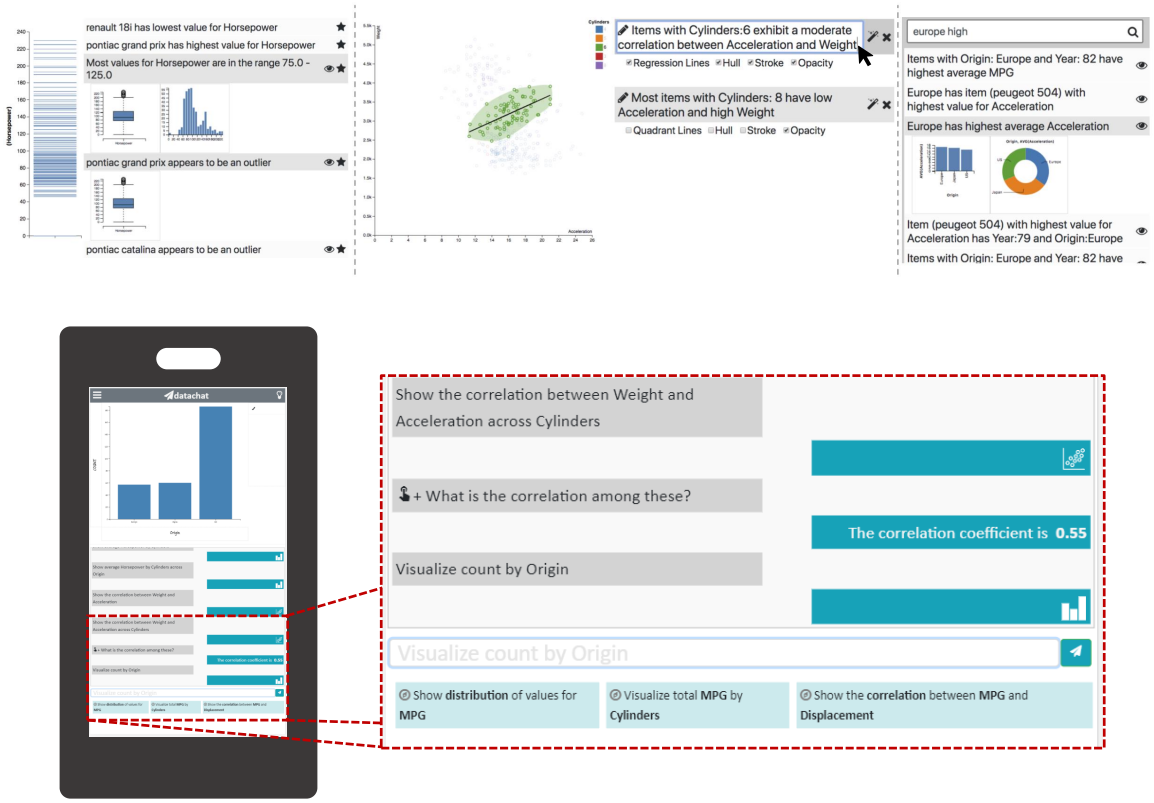



Figure 8.2: Prototypes illustrating how visualization systems can incorporate elements both of multimodal and mixed-initiative interfaces. (Top) Interactive "data facts" in VODER [185] surface visualization and annotation recommendations, helping users explore data and communicate their findings. (Bottom) A conversational data visualization interface, DATACHAT, demonstrating how NL query recommendations can be leveraged to aid data exploration on mobile devices.

guide data exploration through recommendations of related facts and visualizations. Beyond supporting new classes of recommendations, NL-based recommendations can also guide data exploration in alternative settings like mobile devices where current thumbnail-based presentation strategies cannot be used due to practical constraints like limited screen space.

Figure 8.2 shows early stage prototypes I developed to illustrate the aforementioned ideas. The discussions in earlier chapters coupled with the feedback on these early stage prototypes raise some compelling research questions at the intersection of visual analytics and user interfaces including: how can systems effectively capture user interest from

multimodal interactions? Are there particular modalities that are closely tied to a specific type of information (e.g., touch to infer attribute interest, speech to infer tasks)? How can interest captured through one modality be transmitted to another? Can systems use NL or even multiple modes of output to present recommendations? Are these recommendations more interpretable than current recommendation presentation strategies?

### 8.2.2 Multimodal Interfaces for Data Preparation

Following Card et al.'s visualization reference model [27], visualizations are essentially adjustable mappings from data to a visual form (Figure 8.3). Both the work presented in this dissertation and research on post-WIMP visualization interfaces in general have predominantly focused on the second half (i.e., the 'visual form' in Figure 8.3), exploring innovative ways to specify visual mappings and perform view transformations. Research has shown, however, that in practice, a major challenge that data scientists or data enthusiasts [70] face is getting the data into a format that can be visualized [86, 88]. Particularly as portable devices such as tablets and phones become popular platforms for visualizations, we need to design tools that support a more complete analysis workflow—allowing users to adjust their data to fit questions they seek to answer through visualizations.

The expressivity of NL- and DM-based multimodal interactions can offer a unique mix

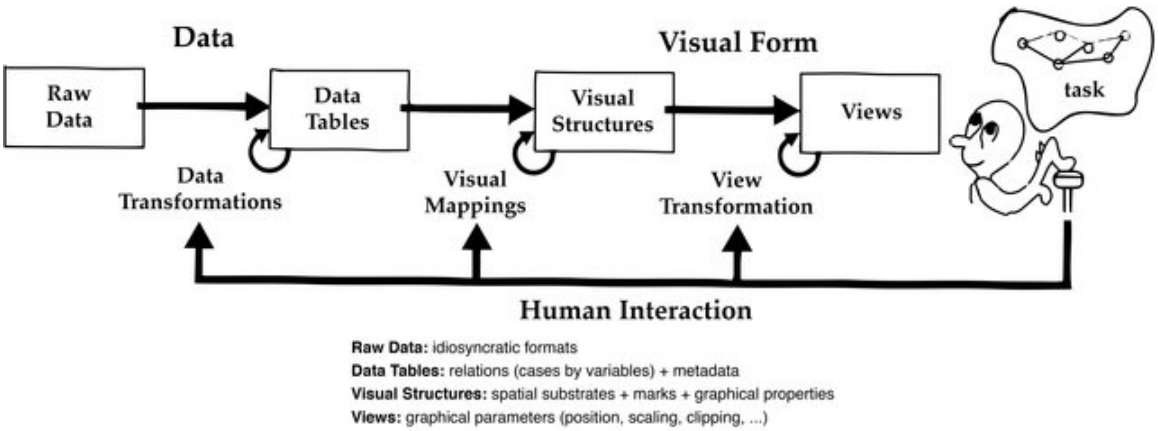



Figure 8.3: Reference model for visualization (from Card et al. [27])

of capabilities to assist data preparation and also enable data preparation on contemporary devices. For instance, instead of interacting with individual data rows or instances as in current DM-only tools (e.g., [87, 187]), with NL, people can pose a higher-level query over a raw dataset specifying their intended goal. Interpreting this query, systems can then present users with transformed data models that could address the input query without requiring users to manually perform the steps to create those data models. Consider a movies dataset like the one shown in Chapter 7, Figure 7.1. When working with this dataset, one may say "*I want to explore the relationship between content rating and genres*." Identifying the attributes from the query, the system can then model multiple networks where the values under 'Content Rating' and 'Genre' are treated as nodes, and nodes are linked based on the intersection of values for one or more other attributes (e.g., a link will be added between two nodes for each movie that has a specific [content rating, genre] combination). Alternatively, combining modalities more synergistically, one could edit one cell or row in a table through DM (e.g., deleting a row with the eraser of a pen) and then selectively propagate those actions to other portions of the table through NL (e.g., say "*Now repeat this for all movies that grossed under 100M.*"). Given the expressivity of multimodal interfaces and the broad user base they can support (e.g., analysts, data enthusiasts, students), exploring multimodal interfaces for data preparation (particularly in post-WIMP settings) is a promising direction for future work.

### 8.2.3 Expressive Tools for Data-Driven Presentation

*Exploration* (where users do not know what is in the data) and *presentation* (where some result has to be communicated to others) are often considered as the two most prominent use cases for information visualization [201]. In my work, I have predominantly focused on data exploration as the target use case. However, given the strong lineage of multimodal interfaces for graphics applications (e.g., [65, 147, 192]) and the creative potential exhibited by recent pen- and touch-based visualization authoring tools [94, 108, 217], de-

veloping more expressive pen-, touch-, and speech-based visualization authoring tools is an exciting topic to explore going forward. Particularly, since visualization authoring tools often require users to go through a plethora of menus to customize the view, it will be interesting to investigate the role that speech plays in this setting, addressing questions such as: does speech improve the user workflow in visualization authoring tools? If so, how? Are there specific authoring operations that are better aligned with certain modalities? Does the added expressivity offered by speech affect user creativity? Furthermore, going beyond "creative" visualization authoring tools that typically focus on helping designers create infographics, a related direction for research is to examine multimodal dashboard design interfaces. Dashboards remain one of the most prevalent form of information visualizations in practice and there is a growing suite of commercial dashboard design tools [168]. Along these lines, future work should explore how current NL interpretation techniques that focus on creating one visualization in response to input queries can be extended to support higher-level user queries (e.g., "*Show me the company performance for last year*") as input and create an entire dashboard template in response. Using these templates in concert with multimodal DM and NL interactions, designers can customize their dashboards and share them with other stakeholders. From an end-user standpoint (as opposed to a dashboard designer), interactions shown in InChorus can then be used to consume these dashboards and explore the data in more engaging ways than currently possible.

### 8.2.4 From Multi-modal to Multi-user Interfaces

Large touchscreen displays and AR/VR devices continue to become more affordable and are becoming ubiquitous in modern workspaces. Due to their form factor, such settings afford a unique opportunity to enable co-located collaboration between multiple users. Collaborative data analysis can combine the analytic power of multiple individuals with the possibility of leveraging varying levels of expertise. This combination can, in turn, lead to increased quality of data-driven decisions and insights.

Multimodal input involving NL can further complement collaborative data exploration by allowing users to more naturally express their intended actions individually or as a team. For example, when working with large displays, users may often sit at a distance from the display to discuss details about the data displayed on the screen. In such settings, to make changes to the view based on their discussion, users can issue spoken commands to adjust the visualization without having to disrupt the conversation and physically move to the screen. Alternatively, users can work on different areas of a shared display (or separate displays) individually using pen/touch, step away from the display and discuss their findings, and make global changes to the shared view using speech. While the visualization research community has shown significant interest in exploring such collaborative scenarios over the past decade (e.g., [10, 11, 82, 97, 102]), these efforts do not consider the role of NL interaction (between users—systems or users—users) in such settings. With the advancements in voice recognition technology and signal separation techniques, we finally have the opportunity to revisit the promising collaborative interaction scenarios envisioned by early work on multimodal interfaces (e.g., [37, 118, 128]) and operationalize them within the context of data analysis and visualization.

From a research perspective exploring such collaborative scenarios presents many intriguing questions at the interaction of user interfaces, visualization design, and sensemaking including: which modalities can be coherently supported during collaborative analytics? How can input from different modalities be integrated across users? What strategies do people adopt when interaction multimodally? Do they follow a turn-taking approach? Or interact simultaneously? How can we design representations that depict the interaction provenance across users to enable collective awareness of the data analysis process? How does having multiple modes of input impact the sensemaking process at an individual-level and as a group?

# CHAPTER 9
# CONCLUSION

The goal of this dissertation is to investigate and promote fluid and expressive human-data interaction experiences through the design of novel multimodal visualization interfaces that combine natural language (NL) and direct manipulation (DM) input.

To give an overview of the role of NL in the context of visualization tools, in *Chapter 3*, I describe a literature review and analysis of NL utterance types, synthesizing the results into a goal-oriented framework to characterize NL interfaces for visualization. Through the literature review, I also identify post-WIMP multimodal interfaces as a promising research theme to investigate and make it the focus of the three subsequent chapters.

In *Chapters 4* and *5*, I explore the design of multimodal interfaces that combine speech with pen and/or touch to support fundamental visual analysis operations across a breadth of chart types (e.g., bar charts, line charts, node-link diagrams). Extrapolating themes from my design process, I provide principles and concepts that can help design and compare multimodal interactions in future visualization systems. Through evaluations of two prototype systems, ORKO and INCHORUS, I show that multimodal interfaces promote fluid and expressive interactions that accommodate varying user interaction patterns and preferences during visual analysis.

Moving past typical representations and analysis scenarios, in *Chapter 6*, I describe an innovative approach that interweaves multimodal interaction with flexible unit visualizations. Operationalizing this approach through a prototype system, DATABREEZE, I demonstrate how it enables a free-form data exploration experience by allowing people to specify both systematically-bound views (e.g., scatter plots, unit column charts) and customized views that best reflect their mental model of a data space.

Building upon my experience of implementing NL-based multimodal systems and knowl-

edge gained from related work on NLIs for data visualization, I develop an interface-agnostic toolkit, NL4DV, that helps developers prototype NL-based visualization interfaces. In *Chapter 7*, I detail NL4DV's design goals and describe how the toolkit formulates an analytic specification (composing of data attributes/values, analytic tasks, and relevant visualizations) from a given NL query and illustrate the practical utility of this specification through a series of example applications.

Finally, in *Chapter 8*, I reflect upon my research experience and discuss three key themes: 1) the importance of designing synergic multimodal interactions, 2) evaluation challenges for NL-based visualization systems, and 3) complementing multimodal input with proactive system behavior to assist user interaction. I also highlight open areas for future research such as developing multimodal interfaces for data preparation, unifying multimodal and mixed-initiative interaction to support fluid and guided data exploration, and designing expressive tools for visualization authoring, among others.

Overall, through the work described in this dissertation, I make the following contributions to the visualization research community:

- A framework of NL utterances based on user goals in the context of visualization tools.
- The design and implementation of two prototype multimodal visualization interfaces: ORKO and INCHORUS, that combine speech with pen and/or touch to support fluid and expressive interactions for visual analysis with canonical chart types (e.g., line charts, bar charts, node-link diagrams).
- Characterizations of multimodal user input and interaction patterns within visualization tools based on evaluations of ORKO and INCHORUS.
- Design guidelines and concepts that can be used to develop and compare multimodal interactions with visualization tools supporting NL and DM.

- An approach to enable free-form data exploration by interweaving multimodal interaction with flexible unit visualizations, along with an operationalization of this approach through a prototype system, DATABREEZE.
- NL4DV, a toolkit that helps developers prototype NL-based visualization systems.

As the amount of data available to us grows at a staggering rate, now more than ever, there is a tremendous need for effective visualization tools that help convert data into information. By leveraging advancements in hardware and input recognition technology, we can develop interfaces that lower the barriers to entry for using visualization tools and empower data analysis through naturalistic interactions. Ultimately, this dissertation hopes to push us towards exploring a new generation of information interfaces that augment our human perceptual, cognitive, and manipulative abilities during human-data interaction.

## REFERENCES


[1] G. D. Abowd and A. J. Dix, "Giving undo attention," *Interacting with Computers*, vol. 4, no. 3, pp. 317–342, 1992.

[2] M. Agarwal, A. Srinivasan, and J. Stasko, "VisWall: Visual data exploration using direct combination on large touch displays," in *2019 IEEE Visualization Conference (VIS)*, IEEE, 2019, pp. 26–30.

[3] R. Amar, J. Eagan, and J. Stasko, "Low-level components of analytic activity in information visualization," in *Proceedings of IEEE Symposium on Information Visualization*, 2005, pp. 111–117.

[4] *Amazon Alexa*, https://en.wikipedia.org/wiki/Amazon_Alexa, 2020.

[5] C. Andrews, A. Endert, and C. North, "Space to think: Large high-resolution displays for sensemaking," in *Proceedings of the Conference on human factors in computing systems*, ACM, 2010, pp. 55–64.

[6] C. Andrews and C. North, "The impact of physical navigation on spatial organization for sensemaking," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2207–2216, 2013.

[7] N. Andrienko and G. Andrienko, *Exploratory analysis of spatial and temporal data: a systematic approach*. Springer Science & Business Media, 2006.

[8] I. Androutsopoulos, G. D. Ritchie, and P. Thanisch, "Natural language interfaces to databases–an introduction," *Natural language engineering*, vol. 1, no. 01, pp. 29–81, 1995.

[9] J. Aurisano, A. Kumar, A. Gonzalez, J. Leigh, B. DiEugenio, and A. Johnson, "Articulate2: Toward a conversational interface for visual data exploration," in *IEEE Visualization*, 2016.

[10] S. K. Badam and N. Elmqvist, "Polychrome: A cross-device framework for collaborative web visualization," in *Proceedings of the International Conference on Interactive Tabletops and Surfaces*, ACM, 2014, pp. 109–118.

[11] S. K. Badam, E. Fisher, and N. Elmqvist, "Munin: A peer-to-peer middleware for ubiquitous analytics and visualization spaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 2, pp. 215–228, 2015.

[12] S. K. Badam, A. Srinivasan, N. Elmqvist, and J. Stasko, "Affordances of input modalities for visual data exploration in immersive environments," in *2nd Workshop on Immersive Analytics*, 2017.

[13] R. C. Basole, A. Srinivasan, H. Park, and S. Patel, "Ecoxight: Discovery, exploration, and analysis of business ecosystems using interactive visualization," *ACM Transactions on Management Information Systems (TMIS)*, vol. 9, no. 2, pp. 1–26, 2018.

[14] M. Bastian, S. Heymann, M. Jacomy, *et al.*, "Gephi: An open source software for exploring and manipulating networks.," *ICWSM*, vol. 8, pp. 361–362, 2009.

[15] D. Baur, B. Lee, and S. Carpendale, "Touchwave: Kinetic multi-touch manipulation for hierarchical stacked graphs," in *Proceedings of the International Conference on Interactive Tabletops and Surfaces*, ACM, 2012, pp. 255–264.

[16] M. Beaudouin-Lafon, "Instrumental interaction: An interaction model for designing post-wimp user interfaces," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2000, pp. 446–453.

[17] F. Beck, M. Burch, S. Diehl, and D. Weiskopf, "A taxonomy and survey of dynamic graph visualization," in *Computer Graphics Forum*, Wiley Online Library, 2016.

[18] J. Berant, A. Chou, R. Frostig, and P. Liang, "Semantic parsing on freebase from question-answer pairs," in *Proceedings of the conference on empirical methods in natural language processing*, 2013, pp. 1533–1544.

[19] R. A. Bolt, "Put-that-there: Voice and gesture at the graphics interface," in *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, Seattle, Washington, USA, 1980, pp. 262–270.

[20] M. Bostock, V. Ogievetsky, and J. Heer, "$D^3$ data-driven documents," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2301–2309, 2011.

[21] M. Brehmer and T. Munzner, "A multi-level typology of abstract visualization tasks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2376–2385, 2013.

[22] S. E. Brennan and E. A. Hulteen, "Interaction and feedback in a spoken language system: A theoretical framework," *Knowledge-based systems*, vol. 8, no. 2-3, pp. 143–151, 1995.

[23] J. Browne, B. Lee, S. Carpendale, N. Riche, and T. Sherwood, "Data analysis on interactive whiteboards through sketch-based interaction," in *Proceedings of the*

*International Conference on Interactive Tabletops and Surfaces*, 2011, pp. 154–157.

[24] T. Buering, J. Gerken, and H. Reiterer, "User interaction with scatterplots on small screens-a comparative evaluation of geometric-semantic zoom and fisheye distortion," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 829–836, 2006.

[25] A. Buja, D. Cook, and D. F. Swayne, "Interactive high-dimensional data visualization," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 78–99, 1996.

[26] N. Bush, R. Wallace, T. Ringate, A. Taylor, and J. Baer, "Artificial intelligence markup language (aiml) version 1.0. 1," *ALICE AI Foundation Working Draft*, 2001.

[27] S. K. Card, J. D. Mackinlay, and B. Shneiderman, *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.

[28] S. M. Casner, "Task-analytic approach to the automated design of graphic presentations," *ACM Transactions on Graphics (ToG)*, vol. 10, no. 2, pp. 111–151, 1991.

[29] A. Cheyer and L. Julia, "Multimodal maps: An agent-based approach," in *International Conference on Cooperative Multimodal Communication*, Springer, 1995, pp. 111–121.

[30] P.-Y. P. Chi, D. Vogel, M. Dontcheva, W. Li, and B. Hartmann, "Authoring illustrations of human movements by iterative physical demonstration," in *Proceedings of the Annual Symposium on User Interface Software and Technology*, ACM, 2016, pp. 809–820.

[31] M. C. Chuah and S. F. Roth, "On the semantics of interactive visualizations," in *Proceedings of IEEE Symposium on Information Visualization*, IEEE, 1996, pp. 29–36.

[32] J. H. Claessen and J. J. Van Wijk, "Flexible linked axes for multivariate data visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2310–2316, 2011.

[33] L. Clark, P. Doyle, D. Garaialde, E. Gilmartin, S. Schlögl, J. Edlund, M. Aylett, J. Cabral, C. Munteanu, J. Edwards, *et al.*, "The state of speech in hci: Trends, themes and challenges," *Interacting with Computers*, vol. 31, no. 4, pp. 349–371, 2019.

[34] C. Codella, R. Jalili, L. Koved, J. B. Lewis, D. T. Ling, J. S. Lipscomb, D. A. Rabenhorst, C. P. Wang, A. Norton, P. Sweeney, *et al.*, "Interactive simulation in a

multi-person virtual world," in *Proceedings of the Conference on Human factors in computing systems*, ACM, 1992, pp. 329–334.

[35] P. Cohen, D. McGee, and J. Clow, "The efficiency of multimodal interaction for a map-based task," in *Extended Abstracts of Conference on Human Factors in Computing Systems*, ACM, 2000, pp. 26–27.

[36] P. R. Cohen, M. Dalrymple, D. B. Moran, F. Pereira, and J. W. Sullivan, "Synergistic use of direct manipulation and natural language," in *ACM SIGCHI Bulletin*, vol. 20, 1989, pp. 227–233.

[37] P. R. Cohen, M. Johnston, D. McGee, S. Oviatt, J. Pittman, I. Smith, L. Chen, and J. Clow, "Quickset: Multimodal interaction for distributed applications," in *Proceedings of the International Conference on Multimedia*, ACM, 1997, pp. 31–40.

[38] P. R. Cohen, M. Johnston, D. McGee, S. Oviatt, J. Pittman, I. Smith, L. Chen, and J. Clow, "Quickset: Multimodal interaction for simulation set-up and control," in *Proceedings of the Conference on Applied natural language processing*, ACL, 1997, pp. 20–24.

[39] P. R. Cohen, M. Johnston, D. McGee, S. L. Oviatt, J. Clow, and I. A. Smith, "The efficiency of multimodal interaction: A case study.," in *ICSLP*, 1998.

[40] M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott, "Imaxes: Immersive axes as embodied affordances for interactive multivariate data visualisation," in *Proceedings of the Annual Symposium on User Interface Software and Technology*, ACM, 2017, pp. 71–83.

[41] M. Cordeil, T. Dwyer, K. Klein, B. Laha, K. Marriott, and B. H. Thomas, "Immersive collaborative analysis of network connectivity: Cave-style or head-mounted display?" *IEEE transactions on visualization and computer graphics*, vol. 23, no. 1, pp. 441–450, 2017.

[42] K. Cox, R. E. Grinter, S. L. Hibino, L. J. Jagadeesan, and D. Mantilla, "A multimodal natural language interface to an information visualization environment," *International Journal of Speech Technology*, vol. 4, no. 3-4, pp. 297–314, 2001.

[43] A. Cremers and S. Ginsburg, "Context-free grammar forms," *Journal of Computer and System Sciences*, vol. 11, no. 1, pp. 86–117, 1975.

[44] W. Cui, X. Zhang, Y. Wang, H. Huang, B. Chen, L. Fang, H. Zhang, J.-G. Lou, and D. Zhang, "Text-to-viz: Automatic generation of infographics from proportion-related natural language statements," *IEEE transactions on visualization and computer graphics*, vol. 26, no. 1, pp. 906–916, 2019.

[45] Ç. Demiralp, P. J. Haas, S. Parthasarathy, and T. Pedapati, "Foresight: Rapid data exploration through guideposts," *arXiv preprint arXiv:1709.10513*, 2017.

[46] E. Dimara and C. Perin, "What is interaction for data visualization?" *IEEE transactions on visualization and computer graphics*, vol. 26, no. 1, pp. 119–129, 2019.

[47] A. Dix and G. Ellis, "Starting simple: Adding value to static visualisation through simple interaction," in *Proceedings of the working conference on Advanced visual interfaces*, ACM, 1998, pp. 124–134.

[48] A. Drogemuller, A. Cunningham, J. Walsh, M. Cordeil, W. Ross, and B. Thomas, "Evaluating navigation techniques for 3d graph visualizations in virtual reality," in *Proceedings of the International Symposium on Big Data Visual and Immersive Analytics (BDVA)*, IEEE, 2018, pp. 1–10.

[49] S. Drucker and R. Fernandez, "A unifying framework for animated and interactive unit visualizations," *Microsoft Research*, 2015.

[50] S. M. Drucker, D. Fisher, R. Sadana, J. Herron, *et al.*, "Touchviz: A case study comparing two interfaces for data analytics on tablets," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2013, pp. 2301–2310.

[51] B. Dumas, D. Lalanne, and S. Oviatt, "Multimodal interfaces: A survey of principles, models and frameworks," in *Human machine interaction*, Springer, 2009, pp. 3–26.

[52] N. Elmqvist, A. V. Moere, H.-C. Jetter, D. Cernea, H. Reiterer, and T. Jankun-Kelly, "Fluid interaction for information visualization," *Information Visualization*, p. 1 473 871 611 413 180, 2011.

[53] A. Endert, L. Bradel, and C. North, "Beyond control panels: Direct manipulation for visual analytics," *IEEE computer graphics and applications*, vol. 33, no. 4, pp. 6–13, 2013.

[54] A. Endert, P. Fiaux, and C. North, "Semantic interaction for visual text analytics," in *Proceedings of the Conference on Human factors in computing systems*, ACM, 2012, pp. 473–482.

[55] E. Fast, B. Chen, J. Mendelsohn, J. Bassen, and M. S. Bernstein, "Iris: A conversational agent for complex tasks," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2018, p. 473.

[56] J. D. Foley, F. D. Van, A. Van Dam, S. K. Feiner, J. F. Hughes, E. Angel, and J. Hughes, *Computer graphics: principles and practice*. Addison-Wesley Professional, 1996, vol. 12110.

[57] M. Frisch, J. Heydekorn, and R. Dachselt, "Investigating multi-touch and pen gestures for diagram editing on interactive surfaces," in *Proceedings of the International Conference on Interactive Tabletops and Surfaces*, ACM, 2009, pp. 149–156.

[58] D. M. Frohlich, "The history and future of direct manipulation," *Behaviour & Information Technology*, vol. 12, no. 6, pp. 315–329, 1993.

[59] D. M. Frohlich, "Direct manipulation and other lessons," *Handbook of human-computer interaction*, pp. 463–488, 1997.

[60] M. Fukumoto, Y. Suenaga, and K. Mase, "'"finger-pointer": Pointing interface by image processing," *Computers & graphics*, vol. 18, no. 5, pp. 633–642, 1994.

[61] A. Furqan, C. Myers, and J. Zhu, "Learnability through adaptive discovery tools in voice user interfaces," in *Extended Abstracts of the Conference on Human Factors in Computing Systems*, ACM, 2017, pp. 1617–1623.

[62] T. Gao, M. Dontcheva, E. Adar, Z. Liu, and K. G. Karahalios, "Datatone: Managing ambiguity in natural language interfaces for data visualization," in *Proceedings of the Annual Symposium on User Interface Software & Technology*, ACM, 2015, pp. 489–500.

[63] *Google Assistant*, https://en.wikipedia.org/wiki/Google_Assistant, 2020.

[64] D. Gotz and Z. Wen, "Behavior-driven visualization recommendation," in *Proceedings of International Conference on Intelligent User Interfaces*, ACM, 2009, pp. 315–324.

[65] A. P. Gourdol, L. Nigay, D. Salber, J. Coutaz, *et al.*, "Two case studies of software architecture for multimodal interactive systems: Voicepaint and a voice-enabled graphical notebook.," *Engineering for HCI*, vol. 92, pp. 271–284, 1992.

[66] B. J. Grosz, "Focusing and description in natural language dialogues," DTIC Document, Tech. Rep., 1979.

[67] B. J. Grosz and C. L. Sidner, "Attention, intentions, and the structure of discourse," *Computational linguistics*, vol. 12, no. 3, pp. 175–204, 1986.

[68] B. J. Grosz, S. Weinstein, and A. K. Joshi, "Centering: A framework for modeling the local coherence of discourse," *Computational linguistics*, vol. 21, no. 2, pp. 203–225, 1995.

[69] C. A. Halverson, D. B. Horn, C.-M. Karat, and J. Karat, "The beauty of errors: Patterns of error correction in desktop speech systems," in *Proceedings of INTERACT*, 1999, pp. 133–140.

[70] P. Hanrahan, "Analytic database technologies for a new kind of user: The data enthusiast," in *Proceedings of the International Conference on Management of Data*, ACM, 2012, pp. 577–578.

[71] A. G. Hauptmann, "Speech and gestures for graphic image manipulation," *ACM SIGCHI Bulletin*, vol. 20, no. SI, pp. 241–245, 1989.

[72] J. Heer, M. Agrawala, and W. Willett, "Generalized selection via interactive query relaxation," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2008, pp. 959–968.

[73] J. Heer and B. Shneiderman, "Interactive dynamics for visual analysis," *Communications of the ACM*, vol. 55, no. 4, pp. 45–54, 2012.

[74] I. Herman, G. Melançon, and M. S. Marshall, "Graph visualization and navigation in information visualization: A survey," *IEEE Transactions on visualization and computer graphics*, vol. 6, no. 1, pp. 24–43, 2000.

[75] K. Hinckley and D. Wigdor, "Input technologies and techniques," in *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*, ACM, 2002, pp. 151–168.

[76] K. Hinckley, K. Yatani, M. Pahud, N. Coddington, J. Rodenhouse, A. Wilson, H. Benko, and B. Buxton, "Pen+ touch= new tools," in *Proceedings of the Annual Symposium on User Interface Software and Technology*, ACM, 2010, pp. 27–36.

[77] E. Hoque, V. Setlur, M. Tory, and I. Dykeman, "Applying pragmatics principles for interaction with visual analytics," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 309–318, 2018.

[78] M. Hulsebos, K. Hu, M. Bakker, E. Zgraggen, A. Satyanarayan, T. Kraska, Ç. Demiralp, and C. Hidalgo, "Sherlock: A deep learning approach to semantic data type detection," in *Proceedings of SIGKDD*, ACM, 2019.

[79] C. Hurter, N. H. Riche, S. M. Drucker, M. Cordeil, R. Alligier, and R. Vuillemot, "Fiberclay: Sculpting three dimensional trajectories to reveal structural insights," *IEEE transactions on visualization and computer graphics*, 2018.

[80] E. L. Hutchins, J. D. Hollan, and D. A. Norman, "Direct manipulation interfaces," *Human-computer interaction*, vol. 1, no. 4, pp. 311–338, 1985.

[81] *IBM watson analytics*, http://www.ibm.com/analytics/watson-analytics/, 2018.

[82] P. Isenberg and D. Fisher, "Collaborative brushing and linking for co-located visual analytics of document collections," in *Computer Graphics Forum*, Wiley Online Library, vol. 28, 2009, pp. 1031–1038.

[83] J. Jo, S. L'Yi, B. Lee, and J. Seo, "TouchPivot: Blending wimp & post-wimp interfaces for data exploration on tablet devices," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2017, pp. 2660–2671.

[84] J. Jo, B. Lee, and J. Seo, "Wordleplus: Expanding wordle's use through natural interaction and animation," *IEEE computer graphics and applications*, vol. 35, no. 6, pp. 20–28, 2015.

[85] R. J. L. John, N. Potti, and J. M. Patel, "Ava: From data to insights through conversations.," in *CIDR*, 2017.

[86] S. Kandel, J. Heer, C. Plaisant, J. Kennedy, F. Van Ham, N. H. Riche, C. Weaver, B. Lee, D. Brodbeck, and P. Buono, "Research directions in data wrangling: Visualizations and transformations for usable and credible data," *Information Visualization*, vol. 10, no. 4, pp. 271–288, 2011.

[87] S. Kandel, A. Paepcke, J. Hellerstein, and J. Heer, "Wrangler: Interactive visual specification of data transformation scripts," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2011, pp. 3363–3372.

[88] S. Kandel, A. Paepcke, J. M. Hellerstein, and J. Heer, "Enterprise data analysis and visualization: An interview study," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2917–2926, 2012.

[89] J.-F. Kassel and M. Rohs, "Valletto: A multimodal interface for ubiquitous visual analytics," in *Extended Abstracts of the Conference on Human Factors in Computing Systems*, ACM, 2018.

[90] D. A. Keim, "Information visualization and visual data mining," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 1–8, 2002.

[91] A. Kendon, "Gesticulation and speech: Two aspects of the," *The relationship of verbal and nonverbal communication*, no. 25, p. 207, 1980.

[92] A. Key, B. Howe, D. Perry, and C. Aragon, "Vizdeck: Self-organizing dashboards for visual analytics," in *Proceedings of the International Conference on Management of Data*, ACM, 2012, pp. 681–684.

[93] D. H. Kim, E. Hoque, and M. Agrawala, "Answering questions about charts and generating visual explanations," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2020, pp. 1–13.

[94] N. W. Kim, N. Henry Riche, B. Bach, G. Xu, M. Brehmer, K. Hinckley, M. Pahud, H. Xia, M. J. McGuffin, and H. Pfister, "Datatoon: Drawing dynamic network comics with pen+touch interaction," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2019, pp. 1–12.

[95] Y.-S. Kim, N. Henry Riche, B. Lee, M. Brehmer, M. Pahud, K. Hinckley, and J. Hullman, "Inking your insights: Investigating digital externalization behaviors during data analysis," in *Proceedings of the International Conference on Interactive Surfaces and Spaces*, ACM, 2019, pp. 255–267.

[96] Y. Kim and J. Heer, "Assessing effects of task and data distribution on the effectiveness of visual encodings," in *Computer Graphics Forum*, Wiley Online Library, vol. 37, 2018, pp. 157–167.

[97] U. Kister, K. Klamka, C. Tominski, and R. Dachselt, "Grasp: Combining spatially-aware mobile devices and a display wall for graph visualization and interaction," in *Computer Graphics Forum*, Wiley Online Library, vol. 36, 2017, pp. 503–514.

[98] D. B. Koons, C. J. Sparrell, and K. R. Thorisson, "Integrating simultaneous input from speech, gaze, and hand gestures," *MIT Press: Menlo Park, CA*, pp. 257–276, 1993.

[99] A. Kumar, J. Aurisano, B. Di Eugenio, A. Johnson, A. Gonzalez, and J. Leigh, "Towards a dialogue system that supports rich visualizations of data," in *Proceedings of the Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2016, p. 304.

[100] C. Lai, Z. Lin, R. Jiang, Y. Han, C. Liu, and X. Yuan, "Automatic annotation synchronizing with textual description for visualization," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2020, pp. 1–13.

[101] R. Langner, T. Horak, and R. Dachselt, "Vistiles: Coordinating and combining co-located mobile devices for visual data exploration," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 626–636, 2018.

[102] R. Langner, U. Kister, and R. Dachselt, "Multiple coordinated views at large displays for multiple users: Empirical findings on user behavior, movements, and distances," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 608–618, 2019.

[103] G. P. Laput, M. Dontcheva, G. Wilensky, W. Chang, A. Agarwala, J. Linder, and E. Adar, "Pixeltone: A multimodal interface for image editing," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2013, pp. 2185–2194.

[104] M. Le Goc, C. Perin, S. Follmer, J.-D. Fekete, and P. Dragicevic, "Dynamic composite data physicalization using wheeled micro-robots," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 737–747, 2019.

[105] B. Lee, M. Brehmer, P. Isenberg, E. K. Choe, R. Langner, and R. Dachselt, "Data visualization on mobile devices," in *Extended Abstracts of the Conference on Human Factors in Computing Systems*, ACM, 2018, W07.

[106] B. Lee, E. K. Choe, P. Isenberg, K. Marriott, and J. Stasko, "Reaching broader audiences with data visualization," *IEEE Computer Graphics and Applications*, vol. 40, no. 2, pp. 82–90, 2020.

[107] B. Lee, P. Isenberg, N. H. Riche, and S. Carpendale, "Beyond mouse and keyboard: Expanding design considerations for information visualization interactions," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2689–2698, 2012.

[108] B. Lee, R. H. Kazi, and G. Smith, "Sketchstory: Telling more engaging stories with data through freeform sketching," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2416–2425, 2013.

[109] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry, "Task taxonomy for graph visualization," in *Proceedings of the 2006 AVI workshop on Beyond Time and Errors: Novel Evaluation Methods for Information Visualization*, ACM, 2006, pp. 1–5.

[110] B. Lee, G. Smith, N. H. Riche, A. Karlson, and S. Carpendale, "Sketchinsight: Natural data exploration on interactive whiteboards leveraging pen and touch interaction," in *Proceedings of IEEE PacificVis '15*, 2015, pp. 199–206.

[111] Y. Li, K. Hinckley, K. Hinckley, Z. Guan, and J. A. Landay, "Experimental analysis of mode switching techniques in pen-based user interfaces," in *Proceedings of the Conference on Human factors in computing systems*, ACM, 2005, pp. 461–470.

[112] R. d. A. Lima and S. D. J. Barbosa, "Vismaker: A question-oriented visualization recommender system for data exploration," *arXiv preprint arXiv:2002.06125*, 2020.

[113] H. Lin, D. Moritz, and J. Heer, "Dziban: Balancing agency & automation in visualization design via anchored recommendations," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2020, 751:1–751:12.

[114] E. Loper and S. Bird, "Nltk: The natural language toolkit," in *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, Association for Computational Linguistics, 2002, pp. 63–70.

[115] C. T. Lopes, M. Franz, F. Kazi, S. L. Donaldson, Q. Morris, and G. D. Bader, "Cytoscape web: An interactive web-based network browser," *Bioinformatics*, vol. 26, no. 18, pp. 2347–2348, 2010.

[116] V. Lopez, M. Fernández, E. Motta, and N. Stieler, "PowerAqua: Supporting users in querying and exploring the semantic web," *Semantic Web*, vol. 3, no. 3, pp. 249–265, 2012.

[117] A. Lundgard, C. Lee, and A. Satyanarayan, "Sociotechnical considerations for accessible visualization design," in *2019 IEEE Visualization Conference (VIS)*, IEEE, 2019, pp. 16–20.

[118] A. M. MacEachren, G. Cai, R. Sharma, I. Rauschert, I. Brewer, L. Bolelli, B. Shaparenko, S. Fuhrmann, and H. Wang, "Enabling collaborative geoinformation access and decision-making through a natural, multimodal interface," *International Journal of Geographical Information Science*, vol. 19, no. 3, pp. 293–317, 2005.

[119] J. Mackinlay, P. Hanrahan, and C. Stolte, "Show me: Automatic presentation for visual analysis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1137–1144, 2007.

[120] T. L. Magnanti and R. T. Wong, "Network design and transportation planning: Models and algorithms," *Transportation Science*, vol. 18, no. 1, pp. 1–55, 1984.

[121] T. Major and R. C. Basole, "Graphicle: Exploring units, networks, and context in a blended visualization approach," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 576–585, 2019.

[122] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit," in *ACL (System Demonstrations)*, 2014, pp. 55–60.

[123] G. L. Martin, "The utility of speech input in user-computer interfaces," *International Journal of Man-Machine Studies*, vol. 30, no. 4, pp. 355–375, 1989.

[124] J.-C. Martin, *Tycoon: Theoretical framework and software tools for multimodal interfaces*, 1998.

[125] O. Mason and M. Verwoerd, "Graph theory and networks in biology," *IET Systems Biology*, vol. 1, no. 2, pp. 89–119, 2007.

[126] D. R. McGee and P. R. Cohen, "Creating tangible interfaces by augmenting physical objects with multimodal language," in *Proceedings of the International conference on Intelligent User Interfaces*, ACM, 2001, pp. 113–119.

[127] D. R. McGee, P. R. Cohen, and L. Wu, "Something from nothing: Augmenting a paper-based work practice via multimodal interaction," in *Proceedings of Designing Augmented Reality Environments (DARE)*, 2000, pp. 71–80.

[128] D. R. McGee and P. R. Cohen, "Augmenting environments with multimodal interaction," Ph.D. dissertation, OGI School of Science & Engineering at OHSU, 2003.

[129] *Microsoft Power BI Q&A*, https://docs.microsoft.com/en-us/power-bi/consumer/end-user-q-and-a, 2020.

[130] G. A. Miller, "WordNet: a lexical database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[131] J. Moody, "The structure of a social science collaboration network: Disciplinary cohesion from 1963 to 1999," *American Sociological Review*, vol. 69, no. 2, pp. 213–238, 2004.

[132] D. Moritz, C. Wang, G. L. Nelson, H. Lin, A. M. Smith, B. Howe, and J. Heer, "Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 438–448, 2018.

[133] J. Mu and A. Sarkar, "Do we need natural language?: Exploring restricted language interfaces for complex domains.," in *Extended Abstracts of the Conference on Human Factors in Computing Systems*, ACM, 2019.

[134] T. Munzner, "A nested model for visualization design and validation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 921–928, 2009.

[135] C. Myers, A. Furqan, J. Nebolsky, K. Caro, and J. Zhu, "Patterns for how users overcome obstacles in voice user interfaces," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2018, 6:1–6:7.

[136] D. Norman, *The design of everyday things: Revised and expanded edition*. Constellation, 2013.

[137] S. Oviatt, "Multimodal interfaces for dynamic interactive maps," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 1996, pp. 95–102.

[138] S. Oviatt, P. Cohen, L. Wu, L. Duncan, B. Suhm, J. Bers, T. Holzman, T. Winograd, J. Landay, J. Larson, *et al.*, "Designing the user interface for multimodal speech and pen-based gesture applications: State-of-the-art systems and future research directions," *Human-computer interaction*, vol. 15, no. 4, pp. 263–322, 2000.

[139] S. Oviatt and P. Cohen, "Perceptual user interfaces: Multimodal interfaces that process what comes naturally," *Communications of the ACM*, vol. 43, no. 3, pp. 45–53, 2000.

[140] S. Oviatt, R. Coulston, S. Tomko, B. Xiao, R. Lunsford, M. Wesson, and L. Carmichael, "Toward a theory of organized multimodal integration patterns during human-computer interaction," in *Proceedings of the International Conference on Multimodal Interfaces*, ACM, 2003, pp. 44–51.

[141] S. Oviatt, A. DeAngeli, and K. Kuhn, "Integration and synchronization of input modes during multimodal human-computer interaction," in *Referring Phenomena in a Multimedia Context and their Computational Treatment*, ACL, 1997, pp. 1–13.

[142] S. Oviatt, B. Schuller, P. Cohen, D. Sonntag, G. Potamianos, and A. Krüger, *The Handbook of Multimodal-Multisensor Interfaces, Volume 3: Language Processing, Software, Commercialization, and Emerging Directions*. Morgan & Claypool, 2019.

[143] S. Oviatt, "Multimodal interactive maps: Designing for human performance," *Human-computer interaction*, vol. 12, no. 1, pp. 93–129, 1997.

[144] S. Oviatt, "Ten myths of multimodal interaction," *Communications of the ACM*, vol. 42, no. 11, pp. 74–81, 1999.

[145] S. Oviatt, "Multimodal interfaces," *The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications*, vol. 14, pp. 286–304, 2003.

[146] B. Patnaik, A. Batch, and N. Elmqvist, "Information olfactation: Harnessing scent to convey data," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 726–736, 2018.

[147] R. Pausch and J. H. Leatherby, "An empirical study: Adding voice input to a graphical editor," in *Journal of the American Voice Input/Output Society*, Citeseer, 1991.

[148] C. Perin, "Direct manipulation for information visualization," Ph.D. dissertation, 2014.

[149] W. A. Pike, J. Stasko, R. Chang, and T. A. O'connell, "The science of interaction," *Information Visualization*, vol. 8, no. 4, pp. 263–274, 2009.

[150] A.-M. Popescu, O. Etzioni, and H. Kautz, "Towards a theory of natural language interfaces to databases," in *Proceedings of the International Conference on Intelligent User Interfaces*, ACM, 2003, pp. 149–157.

[151] M. Porcheron, J. E. Fischer, S. Reeves, and S. Sharples, "Voice interfaces in everyday life," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2018, pp. 1–12.

[152] A. J. Pretorius, H. C. Purchase, and J. T. Stasko, "Tasks for multivariate network analysis," in *Multivariate Network Visualization*, Springer, 2014, pp. 77–95.

[153] J. Raskin, "Meanings, modes, monotony, and myths," in *The humane interface: new directions for designing interactive systems*, Addison-Wesley, 2000, pp. 33–70.

[154] J. C. Roberts, P. D. Ritsos, S. K. Badam, D. Brodbeck, J. Kennedy, and N. Elmqvist, "Visualization beyond the desktop–the next big thing," *IEEE Computer Graphics and Applications*, vol. 34, no. 6, pp. 26–34, 2014.

[155] H. Romat, N. Henry Riche, K. Hinckley, B. Lee, C. Appert, E. Pietriga, and C. Collins, "ActiveInk: (th) inking with data," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2019, pp. 1–13.

[156] R. E. Roth, "An empirically-derived taxonomy of interaction primitives for interactive cartography and geovisualization," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 12, pp. 2356–2365, 2013.

[157] S. F. Roth and J. Mattis, "Data characterization for intelligent graphics presentation," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 1990, pp. 193–200.

[158] J. M. Rzeszotarski and A. Kittur, "Kinetica: Naturalistic multi-touch data visualization," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2014, pp. 897–906.

[159] R. Sadana and J. Stasko, "Designing and implementing an interactive scatterplot visualization for a tablet computer," in *Proceedings of the International Working Conference on Advanced Visual Interfaces*, ACM, 2014, pp. 265–272.

[160] R. Sadana and J. Stasko, "Expanding selection for information visualization systems on tablet devices," in *Proceedings of the International Conference on Interactive Surfaces and Spaces*, ACM, Nov. 2016, pp. 149–158.

[161] R. Sadana and J. Stasko, "Designing multiple coordinated visualizations for tablets," *Computer Graphics Forum*, vol. 35, no. 3, pp. 261–270, 2016.

[162] B. Saket, A. Endert, and Ç. Demiralp, "Task-based effectiveness of basic visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 7, pp. 2505–2512, 2018.

[163] B. Saket, S. Huron, C. Perin, and A. Endert, "Investigating direct manipulation of graphical encodings as a method for user interaction," *IEEE transactions on visualization and computer graphics*, vol. 26, no. 1, pp. 482–491, 2019.

[164] B. Saket, H. Kim, E. T. Brown, and A. Endert, "Visualization by demonstration: An interaction paradigm for visual data exploration," *IEEE transactions on visualization and computer graphics*, vol. 23, no. 1, pp. 331–340, 2017.

[165] B. Saket, P. Simonetto, and S. Kobourov, "Group-level graph visualization taxonomy," *arXiv preprint arXiv:1403.7421*, 2014.

[166] A. Saktheeswaran, A. Srinivasan, and J. Stasko, "Touch? speech? or touch and speech? investigating multimodal interaction for visual network exploration and analysis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 6, pp. 2168–2179, 2020.

[167] M. W. Salisbury, J. H. Hendrickson, T. L. Lammers, C. Fu, and S. A. Moody, "Talk and draw: Bundling speech and graphics," *Computer*, vol. 23, no. 8, pp. 59–65, 1990.

[168] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher, "What do we talk about when we talk about dashboards?" *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 682–692, 2018.

[169] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer, "Vega-Lite: A grammar of interactive graphics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 341–350, 2016.

[170] S. Schmidt, M. A. Nacenta, R. Dachselt, and S. Carpendale, "A set of multi-touch graph interaction techniques," in *Proceedings of the International Conference on Interactive Tabletops and Surfaces*, ACM, 2010, pp. 113–116.

[171] M. Schwab, J. Tompkin, J. Huang, and M. A. Borkin, "Easypz. js: Interaction binding for pan and zoom visualizations," in *Proceedings of IEEE VIS: Short Papers*, 2019, pp. 31–35.

[172] V. Setlur, S. E. Battersby, M. Tory, R. Gossweiler, and A. X. Chang, "Eviza: A natural language interface for visual analysis," in *Proceedings of the Annual Symposium on User Interface Software and Technology*, ACM, 2016, pp. 365–377.

[173] V. Setlur, M. Tory, and A. Djalali, "Inferencing underspecified natural language utterances in visual analysis," in *Proceedings of the International Conference on Intelligent User Interfaces*, ACM, 2019, pp. 40–51.

[174] A. Sharma, S. Madhvanath, A. Shekhawat, and M. Billinghurst, "Mozart: A multimodal interface for conceptual 3d modeling," in *Proceedings of the international conference on multimodal interfaces*, 2011, pp. 307–310.

[175] R. Sharma, V. I. Pavlović, and T. S. Huang, "Toward multimodal human–computer interface," in *Advances in image processing and understanding: a festschrift for Thomas S Huang*, World Scientific, 2002, pp. 349–365.

[176] B. Shneiderman, "Direct manipulation: A step beyond programming languages," *Sparks of innovation in human-computer interaction*, vol. 17, pp. 57–69, 1993.

[177] B. Shneiderman and P. Maes, "Direct manipulation vs. interface agents," *interactions*, vol. 4, no. 6, pp. 42–61, 1997.

[178] B. Shneiderman, "Dynamic queries for visual information seeking," *IEEE Software*, vol. 11, no. 6, pp. 70–77, 1994.

[179] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *Proceedings of IEEE Symposium on Visual Languages*, IEEE, 1996, pp. 336–343.

[180] R. Sicat, J. Li, J. Choi, M. Cordeil, W.-K. Jeong, B. Bach, and H. Pfister, "Dxr: A toolkit for building immersive data visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 715–725, 2018.

[181] K. A. Siek, Y. Rogers, and K. H. Connelly, "Fat finger worries: How older and younger users physically interact with pdas," in *IFIP Conference on Human-Computer Interaction*, Springer, 2005, pp. 267–280.

[182] R. Spence, *Information visualization*. Springer, 2001, vol. 1.

[183] A. Srinivasan, B. Lee, N. Riche, S. Drucker, and K. Hinckley, "InChorus: Designing consistent multimodal interactions for data visualization on tablet devices," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2020, 653:1–653:13.

[184] A. Srinivasan, M. Dontcheva, E. Adar, and S. Walker, "Discovering natural language commands in multimodal interfaces," in *Proceedings of the International Conference on Intelligent User Interfaces*, ACM, 2019, pp. 661–672.

[185] A. Srinivasan, S. M. Drucker, A. Endert, and J. Stasko, "Augmenting visualizations with interactive data facts to facilitate interpretation and communication," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 672–681, 2019.

[186] A. Srinivasan, B. Lee, and J. T. Stasko, "Interweaving multimodal interaction with flexible unit visualizations for data exploration," *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[187] A. Srinivasan, H. Park, A. Endert, and R. C. Basole, "Graphiti: Interactive specification of attribute-based edges for network modeling and visualization," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 226–235, 2017.

[188] A. Srinivasan and J. Stasko, "Natural language interfaces for data analysis with visualization: Considering what has and could be asked," in *Proceedings of EuroVis*, vol. 17, 2017, pp. 55–59.

[189] A. Srinivasan and J. Stasko, "Orko: Facilitating multimodal interaction for visual exploration and analysis of networks," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 511–521, 2017.

[190] A. Srinivasan and J. Stasko, "How to ask what to say?: Strategies for evaluating natural language interfaces for data visualization," *IEEE Computer Graphics and Applications*, vol. 40, no. 4, pp. 96–103, 2020.

[191] H. Subramonyam and E. Adar, "Smartcues: A multitouch query approach for details-on-demand through dynamically computed overlays," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 597–607, 2018.

[192] H. Subramonyam, W. Li, E. Adar, and M. Dontcheva, "TakeToons: Script-driven performance animation," in *Proceedings of the Annual Symposium on User Interface Software and Technology*, ACM, 2018, pp. 663–674.

[193] Y. Sun, J. Leigh, A. Johnson, and S. Lee, "Articulate: A semi-automated model for translating natural language queries into meaningful visualizations," in *International Symposium on Smart Graphics*, Springer, 2010, pp. 184–195.

[194] *Tableau*, https://www.tableau.com/, 2020.

[195] *Tableau Ask Data*, https://www.tableau.com/products/new-features/ask-data, 2020.

[196] J. Thompson, A. Srinivasan, and J. Stasko, "Tangraphe: Interactive exploration of network visualizations using single hand, multi-touch gestures," in *Proceedings of the International Conference on Advanced Visual Interfaces*, 2018, pp. 1–5.

[197] *ThoughtSpot*, https://www.thoughtspot.com/product, 2020.

[198] M. Tory and V. Setlur, "Do what i mean, not what i say! design considerations for supporting intent and context in analytical conversation," in *Conference on Visual Analytics Science and Technology (VAST)*, IEEE, 2019, pp. 93–103.

[199] E. Tse, C. Shen, S. Greenberg, and C. Forlines, "Enabling interaction with single user applications through speech and gestures on a multi-user tabletop," in *Proceedings of the International Conference on Advanced Visual Interfaces*, 2006, pp. 336–343.

[200] A. Van Dam, "Post-wimp user interfaces," *Communications of the ACM*, vol. 40, no. 2, pp. 63–67, 1997.

[201] J. J. Van Wijk, "The value of visualization," in *Proceedings of IEEE Visualization*, 2005, pp. 79–86.

[202] *Vega-embed*, https://github.com/vega/vega-embed.

[203] M. T. Vo and A. Waibel, "Multimodal human-computer interaction," *Proceedings of ISSD*, vol. 93, 1993.

[204] M. T. Vo and C. Wood, "Building an application framework for speech and pen input integration in multimodal learning interfaces," in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, IEEE, vol. 6, 1996, pp. 3545–3548.

[205] T. Von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. J. van Wijk, J.-D. Fekete, and D. W. Fellner, "Visual analysis of large graphs: State-of-the-art and future research challenges," in *Computer graphics forum*, Wiley Online Library, vol. 30, 2011, pp. 1719–1749.

[206] A. Wabel and M. Vo, "A multi-modal human-computer interface: Combination of gesture and speech recognition," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, vol. 150, 1993.

[207] J. Wagner Filho, C. M. Freitas, and L. Nedel, "Virtualdesk: A comfortable and efficient immersive information visualization approach," in *Computer Graphics Forum*, Wiley Online Library, vol. 37, 2018, pp. 415–426.

[208] J. Walny, B. Lee, P. Johns, N. H. Riche, and S. Carpendale, "Understanding pen and touch interaction for data exploration on interactive whiteboards," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2779–2788, 2012.

[209] *Web Speech API*, https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API, 2019.

[210] L. Wilkinson, *The grammar of graphics*. Springer Science & Business Media, 2013.

[211] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer, "Towards a general-purpose query language for visualization recommendation," in *Proceedings of the HILDA Workshop*, 2016, pp. 1–6.

[212] K. Wongsuphasawat, Z. Qu, D. Moritz, R. Chang, F. Ouk, A. Anand, J. Mackinlay, B. Howe, and J. Heer, "Voyager 2: Augmenting visual analysis with partial view specifications," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2017, pp. 2648–2659.

[213] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer, "Voyager: Exploratory analysis via faceted browsing of visualization recommendations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 649–658, 2016.

[214] L. Wu, S. L. Oviatt, and P. R. Cohen, "Multimodal integration-a statistical view," *IEEE Transactions on Multimedia*, vol. 1, no. 4, pp. 334–341, 1999.

[215] M. Wu, C. Shen, K. Ryall, C. Forlines, and R. Balakrishnan, "Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces," in *Proceedings of the International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'06)*, IEEE, 2006, 8–pp.

[216] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *Proceedings of the Annual meeting on Association for Computational Linguistics*, ACL, 1994, pp. 133–138.

[217] H. Xia, N. Henry Riche, F. Chevalier, B. De Araujo, and D. Wigdor, "DataInk: Direct and creative data-oriented drawing," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 2018, p. 223.

[218] X. Xu, J. Liao, and H. Shibata, "Drawing in talking: Using pen and voice for drawing system configuration figures in talking," in *Proceedings of the International Display Workshops. Fukuoka, Japan*, 2017.

[219] N. Yankelovich, "How do users know what to say?" *interactions*, vol. 3, no. 6, pp. 32–43, 1996.

[220] N. Yankelovich, G.-A. Levow, and M. Marx, "Designing speechacts: Issues in speech user interfaces," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, vol. 95, 1995, pp. 369–376.

[221] J. S. Yi, Y. ah Kang, J. Stasko, and J. Jacko, "Toward a deeper understanding of the role of interaction in information visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1224–1231, 2007.

[222] V. Yoghourdjian, D. Archambault, S. Diehl, T. Dwyer, K. Klein, H. C. Purchase, and H.-Y. Wu, "Exploring the limits of complexity: A survey of empirical studies on graph visualisation," *Visual Informatics*, vol. 2, no. 4, pp. 264–282, 2018.

[223] D. Yoon, N. Chen, F. Guimbretière, and A. Sellen, "Richreview: Blending ink, speech, and gesture to support collaborative document review," in *Proceedings of the Annual symposium on User Interface Software and Technology*, ACM, 2014, pp. 481–490.

[224] D. Yoon, N. Chen, B. Randles, A. Cheatle, C. E. Löckenhoff, S. J. Jackson, A. Sellen, and F. Guimbretière, "RichReview++: Deployment of a collaborative multimodal annotation system for instructor feedback and peer discussion," in *Proceedings of the Conference on Computer-Supported Cooperative Work & Social Computing*, ACM, 2016, pp. 195–205.

[225] B. Yu and C. T. Silva, "FlowSense: A natural language interface for visual data exploration within a dataflow system," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 1–11, 2019.

[226] E. Zgraggen, R. Zeleznik, and S. M. Drucker, "PanoramicData: Data analysis through pen & touch," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2112–2121, 2014.

[227] D. Zhang, Y. Suhara, J. Li, M. Hulsebos, Ç. Demiralp, and W.-C. Tan, "Sato: Contextual semantic type detection in tables," *arXiv preprint arXiv:1911.06311*, 2019.

[228] V. Zhong, C. Xiong, and R. Socher, "Seq2sql: Generating structured queries from natural language using reinforcement learning," *arXiv preprint arXiv:1709.00103*, 2017.

[229] M. X. Zhou and S. K. Feiner, "Visual task characterization for automated visual discourse synthesis," in *Proceedings of the Conference on Human Factors in Computing Systems*, ACM, 1998, pp. 392–399.